GrabAR: Occlusion-aware Grabbing Virtual Objects in AR

XIAOWEI HU, The Chinese University of Hong Kong
XIAO TANG, The Chinese University of Hong Kong
CHI-WING FU, The Chinese University of Hong Kong
DANIEL COHEN-OR, Tel Aviv University

Existing augmented reality (AR) applications often ignore occlusion between real hands and virtual objects when incorporating virtual objects in our views. The challenges come from the lack of accurate depth and mismatch between real and virtual depth. This paper presents GrabAR, a new approach that directly predicts the real-and-virtual occlusion, and bypasses the depth acquisition and inference. Our goal is to enhance AR applications with interactions between hand (real) and grabbable objects (virtual). With paired images of hand and object as inputs, we formulate a neural network that learns to generate the occlusion mask. To train the network, we compile a synthetic dataset to pre-train it and a real dataset to fine-tune it, thus reducing the burden of manual labels and addressing the domain difference. Then, we embed the trained network in a prototyping AR system that supports hand grabbing of various virtual objects, demonstrate the system performance, both quantitatively and qualitatively, and showcase interaction scenarios, in which we can use bare hand to grab virtual objects and directly manipulate them.

CCS Concepts: Computing methodologies → Mixed / augmented reality; Neural networks;

1 INTRODUCTION

Augmented Reality (AR) puts virtual contents over real ones to enrich our views in the physical world (Caudell and Mizell 1992; Krueger et al. 1985). This idea recently gains much more popularity, thanks to the ever-improving computing and display capabilities of smartphones and see-through glasses. However, in most AR applications today, virtual objects are simply drawn as an overlay above all real objects, without interactions. It remains challenging to seamlessly incorporate virtual objects into the real world, such that the real and virtual objects are perceived to naturally co-exist in the views, and we can further directly use our hands to interact with the virtual objects.

To achieve the goal, one main challenge is handling the occlusion between the real and virtual objects, since they may fully or partially occlude one another. Figure 1 shows a simple example: (a) user’s hand in the AR view and (b) a virtual can. If we simply draw the can over the hand (c), the result is not realistic, since parts of the fingers should go above the can when the hand grabs the can. Occlusion, also known as occultation and interposition, is an important visual cue (Anderson 2003; Ono et al. 1988), allowing us to rank the relative proximity among objects in our views. It comes naturally in the real world, but is typically ignored in existing AR applications.

One common approach to address the occlusion problem is to employ a depth sensor to acquire the depth information for the real hand, then determine the occlusion per-pixel by comparing the acquired depth of the hand and the rendered depth for the virtual objects (Battisti et al. 2018; Feng et al. 2018). Figure 1(d) shows a typical result. Certainly, this depth-based approach alleviates the occlusion problem. However, since depth acquisition is often noisy and inaccurate, the hand-object boundary is usually erroneous with observable flickering during the interactions. Also, user’s hand may easily penetrate into the virtual object (see arrows in Figure 1(d)), since the real depth from the hand may not precisely match the virtual depth from renderings. Particularly, without haptic feedback, it is hard for one to avoid penetration. Furthermore, this approach requires an additional depth sensor attached to the RGB camera.

In this paper, we show that by learning to handle natural grabbing postures, we are able to obtain accurate occlusion that increases the sense of immersion for hand interactions with virtual objects.

Our main innovation is GrabAR, a new approach to resolve occlusion between hand (real) and grabbable objects (virtual) by learning to determine the occlusion from hand grabbing poses. By using GrabAR, one can use his/her hand in physical space to grab virtual objects in the view. The hand can be naturally composed with the virtual object, as if it is directly grabbing a real object.

Technically, GrabAR is a purely image-based approach that determines occlusion entirely in the image space, without acquiring or inferring 3D depth information. To design and train a neural network to determine the occlusion, the main challenges come from the time performance requirement and dataset generation. To this end, we adopt a compact neural network and compile a large synthetic dataset (using 3D renderings to automatically generate occlusion masks) to pre-train it and a small real dataset (with manually-prepared occlusion masks) to fine-tune it. With these two datasets, we can leverage a large amount of labeled data for training, while addressing the domain difference for the synthetic data. Further, we put the trained network to work in a prototyping AR system, and demonstrate real hand grabbing various virtual objects in AR. We perform a series of experiments to evaluate the system, both quantitatively and qualitatively, and showcase scenarios, in which we can directly use bare hand to grab and manipulate virtual objects.

2 RELATED WORK

Hand-based AR interactions. Direct hand-object interaction in AR has been widely studied in the past decades. Data gloves were introduced (Dorfmuller-Ulhaas and Schmalstieg 2001; Marquardt et al. 2011; Wang and Popovic 2009) to support the interactions in pioneering AR applications, e.g., Benko et al. (Benko and Feiner 2007). However, the device is expensive, tedious to set up, fragile, and moreover, hinders the user actions. Later, many works explore hand tracking with depth sensors for hand-object interactions in AR. Radkowski et al. (Radkowski and Stritzke 2012) use Kinect sensor to enable users to move virtual objects with fist/open hand gestures.
Moser et al. (Moser et al. 2016) implement intuitive interactions using LeapMotion (LeapMotion 2019) in an optical see-through AR system. However, the depth sensor is used only to track the hand, without resolving the hand-object occlusion.

Very recently, some works begin to explore the hand-object occlusion problem in AR. Feng et al. (Feng et al. 2018) use LeapMotion sensor and combine a tracking-based method and a model-based method to estimate the occlusion mask. Though some results are shown for few objects, their approach requires a cumbersome set up to first customize user’s hand shape on paper then track and estimate the hand geometry using a depth sensor for producing the occlusion mask. Battisti et al. (Battisti et al. 2018) leverage stereo images from LeapMotion to produce the depth map of hands, then to compare it with the rendered depth map for virtual objects to determine the hand-object occlusion. As mentioned in Section 1, this approach, however, suffers from various issues due to the depth inaccuracy and mismatch between the acquired depth and rendered depth.

Vision-based hand interaction methods only require a monocular camera to track user’s hands. So, they are more generic for use with, e.g., smartphones. However, since monocular hand-pose or depth estimation are still challenging problems, most existing works focus only on gesture recognition rather than detailed geometry recovery for interactions. Chun et al. (Chun and Höllerer 2013) develop a simple hand recognition method to support different types of interactions in AR. Choi et al. (Choi et al. 2013) adopt a vision-based method to estimate the occlusion map of a hand for simple virtual object overlay. Song et al. (Song et al. 2015) use an RGB camera to recognize various hand shapes and estimate an average distance between hand and camera. The distance value is then utilized for assorted interaction tasks, including scaling and selecting. However, the result is still not sufficiently precise to resolve the hand-object occlusion.

**Hand pose/shape estimation.** Hand pose/shape estimation aims to employ a coarse 3D hand mesh to estimate an approximate 3D hand shape/pose from the RGBD images (Mueller et al. 2017; Rogez et al. 2015; Ye and Kim 2018; Yuan et al. 2018), single RGB images (Baek et al. 2019; Boukhayma et al. 2019; Cai et al. 2018; Ge et al. 2019; Mueller et al. 2018; Panteleris et al. 2018; Zimmermann and Brox 2017), or single depth images (Malik et al. 2018). Among them, Ge et al. (Ge et al. 2019) and Boukhayma et al. (Boukhayma et al. 2019) predict not only the hand pose but also the 3D hand model. However, their main focus is on the hand shape or pose rather than on the precise hand-and-finger depth information. Hence, if we take the results of hand pose/shape estimation (i.e., hand mesh) to determine the occlusion relationship, we still have the hand-object penetration problem, which is nontrivial on its own. Therefore, we design a new approach to bypass the depth estimation and avoid the hand-object penetration altogether by directly predicting the occlusion relationship in the image space.

**Monocular occlusion estimation.** Occlusion estimation (Hoiem et al. 2007; Ren et al. 2006; Teo et al. 2015; Wang and Yuille 2016) is a long-standing and challenging research problem. Given an image, it aims to estimate the boundary between objects in the image, and further identify the occlusion relations between objects at each boundary segment. An early work by Ren et al. (Ren et al. 2006) use an edge detector to find edge features then apply Markov random field to distinguish between the foreground and background. More recently, Teo et al. (Teo et al. 2015) extract multiple local features and geometric grouping cues in input images to detect boundaries and border ownership, while detecting the background and foreground. Wang et al. (Wang and Yuille 2016) build a neural network to detect object boundaries and estimate the occlusion relationships. Wang et al. (Wang et al. 2018) predicted the object occlusion boundaries by adopting multi-level and multi-scale features in a deep convolutional neural network. So far, works in this area focus on foreground (figure) and background (ground) detection. We are not aware of any method that directly analyzes and reasons the occlusion relations between real and virtual objects.

**Single-camera depth estimation.** Depth estimation and occlusion estimation are inter-related problems (Wang and Yuille 2016). On the one hand, if we can obtain the depth information in an image, we can infer the occlusion. On the other hand, if we can determine the occlusion boundaries, we can take it to improve the depth estimation, as shown in various works in the literature (Hoiem et al. 2007; Ren et al. 2006).

Nowadays, single-camera depth estimations are usually powered by SLAM (Engel et al. 2014; Mur-Artal et al. 2015), which takes a video stream from a single camera as input and predicts the depth map by solving an optimization problem. For example, Valentin et
we are motivated to design a new approach that directly predicts depth. Eigen et al. explore for learning features to predict depth. Recently, Fu et al. try to grab and manipulate the virtual object. Taking a depth estimation approach to the problem necessarily requires estimating depth maps with sufficient accuracy in the physical space to grab the virtual object in the AR view and manipulate it, to certain extent. Figure 2 presents an overview of our approach. The input is a pair of hand (real) and object (virtual) images. To compose them with partial occlusion, we first preprocess the two images (b) and feed the results into our neural network (c). The network then generates an occlusion mask, which marks the hand portions over the virtual object. Using this mask, we can then correctly compose the hand and virtual object in the view. Note the partial occlusion achieved in the result (d).

Another stream of work explores depth estimation for single image in monocular views, which benefits scenarios with dynamic interactions. Early works learn to predict depth in images using hand-crafted features, e.g., Markov random field (Saxena et al. 2006, 2008), light flow (Furukawa et al. 2017), and perspective geometry (Ladicky et al. 2014). Recently, convolutional neural networks (CNN) are explored for learning features to predict depth. Eigen et al. (Eigen et al. 2014) employ a CNN to predict a coarse depth map then apply another CNN to refine the result. Laina et al. (Laina et al. 2016) develop a deep residual network and leveraged the reverse Huber loss to predict depth. Recently, Fu et al. (Fu et al. 2018) formulate network learning as an ordinal regression problem and develop a spacing-increasing discretization strategy to discretize depth, whereas Nicodemou et al. (Nicodemou et al. 2018) adopt the hourglass network (Newell et al. 2016) to infer depth of human hand.

In this work, we also target at handling occlusion between real and virtual objects. Taking a depth estimation approach to the problem necessarily requires estimating depth maps with sufficient accuracy at interactive rates, for which existing methods cannot achieve. Thus, we are motivated to design a new approach that directly predicts the occlusion between real hand and virtual grabbable object, and bypasses the need to predict or acquire depth.

3 OVERVIEW

In existing AR systems, users see through an apparatus to look at the real world augmented with virtual objects. So, when we stretch our hands into the AR view to reach the virtual object, the object would simply appear as an overlay over our hands in the view. Hence, existing systems cannot support us to directly use our bare hands to grab and manipulate the virtual object.

GrabAR is a new approach to the problem, enabling not only a natural composition of real hand (camera view) and virtual object (rendering) with partial occlusion, but also direct use of bare hands in the physical space to grab the virtual object in the AR view and manipulate it, to certain extent. Figure 2 presents an overview of our approach. The input is a pair of hand (real) and object (virtual) images (a). To compose them with partial occlusion, we first preprocess the two images (b) and feed the results into our neural network (c). The network then generates an occlusion mask, which marks the hand portions over the virtual object. Using this mask, we can then correctly compose the hand and virtual object in the view. Note the partial occlusion achieved in the result (d).

The main challenges to train the network come from the dataset and the time performance requirement. First, the real hand and virtual object images must be compatible in view space, so the hand pose in training data should appear to grab the virtual object in the view. Second, we need ground-truth occlusion masks to supervise the network training. However, to obtain accurate ground truths is typically very tedious, due to the manual labeling work; and yet, training a network often requires large amount of data. Lastly, the network inference should be fast to support interactive applications, so its architecture cannot be overly complex.

To meet these challenges, we first employ 3D renderings to produce a large synthetic data, in which the labels (occlusion masks) are automatically generated. Also, we compile a rather small set of real image data with manual labels to address the domain difference between 3D renderings and real hand images, so that we can further fine-tune the network (see Section 4). This approach leverages a large amount of training data, and significantly reduces the burden of manual labeling. Next, rather than directly taking the hand and virtual object images as network inputs, we preprocess them by overlaying the virtual object on hand and coloring it in blue (see Figures 2 (b)). With such coloring, the virtual object looks more distinctive from the hand, allowing the network to better learn from the inputs. From these inputs, we then formulate a compact neural network and train it to locate hand portions above the virtual object (see Section 5.1). Equipped with the network to generate occlusion...
masks, we further create a prototyping system and develop various AR interaction scenarios (see Section 5.2).

4 DATASETS

This section presents how we prepare the dataset. Our dataset contains image tuples of (i) the hand; (ii) virtual grabbable object; and (iii) associated occlusion mask, which marks the visible portions of hand and object, as ground truth to supervise the network training. To prepare the data with compatible images in view space, a naive approach is manual: given an image of the virtual object (Figure 1(a)), we capture a corresponding hand image (Figure 1(b)), then manually label the occluding regions in one of the images. Clearly, such manual process is very time-consuming. Also, we cannot guarantee that the hand pose matches the virtual object in the same view. Another approach is to use an RGBD sensor to acquire depth and produce the occlusion mask. However, the acquired depth is noisy and may not match the rendered depth, so we still need to manually edit the occlusion mask. Particularly, since a large amount of data is typically needed to train a neural network, such tedious approaches are practically infeasible.

We address these challenges by designing a two-stage procedure to compile the data. First, we use 3D renderings to prepare a synthetic dataset to pre-train our neural network. The advantage is that we can efficiently produce a large amount of data with synthesized occlusion masks without manual labeling. In the second stage, we address the domain gap problem that arises from the non-realistic rendering. Hence, we build a simple interactive AR system to overlay the virtual objects on captured hand images. By then, we can prepare a small dataset with manual labels to generalize the network for real inputs. Below, we elaborate the two stages.

Synthetic dataset. The procedure to prepare synthetic data is illustrated by the running example shown in Figure 3. First, we use LeapMotion (LeapMotion 2019) to capture the 6-DoF skeletal joints of hand in physical space (a) and deform a rigged 3D hand model to fit these skeletal joints (b). Then, we render the virtual object around the 3D hand model and display them together (c). By adjusting the hand pose to appear to grab the virtual object, we can obtain a plausible hand-object occlusion in the virtual space (d). Hence, we can locate the foreground pixels of the hand and virtual object (e), and automatically generate the corresponding occlusion mask (f). By rendering the results in different camera views and repeating the process for different target objects, we can efficiently produce a large volume of synthetic data with occlusion masks.

Real dataset. Next, we need to prepare real data to address the domain gap problem; see Figure 4 for a running example. To do so, we use a webcam to capture the hand (a) and a display to continuously show it in virtual space (b). Then, we render the target virtual object semi-transparently over the hand (c) to allow interactively adjusting the hand pose until it appears to grab the object (d). Hence, we can obtain a real image of the hand and a compatible rendered image of the virtual object coherently in the same view, and label the visible portion of hand to produce the occlusion mask (e). Further, to efficiently capture more data in different poses and views, we attach

Fig. 3. Synthetic data preparation. (a) Real hand in front of LeapMotion. (b) Rigged 3D hand corresponding to real hand. (c) Target virtual object added next to virtual hand. (d) Hand pose that appears to grab the virtual object. (e) Visible portions of the virtual object in the image view. (f) The resulting occlusion mask.

Fig. 4. Real data preparation. (a) Captured hand in real physical space. (b) Rendered hand in virtual space. (c) Semi-transparent virtual object overlaid on hand. (d) Adjusted hand pose to grab the virtual object. (e) Manually-labeled occlusion mask (on hand). (f) After we grab-and-rotate the virtual object with our real hand.
Fig. 5. The virtual objects employed in preparing the datasets.

an ArUco marker (Garrido-Jurado et al. 2014) below the hand to track its orientation and position in the camera view. Once the hand appears to grab the virtual object, we use the tracked information to move the virtual object with the hand. Therefore, we can capture more images of hand and object in different views (f).

We collected ten virtual objects of various types to compile the two datasets. Figure 5 shows the 3D models and some typical grabbing poses. Note the different grabbing gestures for different objects, and we can have multiple (common) grabbing poses for the same object, e.g., see ping-pong paddle and cola in Figure 5 (left). Altogether, we collected 24,539 image tuples in the synthetic dataset and 957 image tuples in the real dataset. We will release the two datasets upon the publication of this work.

5 METHOD
This section first presents the neural network model and its implementation details, then the prototyping AR system and the interaction scenarios developed based on the system.

5.1 Neural Network Model
Figure 6 shows the overall deep neural network architecture for predicting the hand-object occlusion. Altogether, the network has ten blocks to extract convolutional feature maps from the inputs; see supplemental material for the detailed architecture diagram. Each block includes a convolutional operation (Conv), a group normalization (GN), and a leaky ReLU operation. Moreover, we adopt skip connections to aggregate the detail information encoded in the low-level feature maps at shallow layers to the high-level feature maps with semantic information at deep layers. At the end, we leverage another convolutional operation and a softmax function to generate the network output, i.e., occlusion mask.

Network training strategies. We initialize the network parameters by random noise, which follows a zero-mean Gaussian distribution with a standard deviation of 0.1. Then, we leverage two stages to train the network. In the first stage, we train the network by using the synthetic dataset and optimize it by a stochastic gradient descent optimizer with a momentum value of 0.9, and a weight decay of $5 \times 10^{-4}$. We empirically set the learning rate as $5 \times 10^{-4}$ and terminate the learning process after 100,000 iterations.

In the second stage, we keep the weights learned from the first stage as the initial network parameters. Then, we fix the parameters in the first four blocks and train on the real data to fine-tune the other parts of the network. Here, we adopt the Adam optimizer to refine the network with the first momentum value of 0.9 and the second momentum value of 0.99. The learning rate is initialized as $5 \times 10^{-4}$ and dropped 90% in every 1,600 iterations. The whole training process terminates after 20,000 iterations.

In both stages, we empirically set the batch size as eight, and employ horizontal flip, random rotation, random sharpness, random brightness, and random contrast for data argumentation strategies. Also, we train the network by minimizing the cross-entropy loss between the network output and the ground-truth occlusion mask (see Figure 6) from the prepared data.

Post-processing and image composition. The raw network predictions often contain holes and jagged boundaries. Hence, we use a median filter and a morphologically close filter to smooth edges and fill holes in the output occlusion mask.
Discussion. To meet the interactive processing requirement, the network model has to be kept simple. In the course of this work, we follow the design principle of "fewer channels with more layers" (Kim et al. 2016) to build the network, which consumes less processing time and produces higher accuracy. On the other hand, in the early development of this work, we trained a network model individually for each virtual object in the dataset. The results were, however, not satisfactory. Interestingly, we later trained the network using the whole dataset of all virtual objects altogether. The results improve, since more training data and object categorizes enhance the generalization capability of the deep network.

5.2 Prototyping AR System and Applications

Figure 7 shows the prototyping AR system with (i) a webcam that live captures the hand in front of the camera; (ii) an ArUco marker attached below the hand for tracking; and (iii) a screen that shows the composed AR view. In this work, we focus on a natural composition of hand (real) and grabbable objects (virtual) for enhancing AR applications. Hence, in our current system implementation, we simply put a green curtain behind the hand to allow the system to cut out the hand region, and use the ArUco marker to track the hand pose in front of the camera. We will explore other methods to perform these tasks to improve the system in the future.

In detail, we render the virtual object, as if it is at a distance of 30cm from the camera. When the user’s hand enters the camera view, our system will start to use the neural network to determine the occlusion mask and take it to compose the hand and virtual object in the view. Then, when the user feels that the object has been grabbed, he can use the other hand (the non-dominant one) to press a button to tell the system that the object is grabbed. We leave it as a future work to automatically tell the moment at which the real hand grabs the virtual object in the AR view. After the grab, the pose of the virtual object will be locked with the hand, so we can use bare hand to directly move it and further perform various interesting interaction scenarios, as presented below:

Scenario #1: Virtual but clickable lightsaber. The first scenario features the virtual lightsaber shown in Figure 8. It is a virtual 3D object in the AR view, but yet, has a clickable button. After one grabs the object with bare hand and presses the (red) button on it, the light blade can show up over the lightsaber.

Scenario #2: Virtual but rotatable knob. The second scenario features a virtual knob that is grabbable and rotatable. Figure 9 shows the interaction procedure. After one grabs the virtual knob by hand, he/she can rotate it to the left or to the right. Also, one can release the knob, grab again, and make a larger turn.

Scenario #3: Virtual but scrollable phone. In the third scenario, we present a virtual phone in the AR view; see Figure 10. Unlike virtual objects in existing AR applications, the phone is grabbable...
and also manipulatable. First, when one grabs the phone in the AR view, our system can present the hand (real) and phone object (virtual) with natural partial occlusions (a). Further, when one uses his/her thumb to scroll up (b) and down (c), the virtual phone can respond to the action and acts like a real one on our hand. When see-through AR glasses become ready in the future, this interaction scenario suggests that users may manipulate their phones in the AR views, without physically bringing out the real phones.

Altogether, the three scenarios show that empowered by our prototyping system, we can design AR objects that are not only grabbable, but also clickable, scrollable, and rotatable. With dynamic interactions, we can greatly improve the user experience with AR objects, beyond what the existing AR applications can offer.

6 EVALUATIONS

This section presents a series of experiments to evaluate GrabAR, where we show its time performance, compare it with results produced by other approaches, both qualitatively and quantitatively, and present a user study to explore the user experience.

6.1 Time Performance

GrabAR is built on a desktop computer with a 3.7GHz CPU and a single NVidia Titan Xp GPU. Also, it makes use of the Intel RealSense D435 camera to capture RGB images of the real world. The system runs at 8.8 frames per second for a 512 × 512 image, including the image capture. Table 1 shows the detailed time performance for each algorithm stage. Note that we did not fully optimize the neural network and did not use a faster GPU. We will improve the time performance of GrabAR in our future work.

| Stages                                      | Time   |
|---------------------------------------------|--------|
| (a) Inputs (capturing hands & rendering)    | 30ms   |
| (b) Preprocessing                           | 7ms    |
| (c) Neural network                          | 70ms   |
| (d) Image composition                       | 6ms    |
| Total                                       | 113ms  |

6.2 Visual Comparison with Different Approaches

Most existing methods (Battisti et al. 2018; Feng et al. 2018) adopted depth sensors to estimate the occlusion mask between the real hand and virtual objects. Here, we use the Intel RealSense D435 depth camera to set up an AR system, where we render the virtual objects over the real RGB background, then calibrate the color camera with the virtual camera, and use the depth information of the virtual objects and real hand to determine their occlusion relationship. In detail, the depth information of the virtual objects is obtained directly from the depth buffer after rendering the object, while the depth information of the real hand is obtained solely from the depth sensor.

Figure 11 shows some of the visual comparison results, where the “naïve overlay” approach (see column (b)) clearly destroys the sense of immersion, and fails to give the perception that the real hand and virtual objects co-exist in the same space. The “depth sensor” approach (see column (c)) introduces artifacts on the boundaries between virtual objects and real hand. Note that for better comparison, we also apply the same post-processing step mentioned in Section 5.2 to the results produced by the “depth sensor” approach (see column (d)), since the raw results contain a large amount of noise. In contrast, our method is able to generate more realistic composition of the hand and virtual objects (see column (e)).

Further, we compare our methods with others on unseen virtual objects, meaning that these objects are included in neither our synthetic nor real datasets for training the neural network. Figure 12 shows some of the results, where our method also produces plausible occlusions for these virtual objects. Importantly, our system is cheap and flexible to set up, comparing with other systems that rely on depth sensors. Please look at our supplemental video for animated comparisons and results.

6.3 User Study

We conducted a user study to explore how GrabAR-Net (denoted as “GN”) improves the user experience in AR. In a pilot study, we found that the participants always prefer the depth-sensor-based approach with post-processing (denoted as “DSP”) more than the vanilla depth-sensor-based method (denoted as “DP”). Hence, we only compare GN with DSP in the user study.

Altogether, we recruited 10 participants in the campus: four females and six males, aged between 22 and 28. Among them, eight had experience with AR, and all are right-handed.

Tasks. The user study contains two tasks:

(i) Grabbing Task. Each participant was asked to grab six virtual objects of different kinds and poses (paddle with handshake grip, paddle with penhold grip, can, pistol, flashlight, and lightsaber), where pistol and lightsaber were unseen. The virtual objects are rendered using GN or DSP in each trial. Also, the participant will make two trials for each permutation. So, we have $10 \times 6 \times 2 = 120$ trials in total. Note that the order with the modes was counterbalanced across the participants using a balanced Latin square design.

In each trial, the participant starts it by pressing a button on the keyboard to initiate the trial. After that, the system shows the next virtual object on the screen by rendering it using GN or DSP. The

Fig. 10. Scenario #3: Virtual but scrollable phone. Again, after one grabs the phone object (a), one can use the thumb to scroll up (b) or scroll down (c) to look at different portion of Scandinavian on the map, even though the phone is just a virtual object in AR.

participant can then stretch hand to the virtual object and make a hand pose to appear to grab the object. Immediately after the participant feels that the object has been successfully grabbed in the AR view, he/she can use the other hand to press a button and tell the system to record the time taken by the participant in the trial.

(ii) Interactive Task. The second task uses the same set up as the first task, but we focus on letting the participants try and experience hand interactions with AR objects, with GN, and with DSP. Here, we use the scrollable virtual phone demo presented in Section 5.2, since it is a familiar object in the real world. Supported by our system, the participant can grab the virtual phone, scroll over it as if the phone is real, and look at different parts of the image on the phone screen. Further, the participant can find a button and click on it, and we impose no time limit in this task and require them to interact with GN and with DSP using the same amount of time. During the task, we count the number of times that the participant’s hand penetrates through the virtual objects with GN and with DSP.

Procedure. When each participant came to our lab., we first introduced the AR system to him/her and started a tutorial session for the participant to try to grab a virtual object with na¨ıve overlay, with GN, and with DSP. Particularly, when we showed the na¨ıve overlay to the participants, we asked them to point out the occlusion issue to make sure that they understand the occlusion relationship between real hand and virtual objects. After that, we performed the two tasks as presented above on each participant.

After completing the tasks, each participant was asked to fill a questionnaire. The questionnaire consists of three Likert-scale questions (1 means disagree and 5 means agree) on GN and on DSP: (Q1) the presented occlusion was handled correctly (“Correct Occlusion” for short); (Q2) the image quality is good, i.e., noise-free without flickering (“Image Quality” for short); and (Q3) the hand-object interaction is plausible and immersive (“Interaction Experience” for short). Lastly, we further interviewed the participant and obtained comments from him/her.

Quantitative comparison results. We summarize the quantitative comparison results (GN vs. DSP) in terms of time performance (Task 1) and penetration count (Task 2) in Figures 13 (a) and (b).

(i) Time to grab object. Let $M$ and $SD$ denote mean and standard deviation, respectively. The time taken (in sec.) by the participants to grab the virtual objects under GN is $(M = 4.135$ with $SD = 0.894$), which is significantly shorter than that of DSP $(M = 6.820$ with $SD = 1.314$), as shown also by the t-test: $t(9)=13$ with $p < 0.0001$. This result implies that the participants needed less time to adjust their hand poses to grab the virtual objects under GN.

(ii) Penetration count. The penetration count under GN is $(M = 0.28$ with $SD = 0.270$), which is significantly lower than that of DSP $(M = 2.12$ with $SD = 0.953$), where the t-test result is $t(9)=6.241$ with $p = 0.0002$. Again, the result shows that GN can better avoid object-hand penetration in the AR views, thus introducing less interference during the hand-object interactions.

Participant ratings. Next, we summarize the ratings on the three questions Q1-Q3 in Figures 13 (c), (d), and (e), respectively.

(Q1) Correct Occlusion. The ratings on Q1 under GN is $(M = 4.3$, $SD = 0.483$), which is significantly higher than that under DSP $(M = 1.9$, $SD = 0.567$) with $Z=-2.848$, $p=0.004$. The result implies that the participants observed better occlusion results under GN.

(Q2) Image Quality. The ratings on Q2 under GN is $(M = 4.6$, $SD = 0.516$), which is significantly higher than that under DSP $(M = 1.7$, $SD = 0.8233$) with $Z=-2.836$, $p=0.005$. The result implies that GN produced better image quality as observed by the participants.

(Q3) Interaction experience. The ratings on Q3 under GN is $(M = 3.9$, $SD = 0.568$), which is again significantly higher than that under
Participant comments. During Task 1, all participants agreed that NO (Naïve Overlay) is not ideal for supporting virtual objects grabbing in AR, and showed their preferences to GN, because GN provides sharp and accurate results at edges, while DSP results are noisy and flickering. As for Task 2, we observed that the participants often needed more time to adjust their hand poses to grab the virtual phone under DSP. P2 gave a representative comment: “I feel hard to sense the distance between the camera and the virtual object, so I often reached out my hand to the front or to the back of the virtual object instead of directly grabbing it between my fingers under DSP. But this situation appeared less under GN.” When scrolling on the virtual phone, seven participants pointed out that object-hand penetration happened too often under DSP, due to unstable hand motions. It is “caused by the unintentional scrolling operations on the virtual phone (P1 & P3),” and “interfered me and made me feel unrealistic (P6).” Object-hand penetration also happens under GN, due to failure predictions from GrabAR-Net, but “it occurred at an acceptable frequency (P9).”

However, GN is not perfect all the time. Two participants reported that the interaction progress was not smooth due to a noticeable latency, which is a limitation of our existing prototyping system. Three participants mentioned that although the system is occlusion-aware, they still felt unrealistic due to the lack of tactile feedback. It is a common drawback for AR applications. However, P5 said that “I preferred this comparing to carrying an extra physical device for tactile feedback.”

Table 2. Quantitative comparison in terms of ODSC. Unseen virtual objects are marked with “*”. Note that DS denotes depth-sensor-based approach, DSP denotes depth-sensor-based approach with post-processing, and GN denotes GrabAR-Net.

| Object    | DS   | DSP  | GN (our) |
|-----------|------|------|----------|
| can       | 63.35% | 59.64% | 84.53%   |
| cup       | 85.75% | 63.24% | 86.57%   |
| flashlight| 76.09% | 72.72% | 94.16%   |
| gun       | 80.78% | 79.57% | 94.43%   |
| loupe     | 91.13% | 80.61% | 95.84%   |
| paddle    | 84.78% | 80.68% | 92.08%   |
| phone     | 85.16% | 84.48% | 93.11%   |
| knob*     | 76.89% | 75.72% | 90.37%   |
| lightsaber* | 92.84% | 83.23% | 95.27%   |
| pistol*   | 93.91% | 89.51% | 95.49%   |
| average   | 81.57% | 75.81% | 91.54%   |

6.4 Quantitative Comparison on Seen and Unseen Objects

Evaluation metric. To quantitatively evaluate how well GrabAR handles the occlusion between real and virtual objects, we design a new evaluation metric, namely ODSC, to specifically look at the overlap region between hand and virtual object and ignore other image regions. To do so, we first extract the overlap region (Ω) between real and virtual object, then compute the dice score coefficient (DSC) between the predicted occlusion mask (P) and ground truth image (G) only within Ω. In detail, we define our ODSC as

\[
ODSC = \frac{2|P \cap G|}{|P| + |G|},
\]
where subscript $\Omega$ counts relevant pixels in $\Omega$, $\Omega = H \cap O$, $H$ denotes the hand region, and $O$ denotes the virtual object regions.

**Evaluation results.** To support quantitative evaluation, we first recruit ten volunteers to help us annotate 180 pairs of real hand and virtual object images and produce ground-truth occlusion masks. This evaluation dataset covers 10 different virtual objects. Among the ten virtual objects, two of them have two different grabbing poses and three of them are unseen.

Then, we can quantitatively compare the results produced by GrabAR-Net (GN) with results produced from the depth-sensor-based approach (DS) and depth-sensor-based approach with post-processing (DSP). Table 2 reports the ODSC values for different virtual objects, showing that GrabAR-Net produces far better results than other approaches clearly by a large margin. Further, we can see that GrabAR-Net achieves good performance not only on the seen objects (where the object category is included in the dataset for training the deep network) but also on the unseen objects (where the object category is not included in the training dataset). This result demonstrates the generalization capability of our GrabAR-Net. Note that we train GrabAR-Net by using all the virtual objects in training data together. We believe our network is able to learn the gestures of the real hand rather than simply remembering the virtual objects.

6.5 Evaluate the Network Input Strategy

Our method uses a pre-processing step (see Figure 2(b)) to color the virtual object in blue and overlay it on hand as the network input. By doing so, the virtual object can become more distinctive from hand, allowing the network to better learn from the inputs.

To evaluate the effectiveness of this strategy, we performed an ablation experiment by directly concatenating the virtual object image (without coloring it in blue) and the real hand image as the inputs to GrabAR-Net. Here, the average ODSC achieved by the trained network is only 73.42%, compared to 91.54% reported earlier in Table 2. This result shows that by coloring the virtual object in blue, the textures on the virtual objects are ignored, so we can make easy the network learning process, improve the network’s robustness, and enable it to even work on unseen objects.

7 CONCLUSIONS

This paper presents GrabAR, a new approach that makes use of an embedded neural network to directly predict real-and-virtual occlusion in AR views, while bypassing inaccurate depth estimation and inference. Using our approach, we can enable users to grab and manipulate various virtual objects with accurate occlusion, and increase the sense of immersion for hand interactions with virtual objects. Our key idea is to learn to determine the occlusion from hand grabbing poses that describe natural virtual object grabs with the user’s hand in physical space. Particularly, we compile synthetic and real datasets and use them to train the neural network to leverage the automatically-generated labels in synthetic data and reduce the burden of manually labeling the real data. A prototyping AR system is presented with various interaction scenarios, showing that virtual objects in AR can be grabbable, as well as interactable. We evaluate GrabAR and compared it with depth-sensor-based methods through various experiments, and show that GrabAR provides significantly more plausible results than methods relying on depth-sensors, both visually and quantitatively.

As a first work, we are aware of, that explores the deep neural network to compose virtual objects with real hands, there are still several problems that we plan to work on in the future. (i) We only tested our system on the background with a single color. To improve the system’s robustness, more training data with various scenes and backgrounds is required. (ii) Currently, our system supports interaction using a single hand only. We plan to extend it, allowing bimanual interactions by including more training samples. (iii) Our system relies on an ArUco marker to detect the six-DoF poses of the user’s hand. In the future, we will explore the use of only a single RGB image by training a deep neural network to learn the six-DoF poses automatically. Moreover, we consider further speeding up the processing time by taking the temporal information of video into the network to produce smoother results.

REFERENCES

Barton L Anderson. 2003. The role of occlusion in the perception of depth, lightness, and opacity. Psychological Review 110, 4 (2003), 785.

Seungryul Baek, Kwang In Kim, and Tae-Kyun Kim. 2019. Pushing the Envelope for RGB-based Dense 3D Hand Pose Estimation via Neural Rendering. In IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). 1067–1076.
Caterina Battisti, Stefano Messeledoli, and Fabio Poiesz. 2018. Seamless Bare-Hand Interaction in Mixed Reality. The IEEE Int. Symposium on Mixed and Augmented Reality, 198–203.

Hrvoje Benko and Steven Feiner. 2007. Balloon selection: A multi-finger technique for accurate low-fatigue 3d selection. In 2007 IEEE Symposium on 3D User Interfaces. IEEE.

Adnan Boukhayma, Rodrigo de Bem, and Philip HS Torr. 2019. 3d hand shape and pose from images in the wild. In IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). 10843–10852.

Yujun Cai, Liuha Ge, Jianfai Cai, and Junsong Yuan. 2018. Weakly-supervised 3d hand pose estimation from monocular rgb images. In European Conf. on Computer Vision (ECCV). 666–682.

Thomas P Caudell and David W Mizell. 1992. Augmented reality: An application of heads-up display technology to manual manufacturing processes. In Proceedings of the twenty-fifth Hawaii international conference on system sciences. Vol. 2. IEEE, 659–669.

Junyeong Choi, Jungskik Park, Hanhoon Park, and Jong-Il Park. 2013. iHand: an interactive bare-hand-based augmented reality interface on commercial mobile phones. Optical Engineering 52, 2 (2013), 027286.

Wendy H Chuen and Tobias Höllerer. 2013. Real-time hand interaction for augmented reality on mobile phones. In Proceedings of the 2013 international conference on Intelligent user interfaces. ACM, 307–314.

Klaus Dorffmüller-Ulhaas and Dieter Schmalstieg. 2001. Finger tracking for interaction in augmented environments. In The IEEE Int. Symposium on Mixed and Augmented Reality. IEEE, 55–64.

DavidEigen, ChristianPuhursch, andRobFergus.2014. Depth map prediction from a single image using a multi-scale deep network. In Advances in neural information processing systems. 2366–2374.

Jakob Engel, Thomas Schöps, andDaniel Cremers. 2014. LSD-SLAM: Large-scale direct monocular SLAM. In European Conf. on Computer Vision (ECCV). Springer, 834–849.

Qi Feng, Hubert PH Shum, and Shigeo Morishima. 2018. Resolving occlusion for 3D pose and shape by learning from synthetic depth. In Thirty-Second AAAI Conference on Artificial Intelligence, 1197–1205.

Huan Fu, Mingming Gong, Chaozhu Wang, Kayhan Batmanghelich, and Dachen Tao. 2018. Deep ordinal regression network for monocular depth estimation. In IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). 2002–2011.

Ryo Furukawa, Ryusuke Sagawa, and Hiroshi Kawasaki. 2017. Depth Estimation Using Structured Light Flow—Analysis of Projected Pattern Flow on an Object’s Surface. In IEEE Int. Conf. on Computer Vision (ICCV). 4640–4648.

Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marín-Jiménez. 2014. Automatic generation and detection of highly reliable fiducial markers under occlusion. Pattern Recognition 47, 6 (2014), 2280–2292.

Liuha Ge, Zhou Ben, Yuncheng Li, Xiaofeng Ge, et al. 2018. Depth-based 3d hand pose estimation: From current achievements to future challenges. ACM Trans. on Graphics (SIGGRAPH Asia) 37, 6, Article 2366 (2018), 20:1–20:22.

Jie Song, Fabrizio Pece, and Christian Theobalt. 2017. Real-time hand tracking under occlusion from a single still image. ACM Trans. on Graphics (SIGGRAPH Asia) 37, 6, Article 2366 (2018), 20:1–20:22.

Franziska Mueller, Florian Bernard, Oleksandr Sotnychenko, Dushyant Mehta, Srinath Sridhar, Dan Casas, and Christian Theobalt. 2018. Generated hands for real-time 3d hand tracking from monocular rgb. In IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). 49–59.

Franziska Mueller, Dushyant Mehta, Oleksandr Sotnychenko, Srinath Sridhar, Dan Casas, and Christian Theobalt. 2017. Real-time hand tracking under occlusion from an egocentric rgb-d sensor. In IEEE Conf. on Computer Vision (ICCV). 1284–1293.

Raul Mar-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. 2015. ORB-SLAM: a versatile and accurate monocular SLAM system. IEEE transactions on robotics 31, 5 (2015), 1147–1163.

Alejandro Newell, Kaiyu Yang, andJia Deng. 2016. Stacked hourglass networks for human pose estimation. In European Conf. on Computer Vision (ECCV). Springer, 483–499.

Vassilis C Ncodomou, Iason Oikonomidis, Georgios Tzimiropoulos, and Antonis Argyros. 2018. Learning to Infer the Depth Map of a Hand from Its Color Image. arXiv preprint arXiv:1812.02486 (2018).

Hiroshi Ono, Brian J Rogers, Masao Ohmi, and Mika E Ono. 1988. Dynamic occlusion and motion parallax in depth perception. Perception 17, 2 (1988), 255–266.

Paschalis Panteleris, Iason Oikonomidis, and Antonios Argyros. 2018. Using a single RGB frame for real time 3D hand pose estimation in the wild. In IEEE Winter Conf. on Applications of Computer Vision (WACV). IEEE, 436–445.

Rafael Radkowski and Christian Stritzke. 2012. Interactive hand gesture-based assembly for realistic application scenarios. In Proceedings of the 2012 International Conference on Advances in Computer-Human Interactions. Citeseer, 303–308.

Qi Feng, Hubert PH Shum, and Shigeo Morishima. 2018. Resolving occlusion for 3D object manipulation with hands in mixed reality. The ACM Symposium on Virtual Reality Software and Technology, 119.

Huan Fu, Mingming Gong, Chaozhu Wang, Kayhan Batmanghelich, and Dachen Tao. 2018. Deep ordinal regression network for monocular depth estimation. In IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). 2002–2011.

RobertY Wang and Jovan Popovi´c. 2009. Real-time hand-tracking with a color glove. ACM Trans. on Graphics (SIGGRAPH Asia) 28, 5 (2009), 119.

Franziska Mueller, Florian Bernard, Oleksandr Sotnychenko, Dushyant Mehta, Srinath Sridhar, Dan Casas, and Christian Theobalt. 2018. Generated hands for real-time 3d hand tracking from monocular rgb. In IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). 49–59.