




Intent Detection for Virtual Reality Architectural Design

Romain Guillaume^{1,2} , Jérôme Pailhès¹, Elise Gruhier¹, Xavier Laville²,
Yvan Baudin², and Ruding Lou³

¹ Institut de Mécanique et d'Ingénierie (I2M), Avenue d'Aquitaine, 33170 Gradignan, France
romain.guillaume.2015@gadz.org

² Airbus Operations SAS, 316 Route de Bayonne, 31027 Toulouse, France

³ Arts et Métiers Institute of Technology LISPEN, HESAM
University, UBFC, 71100 Châlon-Sur-Saône, France

Abstract. In the context of optimization and cycles reduction for product design in industry, digital collaborative tools have a major impact, allowing an early stage integration of multidisciplinary challenges and oftentimes the search of global optimum rather than domain specific improvements. This paper presents a methodology for improving participants' implication and performance during collaborative design sessions through virtual reality (VR) tools, thanks to intention detection through body language interpretation. A prototype of the methodology is being implemented based on an existing VR aided design tool called DragonFly developed by Airbus. In what follows we will first discuss the choice of the different biological inputs for our purpose, and how to merge these multimodal inputs a meaningful way. Thus, we obtain a rich representation of the body language expression, suitable to recognize the actions wanted by the user and their related parameters. We will then show that this solution has been designed for fast training thanks to a majority of unsupervised training and existing pre-trained models, and for fast evolution thanks to the modularity of the architecture.

Keywords: Virtual reality · Machine learning · Body language · Intent detection · Computer-aided design

1 Introduction

Products complexification and the need for shorter time to market through a reduction of design loops foster the development of methods and tools enhancing product visualization and cross-disciplinary collaboration. Airbus has thus developed DragonFly – an internal virtual reality (VR) tool – to take advantage of an immersive environment at scale. This tool is primarily suitable for such architectural design tasks like space allocation and design reviews. Although efforts have been made to improve the interface, many experts refuse to learn to use a new design tool often associated to hard-to-master interfaces. We aim at fostering the adoption of the software by decreasing related learning phases. We plan to infer the actions intended by the users thanks to the analysis

© IFIP International Federation for Information Processing 2023

Published by Springer Nature Switzerland AG 2023

F. Noël et al. (Eds.): PLM 2022, IFIP AICT 667, pp. 420–430, 2023.

https://doi.org/10.1007/978-3-031-25182-5_41

of their body language while minimizing VR equipment intrusiveness. More precisely, due to the availability of different motion capture systems and due to the low equipment intrusiveness constraint, we focus in what follow on the language and posture analysis.

In this paper, we present a conceptual framework for a natural body language understanding specialized in the activities realized in DragonFly. Literature provides methods for inferences based on natural language and natural gesture separately [1], but also propose methods to synchronize and find relations between time series of different natures [2] – e.g. audio and video. Literature finally propose empirical descriptions of high level statistical analysis of body language (gesture and voice) [3]. Our proposition fills the gap between language and posture analysis, thus presenting a rich representation of the body language of a person over time and taking into account relationships between both inputs. In a first approach, we make the hypothesis that the action intended by a user can be inferred independently from any environmental variables and so that the information retrieved from the user body language is sufficient for this task.

2 Related Work

In this section, we discuss the different existing methods for body language understanding and the biological variables chosen in literature. More precisely, we focus on the body language empirical analysis, on its links with design actions in a virtual reality environment and we will show that a simple statistical analysis based on high-level features is not sufficient. However, we will see how the addition of a temporal component could help at eliciting simple user state of action in a context of pure gesture analysis. On the other hand, present how to handle speech through widely used “Natural Language Understanding” techniques but also how to identify entities – parameters of an action – in a sequence. Then, we tackle the multimodality of the input analyzing existing methods bringing together inputs of different natures. Finally we highlight existing solutions dealing with limited labelled datasets.

2.1 Body Language Empiric Analysis

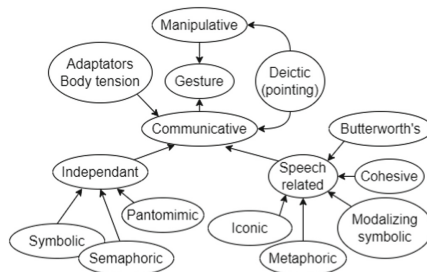


Fig. 1. Gesture classification (Vuletic T. et al. 2019)

Gestures are aimed at two purposes: manipulate, and communicate [4] (see Fig. 1). The first category includes the movements performed when interacting directly with a

physical, virtual or imaginary object generally with our hands whereas the second one relates to information sharing. More precisely, in this classification, the speech-related gestures are linked to communication; but the “modeling symbolic” gestures can carry meaningful element from a manipulative point of view when the speech itself carry such a manipulative message such as “draw a ball this big” spacing hands to a precise distance from each other thus suggesting the size of the ball. Moreover, a Computer Aided Design (CAD) task can be naturally performed with gesture different ways (see Fig. 2) [3] and some gesture patterns can correspond to multiple actions. This puts emphasis on the need for additional information to identify the action. Nevertheless, we can observe that metrics about the symmetry of the arm’s activity (not clearly defined in literature) and also the hand posture often give characteristic distributions (see Fig. 2 for arm’s activity) for the different gesture patterns according to each action to be performed. This tends to show that these metrics are good discriminators for our purpose, even though they are not sufficient. Finally, the most relevant descriptors for hand posture are geometrical (articulation angle) and topological (pinch) [5].

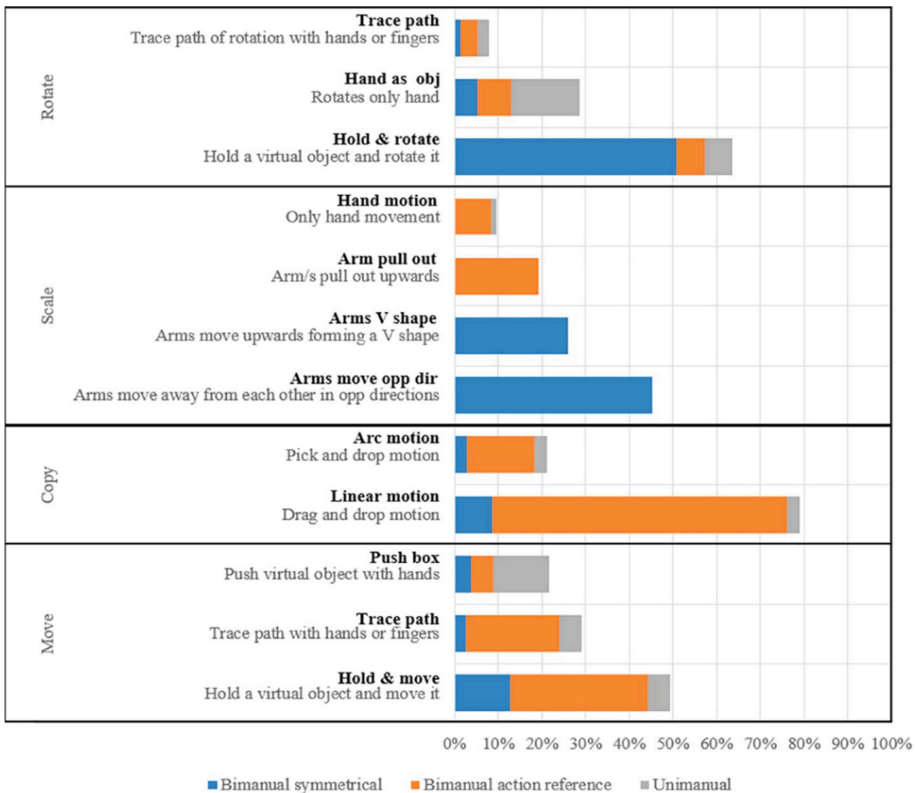


Fig. 2. Themes in manipulation groups (Khan S. and Tunçer B., 2019)

Besides, the distinction between simple user states (for instance waiting for a drink or in our case manipulating an object) can be deduced from other parameters: head

orientation, body trunk inclination and the relative position of the user from others [1]. More precisely, the trunk inclination appears to be correlated to the implication of the user and to his will to interact. Finally, the addition of voice to the gesture ease the understanding of the actions to be performed especially for non-tech users [3].

2.2 Sequence Analysis

As seen in sub Sect. (2.1), the static analysis of video and audio inputs seems to be insufficient for a good action inference. Literature proposes methods to add a temporal component to this analysis. Let's first consider sequence labelling, consisting of properties discrimination for each element of a sequence (for instance function identification for each word in a sentence), and then of sequence understanding which yields to a global property of the sequence (like the prediction of the evolution of a share).

Sequence Labelling. A commonly used solution is a hidden Markov model (HMM) [1, 6] which picks up in real time the most probable sequence of states in time according to previously observed variables. This artificial intelligence (AI) model relies on a transition table showing the most likely transitions from one state to another, and an emission table depicting the likelihood of a specific input for a given state. The model then learns from a tagged dataset these probabilities by counting the different transitions and emissions. The issue with this model is that the function optimized during training is not the likelihood of a given sequence of labels given a particular sequence of inputs, but rather the likelihood to get both at the same time. A solution that outperforms HMM in most cases for discriminative tasks is called conditional random fields (CRF) [7]. The idea there is to assume a log-linear probability distribution for the likelihood of a given sequence of labels given a particular sequence of inputs with respect to observations. The model is then trained to maximize this likelihood by optimizing parameters of this model (weights and biases) instead of frequencies in the HMM. CRF are thus harder to train but give better predictions. Even though both technologies have been primarily made for discrete observations, it is possible to convert inputs into Gaussian probabilities and thus to deal with continuous observations [1, 8].

Sequence Understanding. The idea here is to process the first element of the sequence and reuse this output as a part of the input for the next element [9], and so on until the end of the sequence. This design has been further improved with long-short term memory RNN (LSTM) [10] and gated recursive units (GRU) [11] thus keeping a longer track of the context given by the previous elements. Despite these improvements, these models suffer two flaws. Firstly, it remains hard to detect long term dependencies between elements [12]; secondly, sequence of inputs treatment cannot be parallelized due to the nested outputs for each new element computation. The current state of the art architecture for sequence understanding is based on the transformer architecture [13] initially developed for language tasks. This is a self-attention based auto-encoder [14] architecture that is to say that AI looks specifically for relations between each pairs of tokens. By construction, each interaction related to one token can be computed independently from the others, thus enabling parallel computing. In fact, the position of the token has to be encoded because the model is permutation independent: the input is treated as an unordered set

of inputs, thus allowing the detection of long-term dependencies. Thanks to the diversity of relation, transformers are used for high level image processing and transfer learning (pre-trained transformers [15]). The main limitation of this architecture is that it supports, by construction, a fixed maximum input sequence size and this size increase causes a quadratic growth of the computations.

2.3 Multimodality

A shared representation mixing several modalities (e.g. image and text) has been first done by concatenating all the data of the different modalities into one large input vector, but has been shown to be inefficient and not suitable for capturing dependencies and relations between the different modalities [16]. This is why research then focused on various techniques for a more reliable unification of different modalities. The most recent studies on cross-modality representation usually imply the projection of each modality input into a same semantic (or latent) space [16, 17, 18]. This projection can be done after a preprocessing of the raw inputs [18] and it seems more suitable to project low and medium-level features of the input in order to have a better similarity recognition between the inputs. However, the literature presents usually methods for generative tasks or for discriminative pairing tasks (e.g. associating text to a given image) that have not really been used for time constraint classification tasks. A model proposes to gather each modality in a same input vector where the nature of the modality is identified by a keyword before each new sequence [16].

2.4 Limited Labelled Dataset

Dealing with limited datasets in machine learning is more and more made possible by pre-trained auto-encoders (see Fig. 3) [19], popularized by the transformer architecture [13]. Auto-encoders are pairs of models – an encoder and a decoder – that are usually trained to reproduce the input after condensing the data in a latent space. This unsupervised approach is feature based: the models learn the principal key features of the data. The encoder learns the discriminant features contained in the data whereas the decoder learns the common features found in the dataset. Once trained, either the encoder is kept for discriminative tasks or the decoder is kept for generative tasks. The main advantage of this technique is that it does not require labelled data – the expected output being the input.

Once a model is pre-trained, additional layers are added on top of it to be trained for the initial task – this phase is called “fine-tuning”. This special case of transfer learning [19] speeds up speeds this second phase up because the learning effort is focused on the final task, and not on the learning of contextual data. For instance, in natural language processing, the pre-training learns the grammar and the semantics of the word whereas the fine tuning learns to classify a sentence.

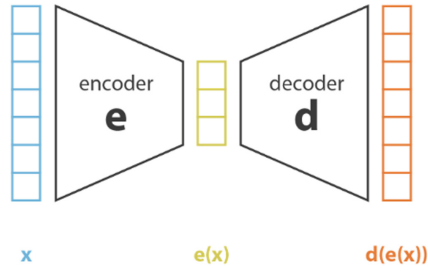


Fig. 3. Principle of a simple auto-encoder (Joseph Rocca, 2019)

3 Proposed Approach

In this section, we describe our new approach tackling multimodal design intent detection. We first describe the conceptual framework, the general architecture and then we describe more precisely each component. In this paper we don't tackle the translation of inferred actions into actual DragonFly commands.

3.1 Conceptual Framework

As a first approach, we propose the set of inputs described in Table 1. The proposed inputs contain all the information for the measures discussed in the literature review with additional information position-wise, so as to enable action parameters (e.g. a pointing direction) extraction. It allows also a clearer definition of such proposed measurements like the symmetricity of the activity of the arms. This inputs selection has two objectives. Firstly we reduce the dimensionality of the problem by exposing only relevant features to the model: doing so, it does not have to learn how to extract these lowest level features. Secondly, it helps understanding the model by making possible to see which input has been more active in a particular context, fostering the maintenance and further improvements for the model.

Table 1. Input selection proposition

Input type	Device	Measures
Hand posture	Leapmotion	For each hand: Flat, Flat hold, Fist, Thumb up, Index, L, C, Pinch or Undefined
Upper body posture	Kinect Azure	Relative positions in 3D to the body referential* of neck, shoulders, elbows, hands and pelvis
Head orientation	HTC Vive Pro	Relative Euler angles of the head with respect to the body referential*
Speech	HTC Vive Pro	Sequence of words obtained through a speech recognizer

*: the sagittal plane is defined as the median plan between shoulders and the frontal plane contains the segment from left shoulder to right shoulder

The outputs of our models correspond to the most used elementary capabilities of DragonFly. Several interviews with DragonFly users and experts, the observation of work sessions on the tool have drawn the following set of actions as our target classes:

- Change position (pan and rotate)
- Hide/unhide
- Grab
- Measure distance
- Make a section
- Select several objects
- Select parent object of selected object
- Take a picture
- Draw a primitive shape (box or cylinder)

3.2 General Architecture

We propose a similar architecture to what we have seen in literature [18] but introducing specific features dealing with the asynchronous nature of the different inputs – the different devices don’t deliver new data neither at the same time neither nor at the same frequencies – but also adapt the method to classification tasks and not to similarity cross-modal pairing or to generative tasks. The general approach is described in Fig. 4.

First of all, we define two modalities for our model: gesture and speech. We extract from these some low level features in the form of time series of one dimensional vector¹ and embed each token of each time series into a same dimensional space before being concatenated as seen in literature [10, 16]. This concatenation is then transformed into a rich and high level representation of these tokens with the full body language context. This representation is directly used for the action classification and each of the new rich features are concatenated to their corresponding low level features in order to identify the most relevant features through modality specific conditional random fields (CRF). Finally, a solver gathers the extracted actions and parameters to make a decision.

In addition, we take into account the non-random chaining of actions during a session as testified during the sessions observations and the interviews: we propose an additional input for the unified modality transformer encoding the previous action performed by the user. This idea is also motivated by the results obtained in literature [1] using Hidden Markov Models (HMM). In fact, this implementation tends to mimic the behavior of a transition diagram by adding information about the previous state – in our case the states are the different possible actions (see Sect. 3.1) – to find the most probable following action.

3.3 Component Details

Gesture Low Level Features. The low level representation of a user’s gesture is built by taking regular snapshots of a one dimensional posture vector to create a sequence

¹ The frequencies of gestures’ features and speeches’ ones are uncorrelated and thus may be completely different (see **Shared Representation.**).

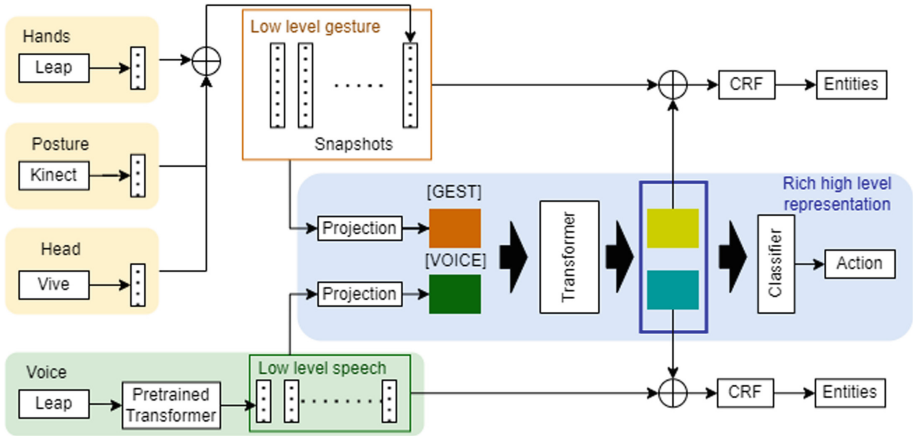


Fig. 4. General architecture of the proposed solution (the last detected action and the solver described below are not represented)

of postures. The frequency of these snapshots may be fine-tuned but we can deduce its order of magnitude considering two criteria. First of all, the interval between snapshots must be sufficient to have a real change of the vector between two consecutive shots. We discuss below the update of this vector but it is directly dependent on the frequencies of involved devices. The slowest device is the Kinect Azure with 30 Hz. Moreover, we observe empirically that an action takes around 5 s to be performed by a regular user – rounded to 10 s to ensure a better context understanding; the size allocated in the shared representation transformer is in the order of 100 tokens. We can thus predict a 10 Hz frequency for this second estimation, giving us a pretty good estimation of the possible range of the snapping frequency: from 10 to 30 Hz.

The posture vector is obtained by the asynchronous concatenation of the different inputs described in Table 1. Each input is obtained as follows: the hands postures are presented as a one hot encoding vector – a vector filled with zeros with a one at the position of the corresponding class – which is obtained from Boolean operations on the geometric and topologic characteristics of each hand (the detailed operations are not presented in this paper) [5]; the relative positions of the upper body articulation are concatenated in a fixed order and all the dimensions are normalized by the pelvis-neck distance; the head Euler angles in radians are also normalized by a factor $\frac{\pi}{4}$. Each part of the vector updates the global posture vector at its device frequency.

Speech Low Level Features. Once the sound has been captured by the HTC Vive Pro microphone, it goes through a voice recognizer (we use the built-in C# library System.Speech.Recognition) to obtain a list of strings. We then use a BERT pre-trained transformer [19] accessible online to get our low-level features for the voice input.

Shared Representation. This part of the process runs in a separate thread. It creates a rich body language representation of the different low level features. First, the low-level features are loaded and projected on a same dimensionality plane by an embedding layer

(the dimensionality is primarily set to the original dimension proposed in the transformer method [13] or 512×1 for each individual low-level feature).

Then, as proposed by the literature, the different modality sequences are put together using keyword separators – for our purpose we define [GEST] and [SPCH] at the beginning of each modality sequence, [PREV] at the beginning of the previous action embedding and [SEP] at the junction between two series. We define a minimum length for each modality. A standard adult communication being around 150 words per minute in English and a common action taking empirically around 5 s, we propose a maximum number of speech-allocated tokens of at least 50 (or about 15 s of normal speech), to capture more easily context dependencies and fast speeches. Similarly for gesture features, considering a snap frequency (see above) of 20 Hz, we propose to dedicate at least 300 tokens to gesture features. Finally, we proposed the following input word considering a 512 long input sequence for the transformer: “GEST[432 gesture tokens]SEPI[74 speech tokens]SEPIP[previous action token]”.

The sequence described above is then transformed by a unified modal transformer [16] to reveal a rich cross-modality representation of each of the input features. More precisely, each input feature is projected on a same semantic plane. This transformer can be pre-trained an unsupervised manner using its auto-encoder form with data gathered during real sessions. Indeed, the previous layer of the solution is already trained so we can gather words during various sessions before using them as a training dataset. The operation is repeated as soon as the following classification and parameter extraction are done, independently from the low level features update frequencies.

Classification and Parameters Extraction. The previous representation is directly used by a classifier to infer the action. We propose to use a simple sparse linear shallow neural network trained on labelled data for fast supervised training.

For the parameters extraction, we use residual connections as proposed in literature [18] by concatenating the low level features with their corresponding rich representation computed above. These augmented features go through conditional random fields trained with labelled data. The labels of the dataset correspond to the different natures of arguments of the different actions. The construction of this dataset needs discussions with the users, so as to identify which part of their body language expression at which moment determines the parameter – for instance a pointing finger can be the direction of an object: this posture is then inferred as an object parameter.

Once the action and the parameters have been extracted, a solver uses this data alongside the rich representation, so as to decide Dragonfly action is to be performed. It first decides if the user has finished to communicate his intent or not and then if there are missing parameters or conflicts. It finally pilots DragonFly accordingly.

4 Conclusion and Perspectives

This paper presents a method for architectural design intent recognition in VR yet to be implemented. This model does not take into account the virtual environment context and is not able to infer several intents at the same time. Nevertheless, it deals with the asynchrony of the input, and with the multimodality used for augmented representation while having a modular architecture for future improvements.

A first prototype is currently under development and will be built incrementally alongside with the methodology presented in this paper. This prototype will have a limited vocabulary to test the relevance of our model and retrieve the missing order of magnitude and hyper-parameters of our current approach. We will also define the precise architecture of the solver. Finally, we would like to define relevant environmental variables to be added to our model to increase even further the F1-score of the model.

We are also preparing a statistical analysis of user's body language when performing design actions in DragonFly. The objectives are to build a reference to assess the recognition performance of our solution, to validate the assumptions made on the biological variables we consider for our model and identifying confusing factors not highlighted by the literature. Moreover, it builds a first dataset for the training of our prototypes. Preliminary results show the necessity of having an assistant-like bot mainly because of the difficulty to speak naturally to a screen without expecting an answer.

References

1. Gaschler, A., Jentzsch, S., Giuliani, M., Huth, K., de Ruyter, J., Knoll, A.: Social behavior recognition using body posture and head pose for human-robot interaction. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2128–2133. IEEE, Vilamoura-Algarve, Portugal (2012)
2. Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., Ng, A.Y.: Multimodal deep learning. In: 28th International Conference on Machine Learning (ICML), pp. 689–696 (2011)
3. Khan, S., Tunçer, B.: Gesture and speech elicitation for 3D CAD modeling in conceptual design. *Autom. Constr.* **106**, 102847 (2019)
4. Vuletic, T., Duffy, A., Hay, L., McTeague, C., Campbell, G., Greal, M.: Systematic literature review of hand gestures used in human computer interaction interfaces. *Int. J. Hum. Comput. Stud.* **129**, 74–94 (2019)
5. Laviola, J.J.R., Zeleznik, R.C.: Flex and pinch: a case study of whole hand input design for virtual environment interaction. In: Second IASTED International Conference on Computer Graphics and Imaging, Innsbruck, Austria, pp. 221–225 (1999)
6. Ghogh, B., Karray, F., Crowley, M.: Hidden markov model: tutorial. *engrXiv* (2019)
7. Lafferty, J., McCallum, A., Pereira, F.C.N.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: 18th International Conference on Machine Learning (ICML), pp. 282–289 (2001)
8. Bevilacqua, F., Zamborlin, B., Sypniewski, A., Schnell, N., Guédy, F., Rasamimanana, N.: Continuous real-time gesture following and recognition. In: Kopp, S., Wachsmuth, I. (eds.) *Gesture in Embodied Communication and Human-Computer Interaction*, pp. 73–84. Springer, Berlin Heidelberg, Berlin, Heidelberg (2010)
9. Elman, J.L.: Finding structure in time. *Cogn. Sci.* **14**, 179–211 (1990)
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)

11. Cho, K., van Merriënboer, B., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: encoder-decoder approaches. [arXiv:1409.1259](#). (2014)
12. Gradient flow in recurrent nets: the difficulty of learning long term dependencies. In: *A Field Guide to Dynamical Recurrent Networks*. IEEE (2001)
13. Vaswani, A., et al.: Attention is all you need. In: *Advances in Neural Information Processing Systems*, pp. 5998–6008 (2017)
14. Kramer, M.A.: Nonlinear principal component analysis using auto associative neural networks. *AIChE J.* **37**(2), 233–243 (1991)
15. Esser, P., Rombach, R., Ommer, B.: Taming transformers for high-resolution image synthesis. In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12868–12878. IEEE, Nashville, TN, USA (2021)
16. Li, W., et al.: UNIMO: towards unified-modal understanding and generation via cross-modal contrastive learning. In: *Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing* (2021)
17. Radford, A., et al.: Learning transferable visual models from natural language supervision. [arXiv:2103.00020](#). (2021)
18. Liu, A.H., Jin, S., Lai, C.I.J., Rouditchenko, A., Oliva, A., Glass, J.: Cross-modal discrete representation learning. [arXiv:2106.05438](#). (2021)
19. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. [arXiv:1810.04805](#). (2019)