Pose Tracking vs. Pose Estimation of AR Glasses with Convolutional, Recurrent, and Non-Local Neural Networks: A Comparison

Ahmet Firintepe[1,2], Sarfaraz Habib[1], Alain Pagani[3], and Didier Stricker[2,3]

[1] BMW Group Research, New Technologies, Innovations, Garching (Munich), Germany {Ahmet.Firintepe,Sarfaraz.Habib}@bmwgroup.com
[2] TU Kaiserslautern, Germany
[3] German Research Center for Artificial Intelligence (DFKI), Kaiserslautern, Germany
{Alain.Pagani,Didier.Stricker}@dfki.de

Abstract. In this paper, we analyze various outside-in approaches for pose tracking and pose estimation of AR glasses. We first provide two frame-by-frame pose estimation approaches. The first one is a VGG-based CNN, while the second method is the state-of-the-art, ResNet-based AR glasses pose estimation method named GlassPoseRN. We then introduce LSTMs in the mentioned approaches to achieve AR glasses pose tracking. We compare methods with and without non-local blocks, which are theoretically promising for Pose Tracking as they consider non-local neighbor features in one image and among multiple images. We further include separable convolutions in some networks for comparison, which focus on maintaining the individual channels and therefore the triple images. We train and evaluate seven different algorithms on the HMDPose dataset. We observe a significant boost on the dataset from pose estimation to tracking approaches. Non-local blocks do not improve our performance further. The introduction of separable convolutions in our recurrent networks results in the best performance with an estimation error of 0.81° in orientation and 4.46mm in position. We reduce the error compared to the state-of-the-art by 76%. Our results suggest a promising approach for more immersive AR content for AR glasses in the car context, as high a 6-DoF pose accuracy improves the superimposition of the real world with virtual elements.

Keywords: Augmented Reality · Pose Tracking · Deep Learning.

1 Introduction

In recent years, Augmented Reality (AR) applications started to flourish in many new scenarios, and the simple laboratory or indoor experiments have extended to much more challenging use cases. In general, in order to ensure a convincing experience, it is paramount to localize the used smartphone or AR glasses with

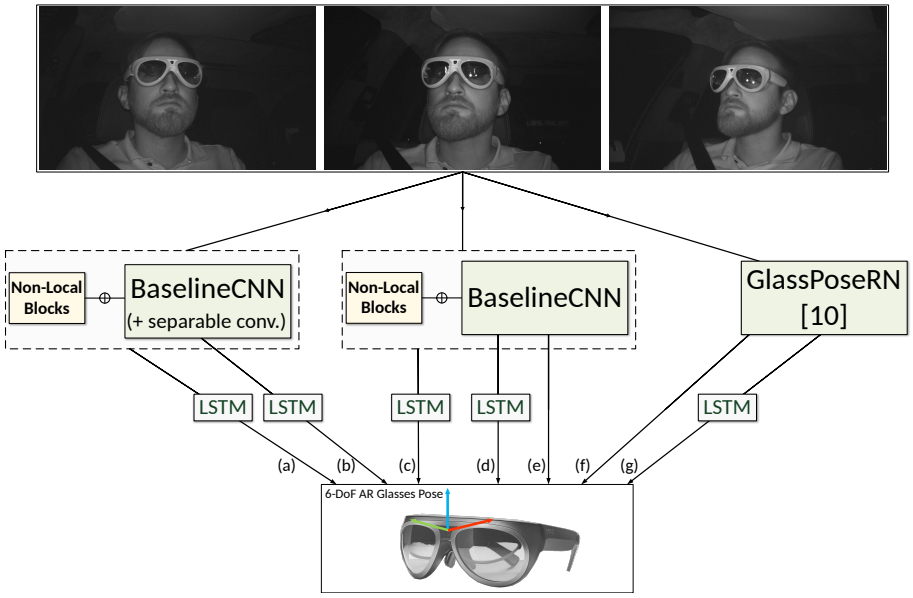

Fig. 1. An overview of our evaluation. We compare seven different pose estimation and tracking approaches on triple images of the HMDPose dataset: (e) and (f) show the baseline, frame-by-frame pose estimation methods, (f) being the state-of-the-art GlassPoseRN [13] approach. In (d) and (g), we introduce LSTMs to the baseline approaches. (c) depicts the standard CNN with non-local blocks and an LSTM. (b) shows a CNN variant with separable convolutions and an LSTM, while (a) extends the approach with non-local blocks.

a high precision in its environment. Usually, this is required in a highly accurate and latency-free way. Smartphones are more error-tolerant regarding tracking, as they are equipped with video-see-through displays. They have the advantage of frequency and speed adjustment of the real world shown through the display. AR glasses steadily matured over the years and are currently being used in the industry. In the future, consumer adaptation of AR glasses is highly likely. In contrast to smartphones, AR glasses mostly consist of optical-see-through displays, where the real world is perceived directly by the wearer. This elevates the latency and accuracy requirements for a realistic experience.

As larger AR glasses usually come with RGB or depth sensors and IMUs to enable inside-out tracking, it is the standard for pose estimation in static environments. In dynamic environments like airplanes or cars, IMUs register both head and vehicle movement. Thus, it is relatively challenging to utilize them in a standalone manner. In the case of a car driver, the optical sensors register mostly parts of the outside world, which delivers a limited set of features of the car interior for tracking.

In this scenario, outside-in tracking becomes a valuable option for tracking. Outside-in tracking is done frequently in case of head pose estimation. Similar

algorithms can be deployed for pose estimation of Head-Mounted Displays as done for head pose estimation. The head pose is different from the AR glasses, as the relation between head and glasses differs from user to user. This relation can change with slight glasses movement on the wearer's head. Thus, it is necessary to estimate the AR glasses directly. Algorithms adjusted to handle different types of glasses models can enable scalability to new types of models. Thus, tracking AR glasses through external sensors placed inside a car can lead to a wider adoption of AR glasses in cars.

Even though depth information improved pose estimation quality in recent works on pose estimation [39, 4, 2], they are still more costly than RGB or infrared (IR) cameras. Due to the changing lighting conditions in a driving scenario, RGB cameras can output significantly varying images. However, IR cameras come in handy due to their property of being less sensitive to change by lighting conditions compared to RGB images.

Recent works on AR glasses pose estimation [13, 14] on the HMDPose dataset [12] have shown accurate results for the pose estimation from a single and multi-view images, but did not attempt to use temporal coherence in the pose estimation. Our idea is to exploit the temporal continuity of the video sequences given in the dataset by proposing new learning-based pose tracking algorithms. We introduce five different alternatives taking advantage of the present continuous data. We first present Recurrent Neural Networks (RNNs) based on VGG- and ResNet backbones. We then introduce non-local blocks into our methods, a novel type of Neural Network maintaining object properties over time [37]. To the best of our knowledge, we are the first to use them in the pose estimation and tracking context. We furthermore introduce separable convolutions [7] into our networks for comparison. We finally evaluate them on the HMDPose dataset, where we compare them with CNN-only frame-by-frame methods, including the state-of-the-art "GlassPoseRN" method.
In detail, our contributions are:

- We present seven approaches based on combinations of three Neural Network categories for AR glasses pose estimation and tracking: convolutional, recurrent, and non-local Neural Networks.
- We introduce non-local blocks in our methods, being the first to utilize them for pose estimation. Additionally, we evaluate networks with separable convolutions instead of normal convolutions.
- We evaluate our approaches on the HMDPose dataset, where we outperform the state-of-the-art and reduce the error by 76%. Our detailed comparison of various methods enables us to recommend certain network types for AR glasses tracking.

In the remainder of the paper, we discuss the related work on object pose estimation and RNN-based pose estimation in Section 2. We present our AR glasses pose estimation and tracking techniques in Section 3. In Section 4, we discuss our evaluation on the HMDPose dataset, comparing our models against the state-of-the-art approach benchmarked on the dataset.

2 Related work

In this section, we review recent work on pose estimation from camera images. Some approaches either work with intensity images or depth maps or combine different data modalities.

2.1 Object Pose Estimation

6-DoF pose estimation approaches can primarily be divided into two categories. One category is direct pose regression, where the image is taken to estimate a pose directly by a Neural Network. The second category must first detect 2D targets and 2D-3D Object correspondence and then solve for Perspective-n-Point (PnP) problem for the 6-DoF pose. We shortly summarize methods for both categories.

The approaches that first detect the targeted object in the given image and subsequently solve a PnP problem for their 6-DoF pose, mostly dominate the state-of-the-art work in object pose estimation. In this category, keypoints-based and 2D-3D correspondence methods are popular. The keypoint-based methods either use the keypoints from the object's surface [28, 34, 35] or directly predict the 2D projections from 3D models of the object using furthest point algorithm [26, 33, 6]. The dense 2D-3D correspondence methods predict the corresponding 3D model point for each 2D pixel of the object and later use the PnP to get the 6-DoF pose [22, 40, 25]. Some methods also use additional networks to further refine the pose by using cropped images of an object.

However, adding processes like keypoints or other landmarks extraction involves chances of introducing errors. An alternative for 6-DoF pose estimation of an object is to regress it directly based on the image information only. PoseNet [19] introduces a deep CNN to estimate the 6-DoF camera pose based on a single RGB image as input. The network can predict the pose in an end-to-end manner with real-time performance. They argue that training individual networks to regress position and orientation separately performed poorly compared to when they were trained with full 6-DoF pose labels. PoseCNN [38] decouples the problem of pose estimation into estimating the translation and orientation separately. It uses a pre-trained VGG network [32] as a backbone for feature extraction and splits into three output branches. Two fully convolutional branches estimate center directions and the depth for every pixel of the image. The third branch consists of a ROI pooling and a fully connected architecture that regresses a quaternion describing the rotation for each region of interest. PoseConvCNN [5] further developed a fully convolutional architecture evolved from PoseCNN [38], and rather than ROI-based orientation prediction, they perform pixel-wise quaternion prediction, keeping the translations and ROI exactly the same from PoseCNN [38]. Recently, Peng et al. [26] also removed the ROI pooled orientation prediction branch and used 2D keypoints predicted using Hough-based voting to estimate the pose, creating a hybrid between the two major categories. In this approach, the ground truth 3D keypoints are generated using 3D models of target objects. The direct estimation of 3D rotations information is difficult because

of the non-linear rotation space. For this reason, other works avoid introducing pose refinement steps after directly estimating poses from monocular images. Li et al. [21] present a render and compare technique as a pose estimation step that improves the estimation only using the original RGB input. Although this can lead to accurate results, the pose refinement procedure requires additional computing time.

Regarding our specific use case of AR glasses pose estimation based on IR images, Firintepe et al. [13] developed and tested various methods on the HMD-Pose dataset. Their first method named "GlassPose" is a VGG-based CNN. They separate the estimation into two branches, estimating the orientation on cropped images of the head while using the full images for translation prediction. Their second approach "GlassPoseRN" is based on a ResNet-18 backbone, predicting the full 6-DoF pose based on the full images, outperforming GlassPose by a large margin. Another AR glasses pose estimation approach derives point clouds from IR images for further pose estimation [14]. A network was trained on triple IR input images in a semi-supervised fashion to generate a self-centered point cloud and to estimate the 3D position of the cloud. After training, the network can extract the point cloud representation from a single IR input image. Two pose regressors called P2R and P2P were proposed to estimate the rotational and translational poses given the point clouds. In P2R, a backbone extracts features followed by a rotation estimation module consisting of convolutional and dense layers. The predicted output represents the rotation in the 4D quaternion. In P2P, the model is extended with a voting process used to identify keypoints and extract descriptor vectors. The descriptive keypoints are fed to rotation and translation estimation modules to get the final 6-DoF pose.
We choose to compare our methods to the GlassPoseRN algorithm, as it performs better than the benchmarked state-of-the-art Regression via Classification head pose estimator [1] and a point cloud based object pose estimator named CloudPose [15] on the HMDPose dataset [13, 14].

2.2 RNN-based Pose Tracking

An alternative to frame-by-frame pose estimation is 6-DoF pose tracking based on RNNs. Inspired by You Only Look Once (YOLO) [29], Ning et al. [24] propose Rotated Logging (ROLO), in which LSTMs are used to learn sequential information in the high-level visual features with the region information. They use YOLO to learn features from images and detect objects' bounding boxes in the images. These extracted features and detected regions are then fed into the LSTM. ROLO only used the regression capability of RNN in the temporal domain and not its temporal correlation. Considering this, Zhang et al. [41] predict directions first based on RNN and later use the direction of the next frame to reduce the search for ROI. Once the search area is reduced, the object is detected only in that area. They perform this by utilizing a direction prediction model based on RNNs and a detection model later for tracking.

Several works in object pose estimation specifically focus on the head as an object. As our work is closely related to head pose tracking, we further discuss

related work in head pose estimation using RNNs. Most of the works used particle filters with face image renderer to track faces [23]. However, these filters require complex, problem-specific design and feature tuning. Work from Gu et al. in [16] presents a comparison between Bayesian filtering and RNNs while proposing an RNN-based approach to solve the tracking problem. They use a VGG network to regress the head pose Euler angles. Instead of improving single frame prediction by modifying the network structure, it focuses on using an RNN to improve pose prediction by leveraging the time dimension. They evaluate their work on a synthetic dataset as well as a real-world dataset. Borghi et al. [3] propose to utilize stream of depth images and perform 3-DoF pose estimation of the head, especially for in-car automotive applications. The principle idea is to utilize time information on the depth-based images and directly estimating the Euler angles. Head detection and localization are not performed in this work, and they assume that the head center is already provided. They utilize this to first crop the head part from the input images and only use it to train the proposed network. They mostly use a shallow architecture with L_2 training loss. The method achieved good results on Biwi dataset [10] and was implemented in a car which shows its real-time feasibility. Similarly, Peng et al. in [27] used spatial-temporal data and RNNs for sequence-based facial tracking without facial landmarks.

Camera pose tracking is an intrinsic part of visual odometry, where several works have focused on CNN and LSTM combinations for pose tracking [42, 36, 8]. Wang et al. [36] and Constante [8] deploy CNNs with consecutive LSTMs to extract continuous poses based on RGB video input. Zou et al. [42] present a self-supervised learning method for visual odometry with special consideration for consistency over longer sequences. They model the long-term dependency by using a pose network that features a two-layer convolutional LSTM module. All methods use CNNs developed for Optical Flow estimation as their backbone. One typical backbone is FlowNet [9]. The extracted features are then fed into the LSTMs for further pose regression.

In contrast to the state-of-the-art, we propose several novel Neural Network concepts combined with LSTMs for AR glasses pose tracking. For the first time, we introduce separable convolutions, non-local blocks, or a combination of both to achieve improved tracking accuracy. Both concepts are promising for our multi-view, continuous data. Especially non-local blocks have not been used for 6-DoF pose tracking, despite its interesting property of learning the long-range object dependencies in video sequences. Based on this comparison of various novel concepts on AR glasses pose tracking, we can make recommendations for our use case.

3 AR Glasses Pose Estimation & Tracking Architectures

3.1 AR Glasses Dataset and Preprocessing

To conduct our analysis, we use the HMDPose dataset [12]. HMDPose is a large-scale data glasses dataset, consisting of around 3 million 1280×752 pixel im-

ages. It contains IR images from three different perspectives of four different AR glasses, worn by 14 different subjects.

We first downscale the images 320×188 and normalize them. Next, the images from different views from the same timestep are stacked together channel-wise. This increases the information space for the neural network to learn from. We present different neural network architectures based on CNNs and RNNs trained on IR images to perform AR glasses pose estimation. We create sequences from these multi-view images using the individual frames' timestamp as required by RNN-based networks. The data split acquired while sequence generation is used for all the networks, CNNs or RNNs. Figure 2 gives an overview of the pipeline of pre-processing such sequences. We train and evaluate various networks based

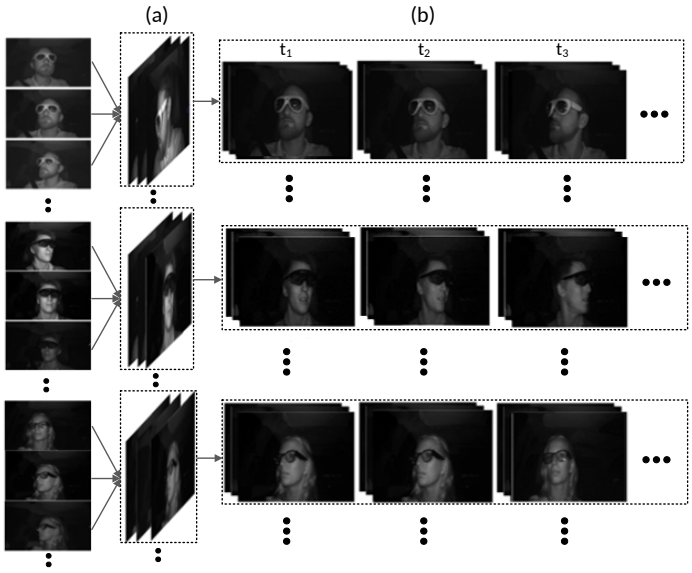

Fig. 2. Our pipeline to conduct sequence generation from multi-view IR images for LSTM-based methods. (a) We first stack IR images channel-wise and then (b) generate sequences of the stacked images according to their timestamp.

on single frames and sequences to regress the 6-DoF pose of AR glasses.

3.2 CNN Baseline Methods

We benchmark two networks that only utilize CNNs to see the effect of learning by using spatial information only. The networks are called "BaselineCNN" and "GlassPoseRN". GlassPoseRN is a deeper model containing a ResNet-18 backbone compared to BaselineCNN. The two networks act as our baseline to compare with our networks based on RNNs.

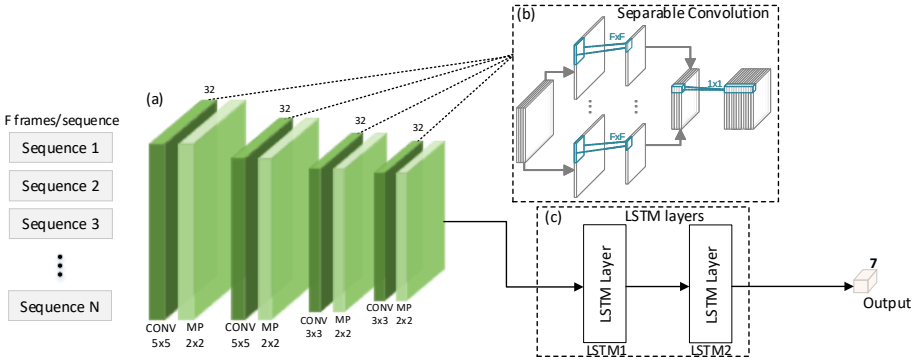

Fig. 3. Network architecture of separable convolution-based CNN-LSTM model. (a) shows our CNN architecture, including the separable convolutions highlighted in (b), which elaborates on their working principle [7]. The light green layers right after the separable convolutions are max pooling layers. In (c), the output of the network is fed into LSTM layers, performing estimation of the orientation in quaternions and position in Euclidean space.

BaselineCNN Our first approach is a basic convolutional neural network, which we consider as our baseline. We name this method "BaselineCNN". It is inspired by the "GlassPose" method in [11]. Instead of decoupling the network into subnetworks for translation and rotation, we take one branch for 6-DoF pose estimation. BaselineCNN does neither use any extra cropped images of glasses nor the bounding box of glasses in the translation subnetwork. It utilizes full downscaled images to regress the 6-DoF pose per frame triple. The network learns the rotations in quaternion space and translation in Euclidean space.

The network comprises nine layers in total, with six convolutional and three fully connected layers. The input to the network is of size 320×188 pixels. Initially, the images are processed by three convolutional layers of filter size 5×5, each having 32 neurons followed by a max pooling layer of 2×2. The remaining three convolutional layers have a filter size of 3×3. The fourth and fifth convolution layers have a hidden size of 32, while the sixth convolution layer has 128 neurons. Only the fourth layer has a consecutive max pooling layer. In the end, the fully connected layers of sizes 128, 80, and 7 neurons are used to regress the 6-DoF pose.

GlassPoseRN We benchmark the state-of-the-art pose estimation "GlassPoseRN" [11] approach of the HMDPose dataset. We compare our methods to the GlassPoseRN method, as it performs better than the benchmarked Regression via Classification head pose estimator [1] and a point cloud based object pose estimator called CloudPose [15] on the HMDPose dataset [13, 14]. It also shows better results than the point cloud-based P2P method [14]. We retrain this network on our 80/10/10 data split. We give more details on our data split in Subsection 3.5. The model consists of a ResNet-18 [17] backbone. GlassPoseRN

maintains information and feature within the building blocks of ResNet with skip connection as described in [17]. The ResNet-18 block of the network is followed by two dense layers of size 256 and 64 with a final output layer which regresses the translation and orientation together.

3.3 LSTM-based approaches

As an alternative to the models mentioned above, we introduce RNN-based networks to compare the convolution and recurrent operations. The idea is to exploit the temporal information in the data by introducing RNNs in the network. Our custom-built hybrid CNN-RNN model utilizes this enhanced spatio-temporal feature space to learn the glass pose in an improved way. CNNs can learn the spatial information avoiding feature engineering, while RNNs avoid manual engineering of object tracking logic, which can be prone to errors. They learn the temporal information directly from the data. Hence, a hybrid model could achieve enhanced AR glasses pose estimation.

We use LSTMs as our RNN variant due to their ability to avoid vanishing gradient problem and the poses time-dependent property. The LSTMs map the pose estimation per frame in a sequence to the sequence of known ground truth poses.

CNN-LSTM The first hybrid model is referred to as "CNN-LSTM". The CNN part of the network extracts optimal features from the IR images, and LSTM tracks the extracted features as sequences. Therefore, the features from a single triple image in a sequence act as one time step for LSTM. The designed model takes a sequence of IR images as input and outputs continuous 6-DoF pose of AR glasses, tracking the pose in all the frames of a sequence.

The architecture consists of only six layers in total. The CNN part is kept identical to BaselineCNN to enable a fair comparison. The CNN consists of four layers where each is followed by max pooling layer of 2×2. Two convolutional layers have a filter size of 5×5, and the remaining two layers have a filter size of 3×3. The hidden size of each layer is 32 neurons. The features from the CNN are flattened before using it as an input to the LSTMs. The LSTM network comprises two layers, each having 128 neurons. The dense layers are used as the output layers to regress the continuous 6-DoF pose of AR glasses.

SepConv-LSTM This method introduces a more efficient network by utilizing depthwise separable convolutions instead of normal convolutions. Depthwise separable convolutions [7] are a special type of convolutions in which convolution operations are separately applied to channels. The information from multiple channels in the input is not mixed together, which are the different views of the triple images in our case. Pointwise convolutions increase the depth of the output to be further processed. Separable convolutions increase efficiency by decreasing the number of computations. They are also used in standard architectures like MobileNet [18].

Figure 3 shows an overview of the architecture, referred as "SepConv-LSTM". The network architecture remains the same as CNN-LSTM. Only normal 2D convolution layers are replaced with separable convolutions. Table 1 highlights the difference in parameters, showing the reduction of approximately 40,000 parameters using separable convolutions.

Table 1. Size and parameter comparison of CNN-LSTM and SepConv-LSTM.

Network	Total Parameters
CNN-LSTM	3,849,575
SepConv-LSTM	3,808,274

GlassPoseRN-LSTM The next LSTM-based model is the GlassPoseRN method coupled with LSTMs. This model is designed for spatio-temporal data. The LSTM at the end of the network learns from the temporal information and is used in a regression manner to predict continuous 6-DoF AR glasses pose. We call this model "GlassPoseRN-LSTM". The model is built over the ResNet-18

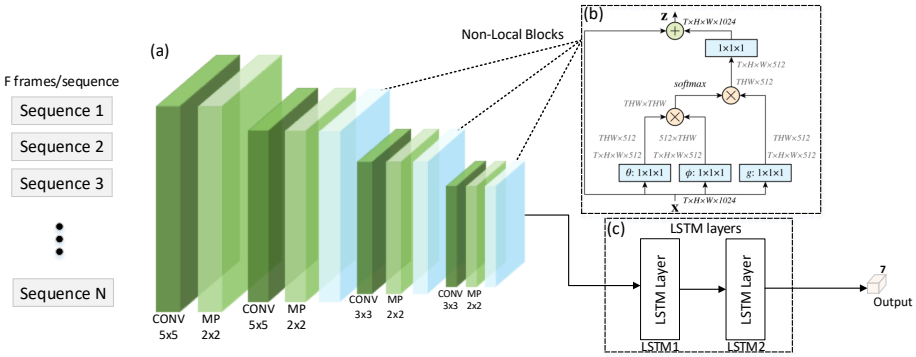

Fig. 4. Network architecture of non-local block-based CNN-LSTM model. (a) shows our CNN architecture, including the non-local layers highlighted in (b), which elaborates on their working principle [37]. In this block, $\otimes$ denotes matrix multiplication and $\oplus$ represents element wise addition. The light blue box within non-local block represents the 1x1x1 convolutions. In (c), the output of the network is fed into LSTM layers, performing estimation of the orientation in quaternions and position in Euclidean space.

architecture as in GlassPoseRN. The reason for using the same backbone is to compare the effect of RNNs and CNN-based methods.

The GlassPoseRN model is modified by removing the three dense layers from the end and adding the LSTM layers. It is done by first adding a global average

pooling layer right after the ResNet blocks to reshape the feature maps. This pooling layer is followed by the two LSTM layers with 128 neurons each. In the end, a dense layer is used to regress the translation and rotations, making it a layer of 7 neurons.

3.4 RNN with Non-Local Blocks

In previous sections, convolutional and recurrent operations were used to build networks due to our data modality's nature. This subsection describes a novel neural network architecture based on non-local operations. Non-local operations help in learning long-range dependencies in the video sequence. Wang et al.[37] introduced non-local filtering as general operations in a neural network as a building block to learn the non-local neighbourhood information and long-range dependencies. Before introducing this variant, learning the long-range dependencies was done by either repeatedly deep stacking convolutional operations with large receptive fields or deep stacking of recurrent operations consecutively. This multi-hop dependency with deep stacked networks, however, is hard to model and difficult to optimize making it an inefficient and computationally expensive solution.

NL-CNN-LSTM We alter our hybrid CNN-LSTM model by adapting non-local blocks in the CNN part of the network. The convolution and recurrent layers only consider the local neighbourhood of a given pixel position while learning features from the input and the fully connected layers lose the positional correspondence. Hence, the long-range dependencies between the sequenced IR frames and even the non-local neighbor pixels within one frame are missed. Therefore, non-local blocks add the capability to learn this non-local information within a single frame and between multiple frames of a sequence to the network.

The network is designed by modifying the CNN-LSTM architecture. The change includes the addition of non-local blocks in a conventional convolutional network part to experiment with their performance in an LSTM-based network. The architecture is visualized in Figure 4. The light blue blocks in the figure represent the location of where the non-local blocks are inserted. Due to limited computation resources and a large image resolution size, the non-local block is not inserted after the first convolutional layer, despite it potential benefit to the overall network. The architecture is kept similar to the CNN-LSTM architecture to keep a fair comparison between the two models. This network consists of six layers with four convolutional layers, three non-local blocks, and two LSTM layers. The convolutional layers are identical to the CNN part of the CNN-LSTM neural network. The non-local blocks do not affect the size of the input feature maps. The block performs multiple 1×1 convolutions to receive different embeddings. The embeddings are later used in element-wise multiplication and get the final output feature vector, representing the similarity between a pixel and its non-local neighbours. This approach is an extension of our hybrid CNN-LSTM model, with the potential to learn the long-range dependencies of different objects for the AR glasses pose estimation task.

NL-SepConv-LSTM Following the idea of depth-wise separation of convolutions just like SepConv-LSTM, we designed a similar model with the additional inclusion of non-local blocks. As non-local blocks are introduced to learn non-local neighbor features [37], by their inclusion in our network, we aim to learn the non-local neighbor features from separate views.

The architecture of the network remains the same as NL-CNN-LSTM, which we described in the previous subsection. The only change is the replacement of normal convolution layers by separable convolutions. We call this network "NL-SepConv-LSTM".

Fig. 5. Example triple images of the four AR glasses models included in the HMD-Pose [12] dataset. It includes the four glasses models a) Mini Augmented Vision, b) Everysight Raptor, c) Microsoft Hololens 1, and d) North Focal Generation 1.

3.5 Network Training

Our training, validation, and test split is 80% for training and 10% for validation and test set. In contrast to the 94/3/3 split used in [13], more extensive test and validation sets are required, as sequencing data comes with the loss of frames. For this reason, we train the GlassPoseRN method again on our data split to enable comparability with our LSTM-based models. We split the dataset for all individual glasses types as well as for individual subjects. For the CNN baseline methods, the dataset is being shuffled.

We train our models with an Adam optimizer with the initial learning rate $\alpha = 0.01$. The learning rate is scheduled to decrease if the validation loss has not improved for more than ten epochs. For our RNN-based neural networks, sequences are generated by stacking the frames, sorted according to the timestamp. Our activation function for all layers besides the output layer is the ReLU activation. We deploy a linear activation for the output layers.

We use the weighted L_2 Euclidean distance for translation and orientation. The loss function is based on Kedall et al. [20] and is defined as follows:

$$Loss = \beta||t - \tilde{t}||_2 + ||q - \frac{\tilde{q}}{||\tilde{q}||}||_2 + \gamma||q|| \tag{1}$$

q and t describe the ground truth quaternion and translation, whereas $\tilde{q}$ and $\tilde{t}$ represent the estimated quaternion and translation. We normalize the predicted

quaternion and compute the Euclidean distance to the ground truth quaternion. It is important to note that regularization parameter γ and the norm of the predicted quaternion are added in the loss to cater to any large predictions. In addition, the Euclidean distance is being computed for the translation. The translation is weighted accordingly through the scaling factor β to have similar scaling to the orientation before adding the orientation loss, which we set empirically. We train all networks on two NVIDIA GeForce 2080Ti GPUs for 200 epochs for all glasses combined and individually. All the networks are trained on full images down-scaled to 320×188 pixels. The initial learning rate is set to 0.001 and scheduled to 50% decrease after every 10 epochs if the validation loss has not improved from previous best loss. The batch size used for CNN baseline methods is 128, while, for RNN-based methods, a sequence length of 8 and batch size of 32 was utilized while training. The trained networks can predict an absolute 6-DoF pose per frame, up to a pose per sequence of 8 frames.

4 Evaluation

4.1 Dataset and Evaluation Metrics

We conduct the training and evaluation of our approaches on the HMDPose dataset [12]. The large-scale multi-view IR dataset HMDPose contains around 3 million images with AR glasses pose annotations, resulting in 1 million image triples. The dataset has been recorded with 14 different subjects, wearing four different AR glasses models each. It includes the four glasses models Everysight Raptor, Microsoft Hololens 1, North Focal Generation 1, and Mini Augmented Vision (Figure 5). In our paper's evaluation, we refer to the Everysight Raptor as EVS, Hololens 1 as HOLO, North Focal Generation 1 as NORTH, the Mini Augmented Vision glasses as MAV, and all glasses combined as ALL for readability. There are around 250,000 image triples per glasses model available.

We benchmark our results on the same metrics as Firintepe et al. [13], with one exception. We consider the Mean Absolute Error (MAE) and the Root Mean Squared Error (RMSE) for orientation. An additional metric is the Balanced Mean Angular Error (BMAE), which takes the unbalanced amount of the full range of head orientations by introducing sections [30, 31, 11] into account. Despite its interesting insight into a networks' performance over the complete range, we exclude this metric, as the section definition is invalid for sequenced data. This would implicate the definition of ranges per sequence, where the sample size is too small to represent the test set. We use the L_2 loss for the position error on all axes separately and together for the position estimation.

4.2 Results

Frame-by-Frame Pose Estimation Approaches We benchmark two frame-by-frame pose estimation methods for comparison. The first network is our GlassPose-inspired BaselineCNN, the second network being the state-of-the-art

Table 2. Rotation results of the Frame-by-Frame Pose Estimators BaselineCNN and GlassPoseRN[13] as well as our LSTM-based Pose Trackers CNN-LSTM, SepConv-LSTM, NL-CNN-LSTM, NL-SepConv-LSTM, and GlassPoseRN-LSTM on the given error metrics in degrees. The Everysight Raptor is referenced as EVS, Hololens 1 as HOLO, North Focal Generation 1 as NORTH and the Mini Augmented Vision glasses as MAV. ALL stands for all glasses combined. The average value of all three axes is given on the defined metrics in this table. The full results including the errors regarding roll, pitch, and yaw can be seen in the end of the document in Table 4.

		Frame-by-Frame Pose Estimation		LSTM-based Pose Tracking				
Glasses-Type	Metric \ Method	Baseline -CNN	GlassPose -RN[13]	CNN -LSTM	SepConv -LSTM	NL-CNN -LSTM	NL-SepConv -LSTM	GlassPoseRN -LSTM
EVS	MAE	2.62	1.79	**0.74**	0.76	0.76	1.27	1.34
	RMSE	3.95	2.71	**0.77**	0.79	0.80	1.34	1.43
MAV	MAE	2.71	2.25	0.81	0.79	0.82	**0.78**	1.22
	RMSE	3.94	3.26	0.84	**0.82**	0.85	**0.82**	1.37
HOLO	MAE	2.75	3.31	0.94	**0.86**	0.97	2.57	1.67
	RMSE	4.63	4.82	0.97	**0.90**	1.01	2.68	1.77
NORTH	MAE	2.47	1.40	**0.89**	0.91	0.89	2.33	1.37
	RMSE	4.29	2.98	**0.94**	0.97	**0.94**	2.44	1.47
ALL	MAE	2.98	3.86	0.88	**0.81**	0.94	0.92	1.90
	RMSE	4.33	5.26	0.92	**0.86**	1.00	0.98	1.98

Table 3. Results for the positional, Euclidean error in millimeters of the Frame-by-Frame Pose Estimators BaselineCNN and GlassPoseRN[13] as well as our LSTM-based Pose Trackers CNN-LSTM, SepConv-LSTM, NL-CNN-LSTM, NL-SepConv-LSTM, and GlassPoseRN-LSTM on all three axes combined. The Everysight Raptor is referenced as EVS, Hololens 1 as HOLO, North Focal Generation 1 as NORTH and the Mini Augmented Vision glasses as MAV. ALL stands for all glasses combined. The full results including the errors regarding the x-, y-, and z-axes can be seen in the end of the document in Table 5.

	Frame-by-Frame Pose Estimation		LSTM-based Pose Tracking				
Glasses-Type	Baseline -CNN	GlassPose -RN[13]	CNN -LSTM	SepConv -LSTM	NL-CNN -LSTM	NL-SepConv -LSTM	GlassPoseRN -LSTM
EVS	12.06	8.41	4.18	**3.96**	4.55	7.18	8.89
MAV	15.07	5.97	4.23	**3.22**	3.59	3.56	5.65
HOLO	13.10	8.75	5.31	**4.57**	5.93	11.67	9.39
NORTH	13.49	4.88	5.18	**5.09**	5.19	12.79	11.83
ALL	14.82	11.95	5.22	**4.46**	5.44	5.43	15.86

"GlassPoseRN" approach. They are the only networks formed on convolutional operations only, so they cannot work on sequential data. Table 2 and 4 lists all orientation results of all methods on our data split. We trained and tested both approaches on all individual glasses as well as combined. BaselineCNN achieves comparable results among all glasses models as well as the glasses combined. The average MAE ranges from 2.47° on NORTH to 2.98° on ALL, showing slightly higher errors on the combination of the glasses compared to them individually. On the RMSE, the largest glasses model HOLO and the smallest glasses NORTH result in the highest errors. GlassPoseRN shows more significant differences among the individual glasses and combined. It can be observed that the smaller glasses types like NORTH and EVS perform better than the larger models. Considering all glasses at once increases the error.

We additionally evaluate the positional error of the methods on our data split (Table 3 and 5). We trained and tested both approaches on all individual glasses as well as combined. BaselineCNN again performs similarly for all glasses types. The GlassPoseRN errors differ more between the glasses models. When comparing the two pose estimation approaches, we observe GlassPoseRN to achieve mostly better results on the metrics for orientation and translation.

LSTM-based Pose Tracking Methods Based on the two pose estimation methods, we introduced and benchmarked five AR glasses tracking approaches.

Table 2 and 4 show the orientation results. Our extension of the GlassPoseRN method with LSTM performs better than GlassPoseRN. GlassPoseRN-LSTM estimates the orientation with less than 1.98° error. Regarding the individual glasses, we observe a similar pattern as for GlassPoseRN. ALL has a lower performance than the glasses individually. On NL-SepConv-LSTM, the error is high for NORTH and HOLO, being the smallest and the largest glasses models. The average error of all other glasses models and their combination is below 1.50°. The performance of CNN-LSTM, SepConv-LSTM, and NL-CNN-LSTM is comparable when differences on individual glasses are considered. The averages of all three methods for MAE on the various glasses models range from 0.74° to 0.94°. On the RMSE, the average values range from 0.77° to 1.01°. Regarding all five methods, SepConv-LSTM performs consistently best overall. The method achieves the best results on ALL and HOLO on both metrics.

The position benchmark results in a comparable pattern (Table 3 and 5). In the case of GlassPoseRN-LSTM, the errors are the highest, with an L_2 error of 15.86mm on ALL and values between 5.65mm to 11.83mm on the individual glasses. NL-SepConv-LSTM has higher errors on HOLO and NORTH, as already seen on the orientation error. NL-CNN-LSTM, Sep-Conv-LSTM, and CNN-LSTM attain errors in a similar range for the individual glasses and combined. Overall, SepConv-LSTM performs best among all methods. This is the case for all objects. Although close error ranges between the three aforementioned approaches are observable, SepCon-LSTM performs consistently best. Similar to our orientation comparison, NL-SepConv-LSTM and GlassPoseRN-LSTM achieve higher errors than the rest.

AR Glasses Pose Estimation & Tracking Comparison One of our fundamental goals in this work is comparing pose estimation and pose tracking methods for AR glasses. For this purpose, we benchmarked two frame-by-frame pose estimation and five pose tracking approaches. The results of the approaches show significant improvements in favor of pose tracking methods.

Pose Accuracy Regarding pose accuracy, the best frame-by-frame method GlassPoseRN achieves orientation errors on the MAE between 1.40° and 3.86°, which is reduced to an error of 0.76° to 0.91° with the SepConv-LSTM method. Thus, we observe a reduction of estimation errors of up to 76%. A similar tendency is visible on the position estimation. Furthermore, the estimation differences between various glasses models decrease significantly with the introduction of LSTMs, especially with separable convolutions without non-local blocks.

Inference Time Comparing their performance regarding inference time is possible by measuring the estimation time between feeding images into a network and the time a network makes a prediction. Our measurements were made on a single NVIDIA GeForce 2080Ti. The Baseline-CNN achieves 100FPS, compared to 42FPS of GlassPoseRN. The pose tracking methods all achieve similar inference times due to their similarity in depth. Thus, each method achieves 74FPS. All methods fulfill the real-time requirement. Only the GlassPoseRN architecture estimates poses slower than 60 fps, which is the acquisition speed of the used HMDPose dataset. Subsequently, both the pose tracking and frame-by-frame pose estimation can perform similarly in real-world conditions.

Discussion We chose our methods with a mostly identical CNN-backbone to ensure a network as similar as possible. This had the aim to deduce if the LSTM, the additional non-local blocks, or separable convolution deployment contributes to the changes in the accuracy. Between the various LSTM-based methods, the combination of non-local blocks and separable convolutions decreased estimation accuracy compared to approaches with the individual components separately. An explanation for this might be the concurring properties of separable convolutions and non-local blocks. Separable convolutions separately handle various channels of the images by applying the convolutions individually, which are the individual IR images in our case. Non-local blocks aim to learn non-local dependencies in the images. The layers containing non-local blocks might enjoy the information of three images mixed, which collides with the separable convolution concept. Individually applied, they output similar errors.

Finally, the introduction of LSTMs to the ResNet-18-based GlassPoseRN brings improvement but underperforms compared to the methods based on a simple CNN. This underlines once more the enhancement of the pose estimation accuracy with the introduction of LSTMs. The advanced feature extraction properties of ResNet seem redundant in the case of IR image-based AR glasses pose estimation when LSTMs are deployed. In addition, our combined methods work more efficiently compared to the GlassPoseRN approach. We achieve

around 74FPS inference time for the BaselineCNN-LSTM combinations, whereas the inference time for GlassPoseRN is 42 FPS.

The LSTM-based methods were trained with image sequence lengths of 8. The images of the HMDPose dataset were recorded with 60FPS, thus, the networks would potentially profit from longer sequence lengths. However, a longer sequence implicates dropping more images in the process of training, validation, and test split generation. This would endanger a proper training of the Deep Neural Networks, especially for individual AR glasses models. A networks with an even lower sequence lengths than 8 frames would most likely not register temporal information, as the movement is hardly visible.

Regarding deploying the algorithms in a real world setting where AR content is shown to a driver of a vehicle through AR glasses, we suggest the efficient CNN with separable convolutions named SepConv-LSTM to track the pose. They save computation time and, compared to CNN-LSTM and NL-CNN-LSTM, achieves better results at the same time. Thus, our LSTM-based methods improve the pose tracking of AR glasses. As a highly accurate pose is required when AR glasses are deployed into the car, our methods can improve the stability in which the AR content is shown while driving, leading to an improved AR experience.

5 Conclusion

In this paper, we analyzed outside-in approaches on pose tracking and pose estimation of AR glasses. We first introduced and benchmarked two frame-by-frame pose estimation approaches. One method is the state-of-the-art GlassPoseRN model, developed for AR glasses pose estimation. Based on the baseline variants, we extended them with LSTMs to achieve AR glasses pose tracking. We presented methods with and without non-local blocks and further added separable convolutions in some networks for comparison. Non-local blocks consider non-local neighbor features in one image and among multiple images, while separable convolutions focus on maintaining the individual channels, and therefore the triple images. We observe a significant boost on the HMDPose dataset from pose estimation to tracking approaches. Separable convolutions improve our recurrent networks' results, where our SepConv-LSTM algorithm shows the best performance with an estimation error of 0.81° in orientation and 4.46mm in position. In contrast to GlassPoseRN, we decrease the estimation error by 76%. The results are promising to improve the in-car AR experience in the case of AR glasses deployment, as a high 6-DoF pose estimation accuracy positively affects the superimposition of the real world with virtual elements, which we further improve in this work.

On the one hand, future work will consist of potential fusion and Neural Network ensemble methods to evaluate combinations of frame-by-frame pose estimation and tracking approaches. On the other hand, the variance of the performance across individuals will be analyzed.

Method Type		Frame-by-Frame Pose Estimation								LSTM-based Pose Tracking															
Glasses-Type	Metric	BaselineCNN				GlassPoseRN[13]				CNN-LSTM				SepConv-LSTM				NL-CNN-LSTM				NL-SepConv-LSTM				GlassPoseRN-LSTM			
		Roll	Pitch	Yaw	Avg	Roll	Pitch	Yaw	Avg	Roll	Pitch	Yaw	Avg	Roll	Pitch	Yaw	Avg	Roll	Pitch	Yaw	Avg	Roll	Pitch	Yaw	Avg	Roll	Pitch	Yaw	Avg
EVS	MAE	2.44	2.55	2.86	2.62	1.65	1.89	1.83	1.79	0.65	**0.63**	**0.94**	**0.74**	0.64	**0.63**	1.01	0.76	**0.62**	0.66	1.01	0.76	0.99	1.01	1.81	1.27	0.99	1.74	1.31	1.34
	RMSE	3.54	3.43	4.89	3.95	2.56	2.40	3.16	2.71	0.67	**0.65**	**1.00**	**0.77**	0.66	**0.65**	1.07	0.79	**0.65**	0.68	1.06	0.80	1.04	1.05	1.93	1.34	1.06	1.79	1.43	1.43
MAV	MAE	2.52	2.90	2.73	2.71	2.56	2.33	1.87	2.25	0.84	0.83	0.77	0.81	0.86	0.73	0.76	0.79	**0.82**	0.88	0.76	0.82	0.88	**0.72**	**0.74**	**0.78**	1.30	1.10	1.25	1.22
	RMSE	3.53	3.93	4.35	3.94	3.52	3.30	2.97	3.26	0.86	0.85	0.82	0.84	0.89	**0.75**	0.81	**0.82**	**0.84**	0.90	0.81	0.85	0.91	**0.75**	**0.80**	**0.82**	1.42	1.22	1.47	1.37
HOLO	MAE	2.38	2.56	3.32	2.75	2.70	4.78	2.44	3.31	0.85	0.76	1.21	0.94	**0.84**	**0.74**	**1.00**	**0.86**	0.96	0.80	1.16	0.97	2.68	1.62	3.41	2.57	1.4	2.14	1.47	1.67
	RMSE	3.18	3.61	7.09	4.63	3.65	6.07	4.74	4.82	**0.87**	0.78	1.26	0.97	**0.87**	**0.76**	**1.06**	**0.90**	0.98	0.83	1.23	1.01	2.70	1.81	3.53	2.68	1.47	2.20	1.65	1.77
NORTH	MAE	2.43	2.47	2.51	2.47	1.48	1.25	1.48	1.40	**0.76**	**0.67**	1.22	**0.89**	0.90	0.75	**1.09**	0.91	0.84	0.74	1.10	0.89	1.63	2.19	3.15	2.33	0.91	1.67	1.52	1.37
	RMSE	3.54	3.83	5.50	4.29	2.68	2.06	4.19	2.98	**0.79**	**0.72**	1.30	**0.94**	0.93	0.80	1.18	0.97	0.86	0.80	**1.17**	**0.94**	1.69	2.28	3.35	2.44	0.97	1.77	1.67	1.47
ALL	MAE	2.82	2.88	3.24	2.98	2.95	6.01	2.61	3.86	0.81	0.84	**0.98**	0.88	**0.74**	**0.72**	**0.98**	**0.81**	0.91	0.87	1.06	0.94	0.84	0.83	1.11	0.92	1.30	3.13	1.25	1.90
	RMSE	3.94	4.06	4.98	4.33	4.08	7.21	4.48	5.26	0.83	0.87	**1.05**	0.92	**0.77**	**0.77**	**1.05**	**0.86**	0.95	0.91	1.14	1.00	0.87	0.88	1.18	0.98	1.36	3.19	1.39	1.98

Table 4. Rotation results of the Frame-by-Frame Pose Estimators BaselineCNN and GlassPoseRN[13] as well as our LSTM-based Pose Trackers CNN-LSTM, SepConv-LSTM, NL-CNN-LSTM, NL-SepConv-LSTM, and GlassPoseRN-LSTM on the given error metrics in degrees. The Everysight Raptor is referenced as EVS, Hololens 1 as HOLO, North Focal Generation 1 as NORTH and the Mini Augmented Vision glasses as MAV. ALL stands for all glasses combined. The roll, pitch, yaw and the average of all three axes are given on the defined metrics.

Method Type	Frame-by-Frame Pose Estimation								LSTM-based Pose Tracking																			
Glasses-Type	BaselineCNN				GlassPoseRN[13]				CNN-LSTM				SepConv-LSTM				NL-CNN-LSTM				NL-SepConv-LSTM				GlassPoseRN-LSTM			
	x	y	z	L_2	x	y	z	L_2	x	y	z	L_2	x	y	z	L_2	x	y	z	L_2	x	y	z	L_2	x	y	z	L_2
EVS	6.31	7.04	4.60	12.06	5.06	3.87	3.70	8.41	2.86	**1.74**	1.44	4.18	**2.84**	1.83	**1.06**	**3.96**	2.90	2.19	1.64	4.55	4.73	3.70	2.11	7.18	6.43	2.74	3.52	8.89
MAV	6.53	10.15	5.40	15.07	3.34	2.50	2.64	5.97	2.85	1.66	1.62	4.23	**2.19**	**1.27**	1.21	**3.22**	2.29	1.59	1.37	3.59	2.55	1.40	**1.18**	3.56	4.11	2.02	1.99	5.65
HOLO	7.33	7.73	4.20	13.10	5.74	3.78	3.22	8.75	3.38	2.51	1.89	5.31	**3.15**	**2.02**	**1.42**	**4.57**	3.52	2.94	2.36	5.93	6.80	3.20	5.65	11.67	5.39	3.41	4.79	9.39
NORTH	7.28	7.66	5.14	13.49	3.26	2.40	**1.63**	4.88	3.13	2.70	1.73	5.18	**3.02**	**2.52**	1.70	**5.09**	3.24	2.70	1.77	5.19	7.45	6.50	5.27	12.79	6.38	3.89	6.55	11.83
ALL	7.23	9.53	5.05	14.82	6.47	5.36	6.04	11.95	3.01	2.70	1.97	5.22	**2.67**	**2.43**	**1.53**	**4.46**	3.48	2.67	1.85	5.44	3.40	2.69	1.91	5.43	8.55	7.67	7.52	15.86

Table 5. Results for the positional, Euclidean error in millimeters of the Frame-by-Frame Pose Estimators BaselineCNN and GlassPoseRN[13] as well as our LSTM-based Pose Trackers CNN-LSTM, SepConv-LSTM, NL-CNN-LSTM, NL-SepConv-LSTM, and GlassPoseRN-LSTM on the individual axes and combined. The Everysight Raptor is referenced as EVS, Hololens 1 as HOLO, North Focal Generation 1 as NORTH and the Mini Augmented Vision glasses as MAV. ALL stands for all glasses combined.

References

1. Berg, A., Oskarsson, M., O'Connor, M.: Deep Ordinal Regression with Label Diversity. In: 2020 25th International Conference on Pattern Recognition (ICPR). pp. 2740–2747 (2021)
2. Borghi, G., Fabbri, M., Vezzani, R., Calderara, S., Cucchiara, R.: Face-from-Depth for Head Pose Estimation on Depth Images. IEEE Transactions on Pattern Analysis and Machine Intelligence **42**(3), 596–609 (2018)
3. Borghi, G., Gasparini, R., Vezzani, R., Cucchiara, R.: Embedded recurrent network for head pose estimation in car. In: 2017 IEEE Intelligent Vehicles Symposium (IV). pp. 1503–1508. IEEE (2017)
4. Borghi, G., Venturelli, M., Vezzani, R., Cucchiara, R.: Poseidon: Face-from-depth for driver pose estimation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017)
5. Capellen, C., Schwarz, M., Behnke, S.: ConvPoseCNN: Dense Convolutional 6D Object Pose Estimation pp. 162–172 (2020)
6. Chen, B., Parra, A., Cao, J., Li, N., Chin, T.J.: End-to-end learnable geometric vision by backpropagating PnP optimization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8100–8109 (2020)
7. Chollet, F.: Xception: Deep Learning with Depthwise Separable Convolutions. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1800–1807 (2017)
8. Costante, G., Mancini, M.: Uncertainty Estimation for Data-Driven Visual Odometry. IEEE Transactions on Robotics **36**(6), 1738–1757 (2020)
9. Dosovitskiy, A., Fischer, P., Ilg, E., Häusser, P., Hazirbas, C., Golkov, V., Smagt, P.v.d., Cremers, D., Brox, T.: FlowNet: Learning Optical Flow with Convolutional Networks. In: 2015 IEEE International Conference on Computer Vision (ICCV). pp. 2758–2766 (2015)
10. Fanelli, G., Dantone, M., Gall, J., Fossati, A., Gool, L.: Random Forests for Real Time 3D Face Analysis. Int. J. Comput. Vision **101**(3), 437–458 (Feb 2013)
11. Firintepe, A., Mohamed, S., Pagani, A., Stricker, D.: The More, the Merrier? A Study on In-Car IR-based Head Pose Estimation. In: 2020 IEEE Intelligent Vehicles Symposium (IV). IEEE (2020)
12. Firintepe, A., Pagani, A., Stricker, D.: HMDPose: A large-scale trinocular IR Augmented Reality Glasses Pose Dataset. In: 26th ACM Symposium on Virtual Reality Software and Technology. ACM (2020)
13. Firintepe, A., Pagani, A., Stricker, D.: A Comparison of Single and Multi-View IR image-based AR Glasses Pose Estimation Approaches. In: 2021 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW). pp. 571–572 (2021)
14. Firintepe, A., Vey, C., Asteriadis, S., Pagani, A., Stricker, D.: From IR Images to Point Clouds to Pose: Point Cloud-Based AR Glasses Pose Estimation. Journal of Imaging **7**(5) (2021), https://www.mdpi.com/2313-433X/7/5/80
15. Gao, G., Lauri, M., Wang, Y., Hu, X., Zhang, J., Frintrop, S.: 6D Object Pose Regression via Supervised Learning on Point Clouds. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). pp. 3643–3649 (2020)
16. Gu, J., Yang, X., De Mello, S., Kautz, J.: Dynamic Facial Analysis: From Bayesian Filtering to Recurrent Neural Network. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1531–1540 (2017)

17. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778 (June 2016)
18. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
19. Kendall, A., Grimes, M., Cipolla, R.: Posenet: A convolutional network for real-time 6-dof camera relocalization. In: Proceedings of the IEEE international conference on computer vision. pp. 2938–2946 (2015)
20. Kendall, A., Grimes, M., Cipolla, R.: PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. pp. 2938–2946 (12 2015)
21. Li, Y., Wang, G., Ji, X., Xiang, Y., Fox, D.: Deepim: Deep iterative matching for 6d pose estimation. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 683–698 (2018)
22. Li, Z., Wang, G., Ji, X.: CDPN: Coordinates-Based Disentangled Pose Network for Real-Time RGB-Based 6-DoF Object Pose Estimation. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 7677–7686 (2019)
23. Murphy-Chutorian, E., Trivedi, M.M.: Head Pose Estimation and Augmented Reality Tracking: An Integrated System and Evaluation for Monitoring Driver Awareness. IEEE Transactions on Intelligent Transportation Systems **11**(2), 300–311 (2010)
24. Ning, G., Zhang, Z., Huang, C., Ren, X., Wang, H., Cai, C., He, Z.: Spatially supervised recurrent convolutional neural networks for visual object tracking pp. 1–4 (2017)
25. Park, K., Patten, T., Vincze, M.: Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7668–7677 (2019)
26. Peng, S., Liu, Y., Huang, Q., Zhou, X., Bao, H.: PVNet: Pixel-Wise Voting Network for 6DoF Pose Estimation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
27. Peng, X., Feris, R.S., Wang, X., Metaxas, D.N.: A recurrent encoder-decoder network for sequential face alignment. In: European conference on computer vision. pp. 38–56. Springer (2016)
28. Rad, M., Lepetit, V.: BB8: A Scalable, Accurate, Robust to Partial Occlusion Method for Predicting the 3D Poses of Challenging Objects without Using Depth. In: 2017 IEEE International Conference on Computer Vision (ICCV). pp. 3848–3856 (Oct 2017)
29. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You Only Look Once: Unified, Real-Time Object Detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2016)
30. Schwarz, A., Haurilet, M., Martinez, M., Stiefelhagen, R.: DriveAHead-a large-scale driver head pose dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 1–10 (July 2017)
31. Selim, M., Firintepe, A., Pagani, A., Stricker, D.: AutoPOSE: Large-Scale Automotive Driver Head Pose and Gaze Dataset with Deep Head Pose Baseline. In: International Conference on Computer Vision Theory and Applications (VISAPP). SCITEPRESS Digital Library (2020)
32. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition (2015)

33. Song, C., Song, J., Huang, Q.: HybridPose: 6D Object Pose Estimation Under Hybrid Representations. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 428–437 (2020)
34. Tekin, B., Sinha, S.N., Fua, P.: Real-Time Seamless Single Shot 6D Object Pose Prediction. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 292–301 (June 2018)
35. Tremblay, J., To, T., Sundaralingam, B., Xiang, Y., Fox, D., Birchfield, S.: Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects. In: Proceedings of The 2nd Conference on Robot Learning. Proceedings of Machine Learning Research, vol. 87, pp. 306–316. PMLR (29–31 Oct 2018)
36. Wang, S., Clark, R., Wen, H., Trigoni, N.: DeepVO: Towards end-to-end visual odometry with deep Recurrent Convolutional Neural Networks. In: 2017 IEEE International Conference on Robotics and Automation (ICRA). pp. 2043–2050 (2017)
37. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local Neural Networks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7794–7803 (2018)
38. Xiang, Y., Schmidt, T., Narayanan, V., Fox, D.: PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. In: Kress-Gazit, H., Srinivasa, S.S., Howard, T., Atanasov, N. (eds.) Robotics: Science and Systems XIV, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, June 26-30, 2018 (2018)
39. Xu, Z., Chen, K., Jia, K.: W-posenet: Dense correspondence regularized pixel pair pose regression. arXiv preprint arXiv:1912.11888 (2019)
40. Zakharov, S., Shugurov, I., Ilic, S.: DPOD: 6D Pose Object Detector and Refiner. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 1941–1950 (2019)
41. Zhang, Y., Ming, Y., Zhang, R.: Object detection and tracking based on recurrent neural networks. In: 2018 14th IEEE International Conference on Signal Processing (ICSP). pp. 338–343. IEEE (2018)
42. Zou, Y., Ji, P., Tran, Q.H., Huang, J.B., Chandraker, M.: Learning Monocular Visual Odometry via Self-Supervised Long-Term Modeling. In: Computer Vision – ECCV 2020. pp. 710–727. Springer International Publishing, Cham (2020)