Deep 6-DOF Head Motion Prediction for Latency in Lightweight Augmented Reality Glasses

1st Seongwook Yoon
School of Electrical Engineering
Korea University
Seoul, Korea
swyoon@mpeg.korea.ac.kr

2nd Hee jeong Lim
School of Electrical Engineering
Korea University
Seoul, Korea
hjlim@mpeg.korea.ac.kr

3rd Jae Hyun Kim
School of Electrical Engineering
Korea University
Seoul, Korea
jhkim@mpeg.korea.ac.kr

4th Yun-Tae Kim
Multimedia Processing Lab
SAIT, Samsung
Suwon, Korea
ytae.kim@samsung.com

5th Hong-Seok Lee
Multimedia Processing Lab
SAIT, Samsung
Suwon, Korea
lhs1210@samsung.com

6th Sanghoon Sull
School of Electrical Engineering
Korea University
Seoul, Korea
sull@korea.ac.kr

2022 IEEE International Conference on Consumer Electronics (ICCE) | 978-1-6654-4154-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICCE53296.2022.9730327

***Abstract*—Computationally expensive rendering of virtual 3D objects in mixed reality applications of lightweight AR glasses can be performed by a remotely connected external server. However, nonnegligible 6DOF pose error caused by the remote rendering latency results in 3D visual inconsistency which can be hardly removed by 2D image correction using IMU. In this paper, we propose a novel 6DOF pose prediction algorithm based on learnable combination of consistent motion model and deep prediction. We formulate the combination of both as controlled residual learning and model ensemble. We build a dataset and demonstrate that our algorithm provides accurate prediction under 200ms.**

***Index Terms*—augmented reality, mixed reality, AR glasses, 6-DOF pose prediction, deep learning**

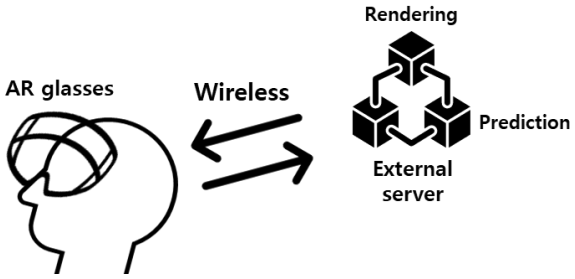

Fig. 1. Our scenario: For lightweight AR glasses, our prediction algorithm predicts a future pose to pre-render an image of virtual objects in external servers.

I. Introduction

Both augmented reality (AR) and virtual reality (VR) are essential components of metaverse. While VR immerses a user in a totally virtual world with near-eye displays, AR overlays the virtual world on a real world. Also, the extended version of AR, namely mixed reality (MR), presents mixture of both virtual and real worlds [1] tightly sharing 3D spatial relationship. AR glasses are a pair of near-eye transparent displays for AR/MR applications. A user wearing the AR glasses can directly see through the displays, and simultaneously observe virtual objects rendered on the displays.

In AR/VR systems, latency is important to the quality of user experience. To realistically display a stationary virtual object in a real scene, the 6DOF pose (position and orientation) at each time is necessary. To this end, simultaneous localization and mapping (SLAM) [2]–[6] algorithms are used to estimate the pose and partial 3D geometry of the real scene, but there inevitably occurs time delay between the estimated pose and the real pose.

For the displays of mobile phones, the latency problem is easily solved by slightly delaying video frames to use the pose estimate of the exact time instant. On the other hand, a VR 360° video streaming system predicts the pose over several seconds to load the video data in advance. Unlike short-term prediction algorithms for low latency under 50ms, it is possible to utilize more complex deep learning architectures including convolutional neural networks to process video frames [7]–[9] or a prior saliency prediction [10]. Also, other approaches [8], [11] formulate residual learning for gaze point prediction.

In the case of AR glasses, a future pose is predicted to render virtual objects, and then, the pre-rendered image is corrected by another prediction from the latest pose estimation using an inertial measurement unit (IMU). Using local rendering such as in self-contained AR glasses, both predictions are reasonably accurate so that the perception of the latency is negligible except for unusual motion. However, the AR glasses assisted by the remote rendering require more accurate prediction for longer latency. The actual pose difference caused by the longer latency results in irremovable 3D registration error of virtual objects, even though the image correction using 2D transformation is applied.

In this paper, we propose a novel 6-DOF pose prediction algorithm for the latency around 150ms, for the lightweight AR glasses which receive pre-rendered images according to the predicted pose. While conventional pose prediction algorithms

978-1-6654-4154-4/22/$31.00 ©2022 IEEE
Authorized licensed use limited to: Korea University. Downloaded on July 18,2022 at 06:49:55 UTC from IEEE Xplore. Restrictions apply.

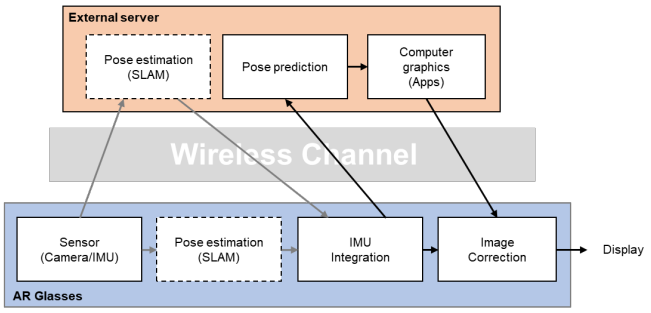

Fig. 2. Our system for AR glasses: Entire data flow of spatially aware applications using remote rendering in lightweight wireless AR glasses. Wherever the pose estimation is performed, the IMU integration can provide the latest pose to the pose prediction. The image is rendered by the predicted pose in the external server and corrected by the IMU integration in AR glasses.

for short-term under 50ms use a basic motion model assuming consistency of motion, two recent studies [12], [13] adopt deep learning based prediction models for a low-latency pre-rendering VR system using an external tracking device. They compare two motion models and two deep models for 6DOF head pose prediction with varying latency under 50ms. Compared to the deep models, the consistent motion model also shows comparable results for orientation prediction. Therefore, we also utilize the consistent motion model and propose a learnable combination of the consistent motion model and a deep model which is also learnable. We extend the simple combination by adding two predicted motions in two different ways. First, we formulate a residual learning controlled by fixed but learnable weighting parameters. Second, we re-formulate it as the weighted ensemble of the motion model and the uncontrolled residual deep model. Then, a shallow network learns the combination weights from the latest prediction error on a constraint that each component of the weights should be in the range of [0, 1]. Also, all components of 6-DOF motion in every prediction term are assigned with their respective combination weights determined by the network.

We build a dataset of head motion sequences using a stereo camera and an IMU attached to prototype AR glasses. Then, we perform experiments by simulating a scenario that the AR glasses are connected to an external server such as a paired mobile device, or even a cloud service. Our experimental results show that our learnable combination model improves the prediction accuracy, compared with the consistent motion model and other possible variations of combination.

II. Preliminaries

In this section, we present a desired scenario as shown in Fig.2 and related existing pose prediction algorithms.

A. Scenario

Due to the rendering latency of most AR/VR devices, 33ms for 30 fps videos, or less than 50ms is a typical maximum prediction term, whereas the wireless communication latency could be around 100ms and inconsistent due to the unstable channel. Although the accurate pose estimation using visual sensors also has non-negligible latency, the latency can be excluded from the prediction term, because the IMU integration can provide the latest pose, whether the pose estimation is done in the AR glasses or the external server. Thus, the total prediction term in our scenario is assumed to be about 150ms.

To display a virtual scene, the image is pre-rendered in the external server and transmitted to the AR glasses. Unless multiple images are pre-rendered simultaneously, a single pose prediction is performed. Thus, our prediction result has no need to be a multimodal distribution but covers diverse time intervals.

Also, we assume that an image-level correction algorithm transforms the pre-rendered image by using the latest pose from the IMU integration and a simple prediction over few milliseconds. Thanks to the low computational cost of the IMU integration and 2D transformation, it can be performed in the AR glasses to remove jitters. Though both the glasses and the server can afford the prediction algorithm, we assume that the prediction algorithm is performed in the external server to reduce computational overhead of the glasses.

B. IMU based prediction

General pose prediction models can be classified with their target motions and sensor usages. For AR glasses, head motion can be predicted using an IMU sensor. Incorporated with a visual SLAM algorithm, IMU can provide reasonably accurate estimates of current pose by integrating the measurement of acceleration and angular velocity from a given accurate initial velocity and sensor bias [14]. Also, the estimates can be extrapolated to predict future poses and motions. For example, a constant motion model is typically used for acceleration and angular velocity from the raw IMU measurement.

III. Method

In this section, we propose a long-term prediction algorithm which is a learnable combination of a consistent motion model and a deep prediction. The motion model is the same as the IMU based prediction model which assumes the consistency of acceleration and angular velocity. The overall architecture is presented in Fig.3.

A. Deep motion prediction

The deep architecture learns motion prediction from the trajectory which is the sequence of previous poses and motions. Recurrent neural network (RNN) [15]–[17] is appropriate to process the sequential data and extrapolate the sequence. The duration of informative input trajectory is tightly set to the maximum prediction term about 200ms. Using the IMU measurement, however, the trajectory has relatively high frequency for RNN to process every single time point in a given window. Instead, we divide the window into four non-overlapping subsequences and let the network process the sequence of the four subsequences to maintain proper length and dimension for both computational efficiency and training stability. In detail, the input window in Fig.3 consists of the four vectorized subsequences of consecutive ten time points

Authorized licensed use limited to: Korea University. Downloaded on July 18,2022 at 06:49:55 UTC from IEEE Xplore. Restrictions apply.

including position, orientation, velocity, angular velocity, and acceleration. Also, all the poses and motions in the window are defined relatively from the very first ones so that the inputs of the network are appropriately regularized for generalization.

The outputs of the network are also regularized as above, but the prediction type should be chosen between pose and motion unlike the input, because the correlation between them is difficult to be constrained implicitly in the network. Actually, we choose the same with the consistent motion model, i.e., as the motion model assumes the consistency of acceleration and angular velocity. The outputs of the prediction model also retrieve acceleration and angular velocity.

Even though highly deep features are not essential, a simple single layer RNN is insufficient to learn hidden motion patterns in the trajectory. To this end, we adopt a stacked structure which is cascaded recurrent neural units to extract deep sequential features. Another adaptation for motion prediction is to choose a leaky ReLU [18] activation function to make nonlinear hidden states behave more like physical features.

Formulating a residual learning for motion prediction, the output does not correspond to the actual motion but the residual of prediction and consistent motion. Note that the consistent motion represents the motion input at the latest time point. This formulation can be written as below:

$$\hat{\mathbf{m}}_{t+dt} = \mathbf{m}_t + \boldsymbol{\alpha}\mathbf{f}(\mathbf{X}_{\leq t}; \theta), \tag{1}$$

where $\mathbf{m}_t$ is the motion at discrete time t, $\hat{\mathbf{m}}_{t+dt}$ is predicted motion at the time after prediction term dt, $\mathbf{f}$ is the stacked RNN with parameter θ, and $\mathbf{X}_{\leq t}$ is the input window including previous poses and motions. An additional parameter vector $\boldsymbol{\alpha}$ can explicitly control how the model believes the consistency of the motion. For example, if $\boldsymbol{\alpha}$ is a zero vector, the model is equivalent to the consistent motion model. The set of control parameters $\boldsymbol{\alpha}$ is assigned to not only the prediction term dt but also to each component of motion such as the respective axes of acceleration or angular velocity. Also, the parameter $\boldsymbol{\alpha}$ can be jointly learned with the network parameters θ.

B. Learnable combination

The residual learning of motion prediction with the control parameter α in Eq.(1) can be interpreted as an ensemble of the linear combination of the consistent motion model and another prediction as below:

$$\mathbf{m}_t + \boldsymbol{\alpha}\mathbf{f}(\mathbf{X}_{\leq t}; \theta) = (\mathbf{1} - \boldsymbol{\alpha})\mathbf{m}_t + \boldsymbol{\alpha}(\mathbf{m}_t + \mathbf{f}(\mathbf{X}_{\leq t}; \theta)). \tag{2}$$

Thus, the combination weights $\boldsymbol{\alpha}$ balance between simple but stable prediction and more complex learned prediction, if its components are between 0 and 1. Every motion component of each prediction term has the respective weight just like the formulation of residual learning in Eq.(1).

Also, we can extend the fixed weights to be learned from plausible clues using a neural network. To this end, we adopt a multi-layer perceptron (MLP) [19] and use the information from the latest prediction result available. Specifically, we compute the difference between the predicted motion and the estimated motion at the same time point in the input window as below:

$$\boldsymbol{\alpha} = \mathbf{g}(\left[\hat{\mathbf{m}}_{t'} - \mathbf{m}_{t'}\right]_{t' \leq t}; \phi), \tag{3}$$

where $\mathbf{g}$ is the MLP network with parameter ϕ.

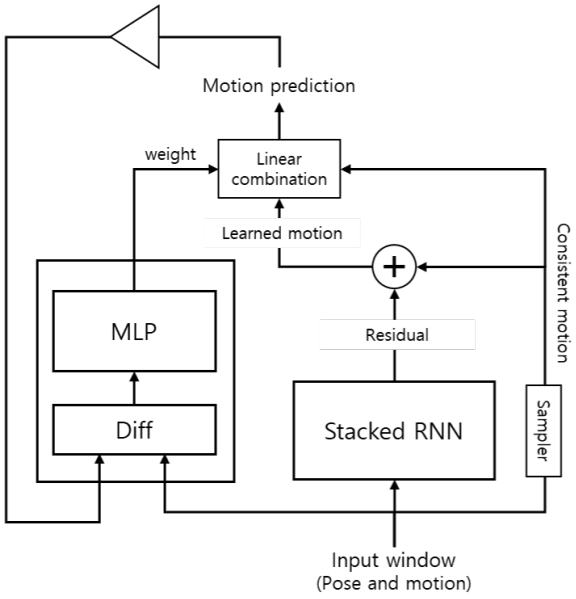

Fig. 3. Our network architecture: Learnable weighted combination of consistent motion model and deep motion prediction. The input window consists of four non-overlapping partitions of recent poses and motions. The learned motion is the sum of the consistent motion and the residual, and it is combined again with the consistent motion. The learnable combination weights are determined by the prediction error between the latest motion prediction and its corresponding motion in the input window.

C. Training procedure

The deep architecture in Fig. 3 is trained in a self-supervised manner, because additional annotation for future pose and motion is not necessary. The ground truth is easily obtained by any trajectory, if it can be estimated accurately by well-designed pose estimation algorithms [2]–[6]. While a lot of sequences have a small amount of sawtooth error caused by the IMU integration, it is sufficiently smaller than the prediction errors and does not induce the differential noise in the sequence of motion.

Because of two incompatible objectives for translation and rotation, the two errors should be appropriately merged to define both an evaluation metric and a loss function. Also, the errors are computed at the multiple prediction terms so that it is not trivial to train and evaluate the model. Fortunately, however, each of the various prediction terms shows similar tendency and a hyper-parameter $\lambda(> 0)$ balancing between the translation and rotation is sufficient to control the training procedure properly. Therefore, the loss function is defined as below:

$$L(\left[(\hat{\mathbf{m}}_{t'}, \mathbf{m}_{t'})\right]_{t<t'}) = \sum_{t<t'} \lambda|\hat{\mathbf{p}}_{t'} - \mathbf{p}_{t'}| + |\hat{\mathbf{q}}_{t'} - \mathbf{q}_{t'}|, \tag{4}$$

Authorized licensed use limited to: Korea University. Downloaded on July 18,2022 at 06:49:55 UTC from IEEE Xplore. Restrictions apply.

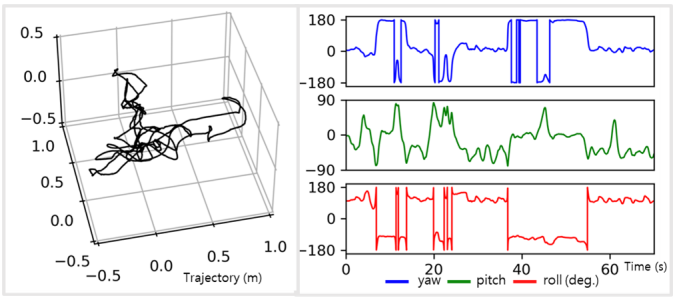

Fig. 4. Trajectory and rotation in one sequence: (a) Trajectory of a sequence taken when a person is walking while seeing multiple AR objects. (b) Yaw, pitch and roll of the trajectory in (a).

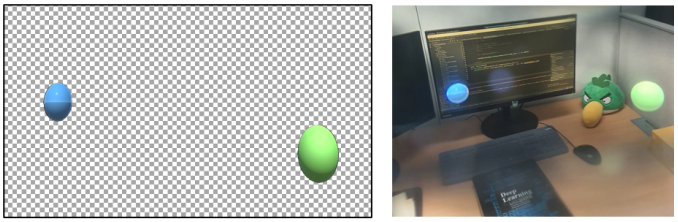

Fig. 5. Example of a mixed scene: (a) Two virtual colored spheres. (b) Mixing the virtual spheres with a real desk scene. Note that the image in (b) is captured by a camera in the position of a right eye.

where $\hat{\mathbf{m}}_t = (\hat{\mathbf{p}}_t, \hat{\mathbf{q}}_t)$ is the predicted motion which consists of translation and rotation, and $\mathbf{m}_t = (\mathbf{p}_t, \mathbf{q}_t)$ is its corresponding ground truth. Note that the rotation error is denoted as l_1-norm because we present the rotation with the Euler vector of which magnitude and direction encode the angle and axis of the rotation, respectively.

IV. Experiment

In this section, we describe several experiments using our own dataset based on simulated settings for the desired scenario. The details of the dataset and the implementation are described. Also, we investigate the evaluation results of the variations of prediction algorithms. Our prediction methods yield better prediction accuracy than baseline methods.

A. Dataset

Our dataset consists of total 24 sequences each of which duration is about 70 seconds. We use 18 sequences for training, remaining 6 sequences for test, respectively. For each sequence, a user wearing AR glasses observes one or more virtual objects, moving freely while sitting or walking around the virtual objects. The sequences were taken while different users are wearing AR glasses, differing also in the number, size, location, and movement of the virtual objects. An exemplary trajectory of the head motion is shown in Fig.4 and a mixed scene is shown in Fig.5.

B. Implementation

We use Intel Realsense T265 device [20] including a stereo camera and an IMU sensor and a prototype of AR glasses. Also, we use extended Kalman filter-based SLAM [21], which is modified not to include global optimization for delayed pose estimation, because it is efficient and quite accurate within 100ms.

For deep learning, we utilize an Nvidia RTX 1080Ti GPU and python TensorFlow library for training and quantitative evaluation, but we re-implement optimal C++ code to incorporate the prediction algorithm with the other part of implementation for AR glasses.

The stacked RNN has three LSTMs of which hidden state has 200, 200 and 6 units, respectively. Also, the network can be more lightweight without severe degradation. For example, a shallow and narrow network using simple RNN cells also works well. As described in Section III-A, each input sequence has four groups of ten time points and the smaller slope of the leaky ReLU is set to 0.5. For the training hyper-parameters, we set the learning rate to 5×10^{-5} and the hyper-parameter for the loss function in Eq.(4) to 0.04.

C. Evaluation

For quantitative evaluation, we calculate the prediction errors of various possible methods using the ground truth poses estimated by the SLAM algorithm. 'No prediction' refers to using previous pose so that the result is equivalent to the magnitude of pose variation. 'Consistent motion model' assumes the consistency of both acceleration and angular velocity which are obtained by the IMU sensor. 'Deep (w/o consistent motion model)' is a single deep prediction model not combined with the consistent motion model.

'Residual learning (fixed $\boldsymbol{\alpha}$)' learns the controllable residual of the consistent model so that there is no constraint on the control parameter $\boldsymbol{\alpha}$ in Eq.(1). 'Learnable combination (adaptive $\boldsymbol{\alpha}$)' adopts the formulation that all components of the combination weights are computed by the network and constrained to be within the interval [0, 1].

Also, in order to investigate the capacity of the network, 'Learnable comb. (narrow)' uses a narrow network with the half number of the hidden units, and 'Learnable comb. (shallow)' omits one of the hidden layers.

TABLE I
Prediction errors for 100, 150 and 200ms

Method	Translation (mm)	Rotation (deg)
No prediction	11.364 / 16.930 / 22.435	2.338 / 3.475 / 4.591
Consistent motion model	2.312 / 4.336 / 7.284	0.560 / 1.104 / 1.758
Deep (w/o consistent motion model)	2.382 / 4.242 / 6.676	1.194 / 1.896 / 2.663
Residual learning (fixed $\boldsymbol{\alpha}$)	2.258 / 4.005 / **6.337**	0.519 / 1.037 / 1.666
Learnable combination (adaptive $\boldsymbol{\alpha}$)	2.273 / 4.047 / 6.417	**0.509** / **1.015** / **1.632**
Learnable comb. (narrow)	2.276 / 4.101 / 6.605	0.519 / 1.034 / 1.657
Learnable comb. (shallow)	**2.253** / **4.003** / 6.338	0.518 / 1.025 / 1.651

Authorized licensed use limited to: Korea University. Downloaded on July 18,2022 at 06:49:55 UTC from IEEE Xplore. Restrictions apply.

In Table.I, various prediction methods are compared using their prediction errors for 100ms, 150ms, and 200ms, respectively. We can see that our learnable combination yields the performance gains with respect to the two errors of translation and rotation. Since the translation is predicted from the initial velocity and the predicted acceleration, various algorithms show comparable results in short prediction terms where the initial velocity can play a major role. Since the rotation is typically more dominant while wearing AR glasses, the result seems suitable for our scenario.

D. Discussion

To improve the prediction performance, we propose a personalization framework which allows additional training for personal trajectory data. Since the ground truth trajectory can be obtained without annotation, we can train the customized neural network for each user who wears the AR glasses for sufficient time around several minutes.

To this end, we simply test on the sequence of which user is identical to one of the users in training dataset. The results can be interpreted as an upper bound of prediction performance where the sophisticated online learning or incremental learning algorithms are utilized. Also, this simple personalization method is possible if the training is performed in an external server.

Table.II shows the personalization results using our learnable combination architecture. Note that the dataset is categorized differently from that of Table.I. 'Consistent motion model' does not require the training and personalization. 'No personal data' uses the training data of other users rather than the test user. 'Only personal data' utilizes only the test user's training data, and 'Mixed data' uses all available training dataset.

From the results, 'Mixed data' shows the best results but 'No personal data' has better results than 'Only personal data'. On the other hand, these results are simply in order of the amount of data. Whether the performance is largely dependent on the amount of data, it would be best to use mixed dataset if possible.

V. Conclusion

Assuming that AR glass is assisted by remote rendering to present spatially aware virtual objects, we have proposed a prediction algorithm to reduce the visual inconsistency caused by the latency of the remote rendering. However, the prediction performance should be further enhanced to satisfy commercial requirements of user experience, despite the challenging nature of the prediction task. Thus, we plan to develop more sophisticated prediction algorithms, for example, using additional input modalities such as latest images to extract scene dependency or online learning to adapt user-specific patterns.

TABLE II
Personalized prediction errors for 100, 150 and 200ms

Method	Translation (mm)	Rotation (deg)
No prediction	15.341 / 22.867 / 30.301	3.117 / 4.644 / 6.143
Consistent motion model	2.842 / 5.393 / 9.09	0.716 / 1.455 / 2.398
No personal data	2.759 / 4.925 / 7.809	**0.562** / 1.312 / 2.144
Only personal data	2.809 / 5.260 / 8.765	0.644 / 1.309 / 2.155
Mixed data	**2.753** / **4.891** / **7.724**	0.636 / **1.281** / **2.101**

References

[1] M. Speicher, B. D. Hall, and M. Nebeling, "What is mixed reality?" in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019, pp. 1–15.

[2] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE, 2007, pp. 225–234.

[3] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 3565–3572.

[4] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1271–1278.

[5] M. Turan, Y. Almalioglu, H. Gilbert, H. Araujo, E. Konukoglu, and M. Sitti, "Magnetic-visual sensor fusion based medical slam for endoscopic capsule robot," *arXiv preprint arXiv:1705.06196*, 2017.

[6] Y. Xu, Y. Ou, and T. Xu, "Slam of robot based on the fusion of vision and lidar," in *2018 IEEE International Conference on Cyborg and Bionic Systems (CBS)*. IEEE, 2018, pp. 121–126.

[7] X. Chen, A. T. Z. Kasgari, and W. Saad, "Deep learning for content-based personalized viewport prediction of 360-degree vr videos," *IEEE Networking Letters*, vol. 2, no. 2, pp. 81–84, 2020.

[8] M. F. R. Rondon, L. Sassatelli, R. Aparicio-Pardo, and F. Precioso, "Track: A new method from a re-examination of deep architectures for head motion prediction in 360-degree videos," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[9] Y. Zhu, G. Zhai, X. Min, and J. Zhou, "The prediction of saliency map for head and eye movements in 360 degree images," *IEEE Transactions on Multimedia*, vol. 22, no. 9, pp. 2331–2344, 2019.

[10] A. Nguyen, Z. Yan, and K. Nahrstedt, "Your attention is unique: Detecting 360-degree video saliency in head-mounted display for head movement prediction," in *Proceedings of the 26th ACM international conference on Multimedia*, 2018, pp. 1190–1198.

[11] Y. Xu, Y. Dong, J. Wu, Z. Sun, Z. Shi, J. Yu, and S. Gao, "Gaze prediction in dynamic 360° immersive videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[12] X. Hou, J. Zhang, M. Budagavi, and S. Dey, "Head and body motion prediction to enable mobile vr experiences with low latency," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–7.

[13] X. Hou and S. Dey, "Motion prediction and pre-rendering at the edge to enable ultra-low latency mobile 6dof experiences," *IEEE Open Journal of the Communications Society*, vol. 1, pp. 1674–1690, 2020.

[14] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2016.

[15] C. L. Giles, G. M. Kuhn, and R. J. Williams, "Dynamic recurrent neural networks: Theory and applications," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 153–156, 1994.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[17] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[18] A. L. Maas, A. Y. Hannun, A. Y. Ng *et al.*, "Rectifier nonlinearities improve neural network acoustic models." Citeseer.

[19] M. Minsky and S. A. Papert, *Perceptrons: An introduction to computational geometry*, 2017.

[20] *https://www.intelrealsense.com/tracking-camera-t265*.

Authorized licensed use limited to: Korea University. Downloaded on July 18,2022 at 06:49:55 UTC from IEEE Xplore. Restrictions apply.

[21] K. Sun, K. Mohta, B. Pfrommer, M. Watterson, S. Liu, Y. Mulgaonkar, C. J. Taylor, and V. Kumar, "Robust stereo visual inertial odometry for fast autonomous flight," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 965–972, 2018.

Authorized licensed use limited to: Korea University. Downloaded on July 18,2022 at 06:49:55 UTC from IEEE Xplore. Restrictions apply.