Virtual Agent Positioning Driven by Scene Semantics in Mixed Reality

Yining Lang[*]
Beijing Institute of Technology

Wei Liang[†]
Beijing Institute of Technology

Lap-Fai Yu[‡]
George Mason University

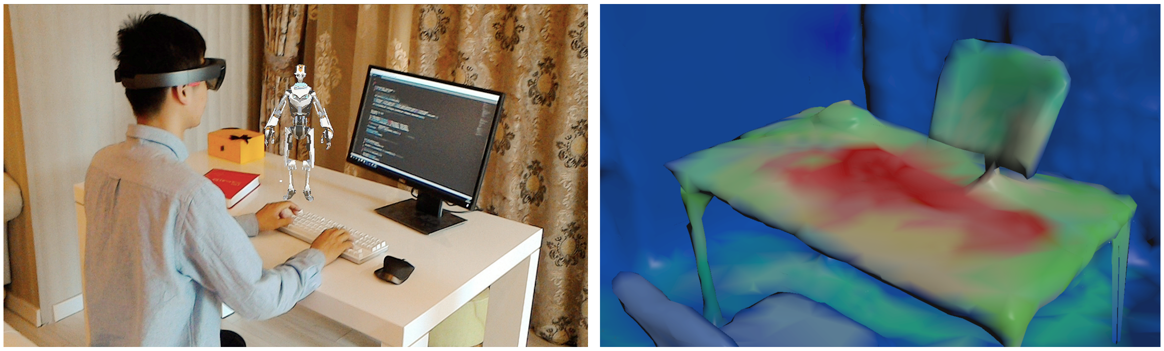


Figure 1: Left: by using a Hololens, a user interacts with a virtual agent whose location and orientation are optimized by our approach. Right: a heat-map showing the cost value of each candidate location for placing the virtual agent. The redder the color of a location is, the more suitable the location is for placing the agent.

Abstract

When a user interacts with a virtual agent via a mixed reality device, such as a Hololens or a Magic Leap headset, it is important to consider the semantics of the real-world scene in positioning the virtual agent, so that it interacts with the user and the objects in the real world naturally. Mixed reality aims to blend the virtual world with the real world seamlessly. In line with this goal, in this paper, we propose a novel approach to use scene semantics to guide the positioning of a virtual agent. Such considerations can avoid unnatural interaction experiences, e.g., interacting with a virtual human floating in the air. To obtain the semantics of a scene, we first reconstruct the 3D model of the scene by using the RGB-D cameras mounted on the mixed reality device (e.g., a Hololens). Then, we employ the Mask R-CNN object detector to detect objects relevant to the interactions within the scene context. To evaluate the positions and orientations for placing a virtual agent in the scene, we define a cost function based on the scene semantics, which comprises a visibility term and a spatial term. We then apply a Markov chain Monte Carlo optimization technique to search for an optimized solution for placing the virtual agent. We carried out user study experiments to evaluate the results generated by our approach. The results show that our approach achieved a higher user evaluation score than that of the alternative approaches.

Index Terms: Mixed Reality—Scene Understanding—Virtual Agent Positioning

1 Introduction

Virtual agents visualized by mixed reality devices enhance the ability of a user to interact with the real world in many practical scenarios, e.g. engineering, medicine, education, and so on [26]. Mixed reality applications aim to blend the virtual world with the physical world. Such blending needs to consider the semantics of a scene to create a natural and convenient user experience. Take virtual agent applications as an example. Some applications (e.g., Pokemon Go [34]) place virtual agents directly in front of the user without considering the scene geometry. This causes the virtual agents to be floating in the air, resulting in unnatural interactions. An alternative, common approach for placing virtual agents is to determine some surfaces on objects near the user, and to place virtual agents on these surfaces (e.g., in Young Conker [30]). Without considering scene semantics, such an approach may result in improper placement of virtual agents. For example, the virtual agent may occlude some key objects that the user is watching or interacting with, e.g. a TV or a computer screen, causing an inconvenient interaction experience.

To enable a virtual agent to interact with the user naturally, a mixed reality system should understand the physical world well. Suppose a virtual agent is visualized as a desktop personal assistant to guide a user about operating a computer. The agent should be properly placed and oriented with respect to the user to create a convenient and comfortable interaction experience. In addition, the agent should not block the screen of the computer. To position the agent well, the system should understand the semantics of the scene: where the user and the computer screen are, and which surfaces are available for the virtual agent to stand on.

In this paper, we introduce a novel approach which makes use of scene semantics for properly placing virtual agents in mixed reality applications. We focus on three aspects of the semantics which affect the user interaction experiences: what kinds of objects are in a scene, where the objects are, and where the user is. First, our approach reconstructs the 3D model of the scene in which the user interacts with the virtual agent, by using the RGB-D data captured by the depth cameras mounted on a mixed reality device (e.g., a Hololens). Second, our approach detects the key objects in the scene by using a MASK R-CNN method. Finally, through an optimization process, our approach determines a proper location and orientation for placing and visualizing the virtual agent according to the scene semantics and the purpose of the interaction. The optimization considers spatial factors (e.g., distance and orientation of the virtual agent relative to the user), as well as visibility of the key objects from the viewpoint of the user during the interaction. The major contributions of our work include:

- Introducing a new problem of properly positioning virtual agents according to scene semantics to interact with the user in mixed reality.
- Proposing a novel optimization-based approach for properly placing and orienting virtual agents based on scene semantics. We implemented our approach as a Hololens plug-in, which can be used for positioning a virtual agent in mixed reality.

[*] e-mail: lucky@langyining.com
[†] e-mail: liangwei@bit.edu.cn
[‡] e-mail: craigyu@gmu.edu

- Conducting an extensive user study to evaluate and validate the effectiveness of the proposed virtual agent positioning approach.

2 Related Work

2.1 Mixed Reality Auxiliary

Mixed reality can enhance the efficiency of working or learning [16, 27]. With the development of consumer-grade mixed reality devices, mixed reality is widely used as an auxiliary technology. Henderson et al. [15] explored the benefits of augmented reality for maintenance and repairment. Assisted by mixed reality, users can learn maintenance skills in a short time. Such an approach can save the cost of teaching by a human trainer. Kim et al. [23] investigated the effects of visual warning presentation methods on human performance in mixed reality driving. The proposed approaches can be used and further developed by researchers to better understand driver performance in augmented reality as well as to inform usability evaluation of future automotive mixed reality applications. Narzt et al. [33] proposed a navigation system that visualizes traffic information in the form of MR and enhances user interaction. Brandao et al. [5] used mixed reality to improve dismounted operators' situation awareness. Liu et al. [28] presented a novel augmented reality approach, through Microsoft HoloLens, to address the challenging problems of diagnosing, teaching, and patching interpretable knowledge of a robot.

To enhance the user interaction experience of mixed reality, some applications offer a virtual agent for interacting with users [4, 17]. Anabuki et al. [2] created an embodied conversational agent living in mixed reality space which can achieve some simple interactions with the user. Balcisoy et al. [3] created a virtual agent which can play checkers with the user. Miyawaki et al. [31] created a virtual agent for a cooking navigation system which uses the ubiquitous sensors in augmented reality. The cooking navigation system can recognize the progress of cooking and can show appropriate assistive contents accordingly. Besides, Inoue et al. [20] and Fadil et al. [10] proposed physical task supporting systems using virtual agents visualized by mixed reality. Holz et al. [18] proposed a mixed reality agent as museum guides, but their virtual agent needs to cooperate with a physical robot to achieve positioning and movement.

Compared with the previous works on auxiliary virtual agents, our approach focuses on properly positioning the agent to enhance the interaction between the virtual agent and the user in a mixed reality setting. The location of the user, the orientation of the user, as well as what the user wants to do are all considered during the optimization process.

2.2 Virtual Agent Positioning

The essence of positioning is to compute an appropriate position and orientation for the virtual agent. In the field of robotics, positioning is an important problem. Se et al. [37] described efficient algorithms for positioning a mobile robot in an environment with landmarks. Chong et al. [6] proposed an interactive method which can plan collision-free paths for the robot using mixed reality. Sud et al. [39] presented a novel approach for real-time path planning of virtual agents in complex dynamic scenes. Narang et al. [32] simulated movement interactions between avatars and agents in virtual worlds using human motion constraints. Elber et al. [9] investigated object positioning and displaying in virtual environments. Kato et al. [22] investigated virtual object interaction and user tracking for a table-top mixed reality interface.

Deep learning approaches have also been applied for agent positioning. Akiyama et al. [1] proposed a novel agent positioning mechanism for dynamic environments. They computed suitable positions for each agent by a deep learning method according to the current status of the environment. Korjani et al. [25] proposed a method for dynamic autonomous agent positioning based on a modular neural network.

Most of the existing works achieve the positioning by detecting markers or by prediction using a deep learning method. By explicitly encoding spatial and visibility considerations as cost terms, our approach optimizes the location and orientation of the virtual agent to enhance its interaction with the user. For example, the visibility cost ensures that the virtual agent does not block the user from the key objects that he is interacting with. Other ergonomic factors can be similarly incorporated into our optimization framework.

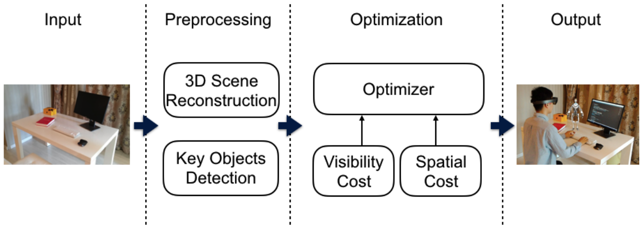


Figure 2: Overview of our approach.

2.3 Human-centric Scene Understanding

A major goal of our approach is to facilitate the interactions between a virtual agent and a user through proper positioning of the virtual agent. To achieve this, our approach needs to obtain the semantics of the scene with regard to human-centric guidelines. Human-centric understanding of environments, which is in line with the concept of affordances introduced by Gibson [13], has a long history.

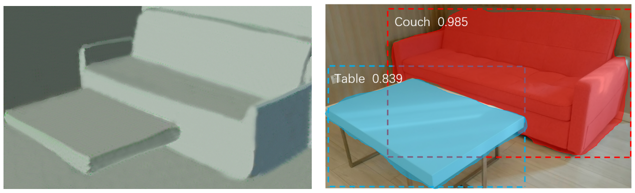
Some recent works learn the semantics of a scene to enhance the interaction between the scene, and the virtual agent or the user. Savva et al. [35] proposed a method which aims to understand actions by modeling how people interact with their 3D environments. More specifically, they modeled the space of interactions between the human body and the geometry of the environment in which actions take place. This allowed them to make predictions about the functionality of the 3D scene geometry and the implied functional characteristics of virtual scenes. Furthermore, they learned interactions between human and other objects in the daily scene from observations [36]. Fisher et al. [11] proposed a data-driven method for functional 3D scene modeling and represented scene functionalities through virtual agents, associating object arrangements with the activities for which they are typically used. Wang et al. [40] presented a convolutional neural network based approach for indoor scene synthesis. Jiang et al. [21] proposed a systematic learning-based approach to the generation of massive quantities of synthetic 3D scenes and arbitrary numbers of photorealistic 2D images. Kim et al. [24] proposed an algorithm for automatically predicting a static pose that a person would need to adopt in order to use an object. Fu et al. [12] proposed an approach for the adaptive synthesis of indoor scenes via activity-associated object relation graphs. Using the labeled human positions and directions in each plan, they detected the activity relations and computed the coexistence frequency of object pairs to construct activity-associated object relation graphs.

Based on the human-centric guidelines, our approach achieves the positioning by considering the scene semantics surrounding the user. Compared with the other works, we focus on the interaction between a user and a virtual agent in addition to understanding how the user may interact with the scene. Therefore, we need to not only understand the semantics of the surrounding scene where the user performs tasks but also to automatically select a suitable location and orientation for the virtual agent to enhance the interaction.

3 Overview

The goal of our work is to position a virtual agent with a proper location and orientation relative to a user in a mixed application according to the semantics of a scene. Within the constraints of the scene, the location and the orientation of the positioned virtual agent need to be appropriate and consistent with the scene semantics. The framework of our approach is shown in Figure 2, consisting of two parts: **preprocessing** and **optimization**.

In the preprocessing part, our approach obtains the semantics of the scene, including what and where the objects are. First, we scan



(a) 3D Scene Reconstruction

(b) Key Object Detection

Figure 3: Preprocessing results. (a) A 3D scene reconstructed by a Hololens. (b) Object detection results achieved by the Mask R-CNN object detector. The bounding boxes, categories and the classification confidence scores of the detected objects are shown. The two objects, shown in blue and red, are also automatically segmented by the approach.

the scene via a Hololens helmet to obtain the 3D model with texture of the scene. Second, our approach detects some predefined key objects for relevant interaction tasks via the Mask R-CNN approach. After these two steps of preprocessing, we obtain the 3D model of the room and the detected key objects.

In the optimization part, our approach optimizes the location and orientation of the virtual agent iteratively, considering the semantics of the scene. A cost function with a visibility term and a spatial term is designed to evaluate how well the location and orientation of the virtual agent is. The visibility term ensures the user's view of watching key objects would not be occluded by the virtual agent. The spatial term controls the distance and orientation of the virtual agent with respect to the user for a natural and convenience interaction. A Markov chain Monte Carlo optimization algorithm is applied to search for a solution.

4 Preprocessing

In this section ,we discuss how to obtain the semantic information of a scene. We perform two steps: 3D scene reconstruction to obtain the geometry and texture of the scene; and key object detection to obtain the scene semantics (including what and where the objects are in the 3D scene.

4.1 3D Scene Reconstruction

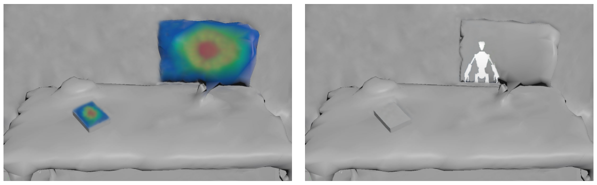
To obtain the geometry of the scene, we use the spatial mapping technique of the Hololens to scan the scene. For a room in an apartment, it generally takes about two minutes to scan the scene and reconstruct the 3D model (depending on the size of the scene). The 3D model is represented by a set of dense triangular meshes. Figure 3(a) shows an example of the 3D model, scanned and reconstructed from a scene .

The texture of the 3D model is captured synchronously during the scanning process using the cameras mounted on the Hololens. During the scanning, we capture a video stream through the RGB cameras. Then the stream is used to color a UV map and texture the 3D model. The implementation is similar to the work of Dong et al. [8]. In this way, we apply the corresponding texture for the 3D model.

4.2 Key Object Detection

During the interaction, there could be some objects that the user is interacting with and that should not be occluded by a virtual agent. We term the objects, whose blockage will affect the user's interaction experience, as key objects. These key objects are closely related to the user's interaction task. For our experiments, we define 3 types of interaction tasks: communication, teaching and guiding. Furthermore, we define 5 key objects for each interaction task. It is worth to note that the predefined key object set can be extended according to different interaction tasks.

In our experiment, the user may inform the virtual agent about his interaction task through voice control via the Hololens. For example, if the user wants to work with the computer, he could say: ”can you teach me to use the computer?” Then, the keyword ”teach” could be recognized by the voice recognition module of the Hololens and our approach can optimize the virtual agent with a teaching task setting. The key objects predefined in the teaching task, such as computers, blackboards, books, and so on, will be considered when positioning the virtual agent.



(a) The weight matrix ω.

(b) The occlusion map I.

Figure 4: A visualization of the visbility cost. (a) The weight matrix denotes how important each location on the key object is. The visibility cost penalizes occlusion of important locations by the virtual agent. (b) The occlusion map rendered from the user's viewpoint. The white region shows the locations on the key object occluded by the virtual agent.

To obtain what and where the key objects are in a 3D scene, we first detect the key objects in 2D images (from the captured video stream in the scanning process) by the Mask R-CNN approach ([14]). The Mask R-CNN approach can efficiently detect objects in an image while simultaneously generating a high-quality segmentation mask for each instance. Figure 3(b) shows an example of the key object detection, in which the sofa and the table are detected and applied the corresponding masks via the Mask R-CNN approach.

After that, we map the detected key objects to the 3D model of the scene. We estimate the camera's parameters roughly from the 2D image using the method of Horry et al. [19]: extracting the vanishing point based on the principle of perspective projection, and estimating the camera parameters. With the estimated parameters, we set up a camera in the reconstructed 3D scene and render an image. This rendered image corresponds to the 2D image. Using these two images, we can calculate the corresponding positions of the key objects in the 3D scene.

This preprocessing step provides the semantics of the scene: what kinds of objects are in the scene and where the objects are.

5 Technical Approach

The goal of our approach is to position a virtual agent with a proper location and orientation with respect to the semantics of a scene. We solve the problem of positioning the virtual agent by an optimization process. A cost function, consisting of a visibility term and a spatial term, is designed to measure how proper a positioning of the virtual agent is. By optimizing the cost function, our approach searches for a desired location and orientation for the virtual agent.

We represent a virtual agent configuration in a scene as a tuple $a = (l_x, l_y, l_z, o_x, o_y, o_z)$, where (l_x, l_y, l_z) contain the 3D coordinates of the virtual agent, (o_x, o_y, o_z) contain the orientation angles of the virtual agent about the x, y and z axes respectively.

The total cost function is defined as:

$$C_{\text{total}}(a) = \lambda_{\text{V}} C_{\text{V}}(a) + \lambda_{\text{S}} C_{\text{S}}(a). \tag{1}$$

$C_{\text{V}}(a)$ is the visibility cost, which penalizes the situations where the key objects are occluded from the viewpoint of the user. $C_{\text{S}}(a)$ is the spatial cost, which encourages the proper location and orientation of the virtual agent from the user, learned from crowds. λ_{V} and λ_{S} are the weights of the corresponding cost terms. To balance the effect of each cost term, we make the weights sum to 1. In our experiment, the weights are set as $(0.5, 0.5)$ by default. Users can also adjust the weight of two cost functions according to their personalized needs as the ablation studies show.

5.1 Visibility Cost

The visibility of key objects (e.g., a computer) from the user's viewpoint is an important factor affecting the interaction experience, because the key objects' fundamental functionality is compromised if their frontal surfaces are occluded by the virtual agent. To this end, we devise a visibility cost to restrain the occlusion issue. If the virtual agent blocks the key objects from the user's viewpoint, the visibility cost will increase to penalize such a configuration.

Furthermore, considering that an occlusion near the center of the key objects is likely to affect the interaction experience more severely compared to an occlusion near the key objects' boundary, we define a weight matrix to penalize the former situation by a larger extent. The visibility cost is defined as:

$$C_{\mathrm{V}}(a) = \frac{1}{||\omega||_2 \cdot I_{max}} \sum_{i,j} I(i,j) \cdot \omega(i,j), \tag{2}$$

where I is an occlusion map, rendered from the viewpoint of the user. $I(i,j)$ is 1 if the pixel at (i,j) shows an overlapping between the virtual agent and a key object (i.e., the virtual agent is occluding the key object), otherwise $I(i,j)$ is 0. As shown in Figure 4(b), the white region corresponds to the occluded region of the key object. We also rendered an image I' without the virtual agent from the same viewpoint with I. I_{max} represents the number of pixels in I' occupied by all key objects. In our experiment, the number of the occluded key object is not limited to one.

ω is the weight matrix, whose dimensions are equal to those of the rendered image I. $||\omega||_2$ means the 2-Norm of matrix ω. $||\omega||_2$ could be zero. However, in our experiment, the solution space is limited, so that this situation does not exist in the actual optimization. $\omega(i,j)$ is 0 if pixel (i,j) does not belong to any key object. If pixel (i,j) belongs to the n-th object, we set $\omega(i,j) = \frac{e^n_{max}}{1+e^n_{i,j}}$, where e^n_{max} represents the maximum Euclidean distance from the center pixel of the key object n to the marginal pixel of it. $e^n_{i,j}$ represents the Euclidean distance from the center pixel of the key object n to the pixel (i,j). Figure 4(a) shows two key objects in the scene with the corresponding weight matrices visualized. A large occlusion area or an occlusion near the center of the key object results in a high visibility cost. Therefore, the occlusion of key objects by the virtual agent will be restrained effectively by the visibility cost.

5.2 Spatial Cost

Spatial factors, namely the location and orientation of the virtual agent relative to the user, are vital for an interaction. For the location, if the distance of the virtual agent to the user is less than the minimum social distance [38], the user may feel uncomfortable. In contrast, if the distance is too far, the user may not be able to see the virtual agent clearly. On the other hand, if the virtual agent is not properly oriented (e.g., not facing the user), the interaction would also be unnatural and uncomfortable.

To obtain a proper location and orientation, we learn from crowds and fit a mixed Gaussian distribution with K Gaussian kernels. The spatial cost term is defined as:

$$C_{\mathrm{S}}(a) = 1 - \sum_{k \in K} \pi_k G_k(d, \theta; \mu_k, \Sigma_k), \tag{3}$$

where d and θ represent the distance and orientation of the virtual agent relative to the user, respectively. d is calculated by the Euclidean distance between the virtual agent and the user. θ is the angle between the normal vectors of the user and the virtual agent on the x-z plane, which denote the frontal directions of the user and the virtual agent. G_k represents the k-th Gaussian kernel with the mean μ_k and the covariance matrix Σ_k. π_k is the weight of the k-th Gaussian kernel. μ_k, Σ_k and π_k are learned from crowds.

Learning from Crowds. For different interaction tasks, the comfortable distance and orientation of the virtual agent could be different. Therefore, we fit one Mixed Gaussian distribution for each interaction task to model the conventional choices of crowds.

We recruited 30 participants, ranging from 18 to 50 years old, to collect their conventional choices. The three interaction tasks are performed in 3 scenes which contains typical furniture, such as television, couch, chairs. For each task, we placed a virtual agent randomly in front of the participant. Each participant was asked to wear a Hololens helmet and to observe the virtual agent in the scene. The participant could move and rotate the virtual agent, until he felt comfortable to interact with it. Then we recorded the current configuration of the virtual agent. Each participant repeated the choice 5 times for each interaction task.

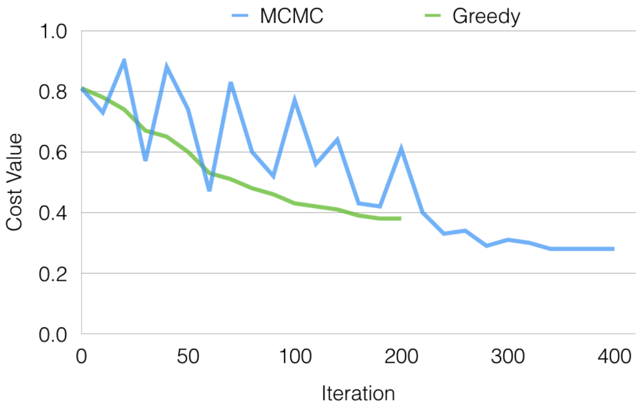


Figure 5: A comparison between optimizing the virtual agent's configuration using the MCMC algorithm and the greedy algorithm. The MCMC algorithm finds a solution of a lower cost. The optimization finishes in about 2 seconds.

All participants performed the choices for three tasks. In total, we obtained 150 locations and 150 orientations for each interaction task. Based on the collected data, we calculated the distance and orientation of the virtual agent relevant to the user. Then, we use the maximum likelihood approach [7] to estimate μ_k, Σ_k and π_k.

By the spatial cost function, the location and orientation of the virtual agent will tend to follow the preferred configuration which is learned from crowds for a particular interaction task.

6 Optimization

We search for a proper location and orientation of the virtual agent by an optimization process. The goal of the optimization is to find a location (l_x, l_y, l_z) and an orientation (o_x, o_y, o_z) that minimize the total cost function $C_{\mathrm{total}}(a)$. During the optimization, our approach searches for the suitable location and orientation of the virtual agent for an interaction task against the defined cost functions. A Markov chain Monte Carlo (MCMC) sampler is applied to explore the solution space and it has the ability of jumping out from a local minimum by the strategy of accepting a non-optimal choice with a certain acceptance probability.

6.1 Move

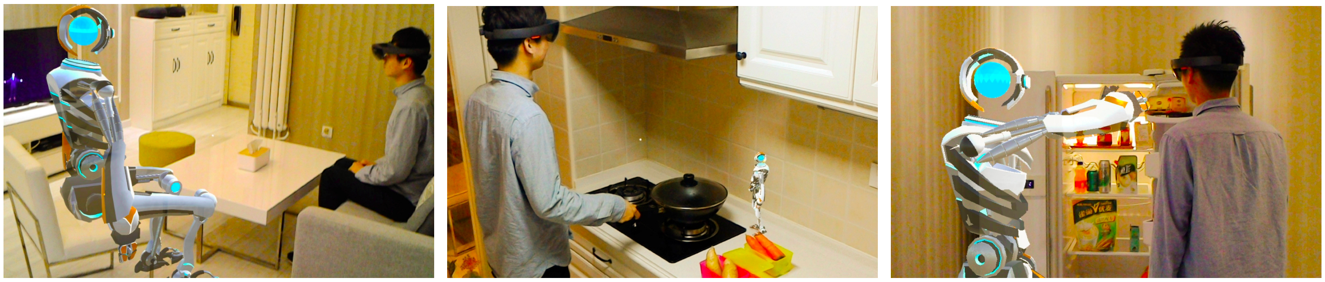
The MCMC sampling works by choosing a "move" to generate a new sample. We define two moves to explore the solution space: distance move and orientation move, corresponding to adjusting the location (l_x, l_y, l_z) and the orientation (o_x, o_y, o_z) of the virtual agent. The sampling process performs these two moves alternately.

Distance Move. In the distance move, the sampler generates a new location (l'_x, l'_y, l'_z) based on the current location (l_x, l_y, l_z):

$$(l_x, l_y, l_z) \rightarrow (l'_x, l'_y, l'_z), \tag{4}$$

The distance move is achieved along the surface of the 3D model of the scene. From the current location, we use a sampling radius to define the sampling range. Since we sample along the surface, we use the geodesic distance to define the sampling radius. Within the sampling range, we sample a new location (l_x, l_y, l_z) randomly in each move. The sampling radius is set as 20 cm at the beginning. It decreases by 1cm every 10 iterations of the optimization until it reaches 1cm.

Orientation Move. In the orientation move, the sampler generates a new orientation (o_x, o'_y, o_z) based on the current orientation (o_x, o_y, o_z):



(a) Communication Task (b) Teaching Task (c) Guiding Task

Figure 6: The results of positioning the virtual agent in three interaction tasks. For more intuitive visualization, we use a third-person view to render the virtual agent. (a) The virtual agent chats with the user about the TV content. (b) The virtual agent teaches the user how to cook. (c) The virtual agent introduces the food in the refrigerator.

$$(o_x, o_y, o_z) \rightarrow (o_x, o'_y, o_z), \tag{5}$$

where $o'_y = o_y + \delta_y$, and $\delta_y \in [-\delta_{max}, \delta_{max}]$ is a random value. The initial value of δ_{max} is 60°. It decreases by 10° every 60 iterations of the optimization until it reaches 10°.

6.2 Solution Space Constraints

To speed up the optimization, we constrain the solution space by three conditions:

Range Constraints. We constrain the solution space to be a fan-shaped region in front of the user. The fan-shaped region is defined using the location of the user as the fan's apex, with a radius of 2.5 meters and an angle of extension of 120 degrees. The distance move will be performed within this region to sample a new location. See Figure for an illustration.

Surface Constraints. Some solutions may cause the virtual agent to be floating in the air, which is not unnatural. So we only consider placing the virtual agent on the upper surface of an object in the scene or on the floor.

Occlusion Constraints. We also prevent the virtual agent from crashing into objects. In the sampling process, our approach performs a collision detection between the virtual agent placed at the sampled location and orientation, and objects in the scene. If a collision occurs, another random location or orientation will be sampled.

6.3 Sampling

At the beginning of the optimization, the virtual agent is initialized with a random position and a random orientation. As the optimization proceeds, the new location (l'_x, l'_y, l'_z) and orientation (o_x, o'_y, o_z) are proposed and evaluated using the total cost function. The proposed location and orientation are accepted or rejected according to the Metropolis criterion:

$$Pr(a'|a) = \min(1, e^{\frac{1}{T}(C_{\text{total}}(a) - C_{\text{total}}(a'))}), \tag{6}$$

where T is the temperature of the simulated annealing process. We set the value of T empirically as 300 at the beginning of the optimization, allowing the optimizer to explore the solution space more aggressively. The value of T drops by 0.5 every iteration of the optimization until it reaches 0, allowing the optimizer to refine the solution near the end of the optimization. We terminate the optimization if the absolute change in the total cost value is less than 5% over the past 25 iterations. Generally, it takes about 400 iterations to obtain the solution for the three types of interactive tasks in our experiments.

Figure 5 shows an example of optimizing a position for the virtual agent using the MCMC approach and the Greedy approach. The MCMC approach obtains a solution with a lower cost value (0.28) compared to the Greedy approach (0.39). The cost value of the greedy algorithm experiment didn't change from about iteration 175 to about iteration 200. Thus, the optimization (green line) stopped at about iteration 200. Since the MCMC algorithm can accept a solution with a cost higher than that of the current solution with a certain acceptance probability, the sampling is capable of jumping out from a locally optimal solution. This prevents the sampling from being performed locally, and eventually locating a more optimal solution with a lower cost value.

7 Experiments

7.1 Implementation

We implemented our approach using C# and Unity 5.6. Due to the limited computing power of the Hololens, we ran the optimization approach on a PC equipped with 16GB of RAM, a Nvidia Titan X graphics card with 12GB of memory, and a 2.60GHz Intel i7-5820K processor. The position of the user and the optimized results are transmitted between the Hololens and the PC by WiFi. Based on the transmitted results, the user can interact with a virtual agent properly placed and oriented according to the optimization result via the Hololens mixed reality headset. The time of optimization and transmission is about 4 seconds on average.

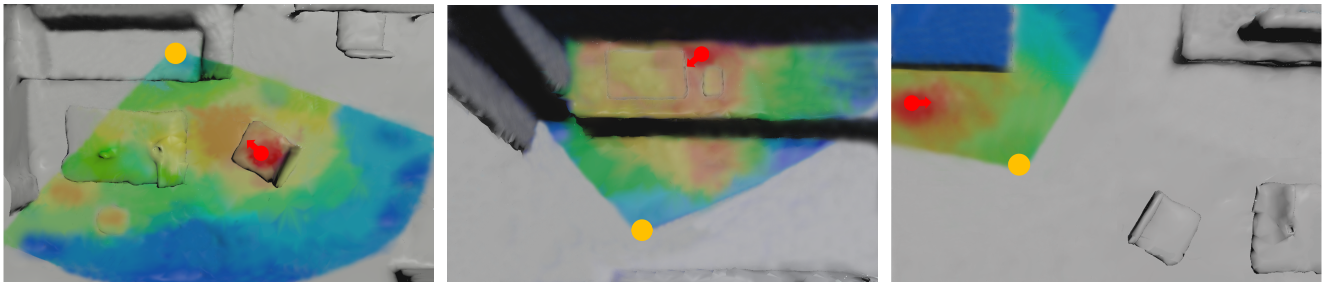
7.2 Interaction Tasks

We design three types of interaction tasks which cover the common interactions between a user and a virtual agent, to demonstrate our approach: communication task, teaching task, and guiding task.

Communication Task. Communication is a very common interaction between a user and a virtual agent. It is important to have a suitable distance and orientation during the communication process. If the distance is too close, it may cause discomfort due to the invasion of privacy; if the distance is too far, the dialogue may not be heard and the agent may not be clearly seen. In addition, the orientation of the virtual agent will also affect the user's interaction experience. If the orientation deviates from a reasonable range, the virtual agent could be facing away from the user, which hinders communication. Besides, if some key objects (e.g., a TV) exist in the scene, we need to avoid the problem of occlusion.

Figure 6(a) shows a result generated for a communication task. Optimized by our approach, the virtual agent sits on a chair and communicates with the user, maintaining a certain social distance with a natural orientation from the user. During the interaction, the virtual agent may change its pose according to the nearby scene. If the conversation takes place in an open area, our approach assigns a standing pose for the virtual agent. If the sampled location happens to be on the top surface of a stool of sofa, our approach assigns a sitting pose for the virtual agent. Since the user is watching the TV when chatting with the virtual agent, the visibility cost will give a certain constraint on the location of the virtual agent, preventing the virtual agent from blocking the TV from the user.

Teaching Task. An important role a virtual agent plays is to teach or to assist a user. For some tasks, such as cooking, assembling furniture, and repairing things, it could be unintuitive to learn by reading instruction manuals, while face-to-face teaching is expensive. Teaching via a mixed reality device provides a low-cost yet promising solution that mimics face-to-face teaching.



(a) Communication Task

(b) Teaching Task

(c) Guiding Task

Figure 7: Visualization of the total cost in the three interaction tasks. The color corresponds to the lowest cost value at each location considering all orientations at that location. The redder the region, the lower the cost value. The yellow circle shows the current location of the user. The red circle shows the virtual agent's location and orientation selected by our optimization approach. The gray region is excluded from the solution space due to the range constraint.

Figure 6(b) shows an example. The virtual agent stands nearby the stove and teaches the user how to cook. We may notice that the distance between the user and the virtual agent in the teaching task is closer than that in the communication task. Such a prior distance was learned from the crowds. A possible reason for the shorter distance for teaching tasks is that teaching often involves close interaction. Besides, the size of the virtual agent varies according to the surrounding scene. If there is a platform around the virtual agent, and if it is more suitable for teaching, our approach sets its height to be 30cm empirically. This height can be adjusted according to the user's own needs. In addition, the visibility cost prevents the virtual agent from occluding the key objects (vegetables) in this interaction task.

Guiding Task. Sometimes, the user may need to constantly change his position in the scene, which is different from the situation in the communication and teaching tasks where the position of the user rarely changes. For example, when the user enters a new apartment, he may want a virtual agent to guide him. As the user walks around to explore the apartment, the virtual agent needs to update its optimized position quickly to keep up with the user. We design the guiding task to cope with this situation.

Figure 6(c) shows an example. The virtual agent is introducing the food in the refrigerator to the user. The visibility cost prevents the virtual agent from occluding the key objects (the refrigerator and the drinks) from the user. The spatial cost maintains an appropriate distance between the user and the virtual agent.

Our approach can handle different types of interaction tasks. It makes the interaction between a virtual agent and the user more convenient and user-friendly. We conducted a user study to evaluate the virtual agent positioning for the three different types of interaction tasks, and compared the results achieved by our approach with two other positioning approaches. We will discuss them in Sec 8.

7.3 Visualization of Cost Function

To illustrate the effect of the defined cost function more intuitively, we visualize the cost values of the entire solution space with a heat-map as shown in Figure 7. For the visualization result, we plot the value of $1 - C_{\text{total}}$ at each point on the 3D model. So, the redder a location is, the lower the cost value it carries, and the more suitable it is to place the virtual agent.

Figure 7(a) shows a scenario of the communication task. The middle region of the chair on the right has the deepest red, which means it is the best region for placing the virtual agent. This region has a proper distance between the user and the virtual agent. The stool in front of the coffee table also has an acceptable distance, but the virtual agent would occlude the key object (TV) if it sat there. Other areas are mainly green or blue, hence are less suitable for positioning the virtual agent.

Figure 7(b) shows the virtual agent teaching the user to cook near a stovetop. The region on the right of the stovetop gets lower cost values. Since there are key objects (vegetables) on the right side of stovetop, the middle part of the red region is yellow, which means its cost value is higher than the surrounding area. Placing the virtual agent there would have blocked the vegetables from the user. Besides, the regions around the stovetop (a key object) is also yellow (having a higher cost value), because placing the virtual agent there would have blocked the user from seeing the stovetop.

Figure 7(c) shows a guiding task in which the user stands nearby the refrigerator and the virtual agent introduces the food inside. The right side in front of the fridge gets the lowest cost value, which is most appropriate for placing the virtual agent for guiding the user. The further away from the refrigerator, the higher the cost value is. It is worth noting that the region on the left front of the refrigerator is green, because that region is too close to the user.

Through the visualization of the cost values of the solution space, it can be seen that the definition of our cost functions are effective and the positioning of the virtual agent can be optimized for three types of interaction tasks appropriately .

7.4 Ablation Experiment

To analyse the influence of the weights in the cost function, we did an ablation experiment. The cost function consists of the visibility cost and the spatial cost, and the corresponding weights are λ_{V} and λ_{S} respectively. We implemented the ablation experiment by changing the weights.

First of all, we want to know what the result is if we do not consider the visibility cost term. We set λ_{V} and λ_{S} as 0 and 1. The results are shown in Figure 8(a). Without the visibility cost constraint, the distance and orientation of the virtual agent relative to the user are still reasonable. However, the key object (computer screen) is occluded by the virtual agent, which will affect the user's interaction experience.

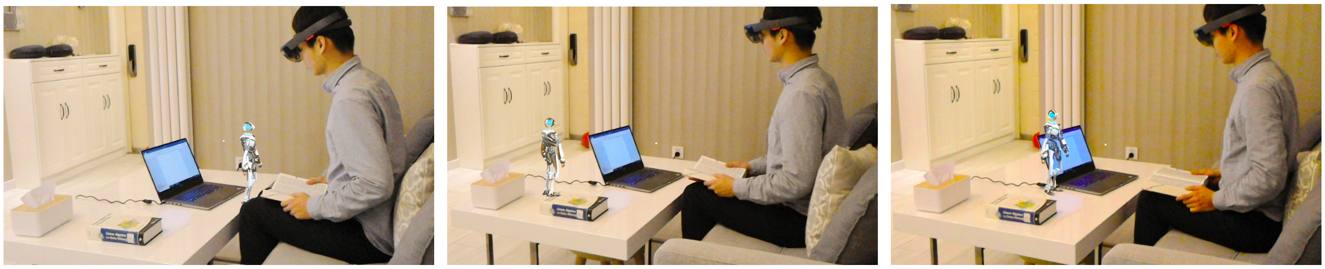
Secondly, we test the approach without the spatial cost term. We set λ_V and λ_S as 1 and 0. As Figure 8(b) shows, the virtual agent does not occlude the key object at all with the visibility cost term. However, since the weight of the spatial cost term is 0, the distance between the virtual agent and the user is too far, and the orientation is not suitable for interaction. Therefore, without the spatial cost, the virtual agent can only ensure no occlusion with the key objects. The improper distance and orientation may cause inconvenience in user interaction.

Finally, we test the optimization results when λ_V and λ_S are set as 0.5 and 0.5. As Figure 8(c) shows, the optimization result is much more reasonable than the first two. However, the virtual agent still has a little overlap with the key object (computer screen). It is because the visibility cost has an equal weight compared to the spatial cost, so when the distance and orientation are appropriate, even with some occlusion, the total cost is still low, causing the optimizer to accept this result.

In general, this set of experiments demonstrate the rationality of the visibility cost and spatial cost design. The two cost terms work together to achieve a reasonably positioning of the virtual agent.

8 User Study

We carried out a user study to evaluate the effectiveness of the proposed virtual agent positioning approach. We attempted to verify



(a) $\lambda_V = 0, \lambda_S = 1$ (b) $\lambda_V = 1, \lambda_S = 0$ (c) $\lambda_V = 0.5, \lambda_S = 0.5$

Figure 8: Results of the ablation experiments. λ_V and λ_S are the weights of the visibility cost and the spatial cost.

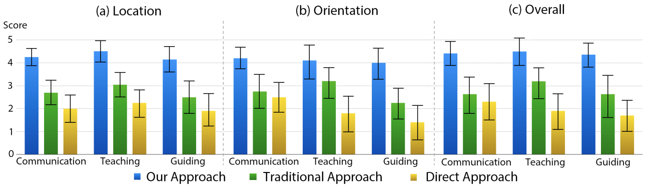


Figure 9: Location, orientation, and overall evaluation scores for the virtual agent placed by our approach, the traditional approach and the direct approach. Error bars denote the standard errors of the mean.

two conditions, by two experiments: first, a comparison experiment to testify whether our approach can position the virtual agent more reasonably than other approaches; second, a regression analysis to verify the rationality of the designed cost function.

Participant. 30 volunteer participants were recruited who were unaware of the purpose of the user study. The participants included 15 males and 15 females whose ages ranged from 18 to 50. All the subjects reported normal or corrected-to-normal vision with no color-blindness. 20 subjects reported that they did not have any experience in mixed reality devices using.

Data. We tend to compare three approaches: our approach, traditional approach, and direct approach.

The first approach is our approach. The virtual agent is positioned with the location and orientation optimized by our approach based on the semantics of the scene.

The second approach is a traditional approach which has been widely used in AR applications nowadays (e.g., Young Conker [30]). In this approach, the user needs to set a virtual place (e.g., a desktop) and the virtual agent will be positioned on the plane randomly. The orientation is generated within a range of $-30°$ to $+30°$ around the angle of facing towards the user.

The third approach is a direct approach which has been commonly used in some early AR applications (e.g., Pokemon Go [34]). In this approach, the virtual agent is directly positioned in front of the user. The orientation was generated as the second approach does.

To reduce the bias, besides the former three approaches, we generated extra 47 configurations of the virtual agent for each interaction task. The 47 results were generated based on our optimized result: the location generated randomly within 1m range of our optimized location along the upper surfaces of the objects in the scene; the orientation generated randomly within a range of $-45°$ to $+45°$ around our optimized orientation. The generated virtual agent were mixed with the other virtual agents generated from the 3 compared approaches. All virtual agents appeared with random orders. The participants did not know which approach the current virtual agent was generated from.

Procedure. The user study was carried out via a Hololens with a controller. Before the evaluation, each participant went through a 5-minute warm-up session so that they got familiar with using the Hololens helmet. A MR application, Holotour [29], was used as the warm-up program, in which the user could learn about the basic operation (e.g., observing, selecting) via the Hololens.

Instructions for each experiment were provided via a window in the MR environment; participants read the instructions and indicated, with their controller, when they were ready to proceed to the next window. This was done to prevent any bias resulting from verbal instructions from the experimenter. Participants were, however, told to notify the experimenter if they had any questions during the instruction period.

To eliminate the bias caused by the task order, we take the strategy of counterbalancing. All participants were divided into three groups randomly. Then a 3×3 Latin squares is used to arrange the task order for each group.

Each participant was required to observe the virtual agent via the Hololens. Then he was required to score for the virtual agent in terms of the location, the orientation, and the overall impression by choose the corresponding score on a scoreboard via the controller. The scores varied from 1 to 5. It took about 10 seconds for a participant to score for a virtual agent.

Participants were told to report any sickness or discomfort with the apparatus at any point in the experiment and that they could terminate their session at any time.

Metrics. We use three metrics to evaluate and analyze the performance of positioning the virtual agent by each approach: **location score**, **orientation score**, and **overall score**. These scores represent how comfortable the participants feel about the virtual agent in terms of the location, orientation, and overall impressions. All scores range from 1 to 5. The higher a score is, the better the evaluated term is.

8.1 Statistical Analysis

We compare the results generated from the three approaches by the mean and standard deviation of the users' ratings. We also carried out ANOVA tests on the scores to analyze the significance. Figure 9 shows the average scores and their standard deviation of the location, orientation, and overall over three interaction tasks.

Location. The average location scores over three interaction tasks with our approach achieved the highest score 4.29, followed by the traditional approach 2.76, and the direct approach 2.05.

Take the communication task as an example. The mean score of the location for our approach, the traditional approach, and the direct approach was 4.23 ± 0.43, 2.7 ± 0.59, and 2.03 ± 0.67, respectively. The difference between our approach and the traditional approach ($F_{[2,58]} = 83.13$, $p = 9.55E - 13 < 0.05$), or between our approach and the direct approach ($F_{[2,58]} = 119.34$, $p = 1.08E - 15 < 0.05$) was statistically significant. The direct approach got the lowest score in this task. Since the direct approach placed the virtual agent in front of the user without considering any scene semantics, the virtual agent may float in the air or overlap with other objects, which resulted in the lowest rating score. The traditional approach placed the virtual agent on a plane, which was more reasonable than the directed approach. Whereas, it did not consider the occlusion of the key objects in the scene, which may influence the interaction experiences, e.g. placing the virtual agent on a table in front of the user and causing occlusion when the user was watching TV. Optimized by our spatial cost and visibility cost, the virtual agent could keep an appropriate distance when interacting with the user and avoid overlapping with other key objects, so that the location score of our approach was better than the other two approaches.

Orientation. For the orientation evaluation, over three interaction tasks, the average score of the results optimized by our approach

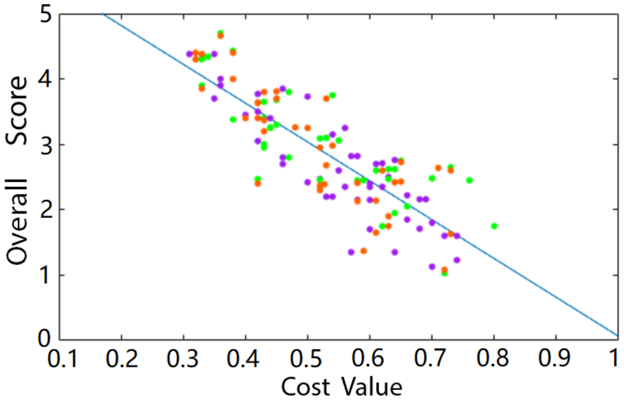


Figure 10: Scatter plot of the overall scores and the cost values in three interaction tasks. The green, red and purple points represent the results from the communication, teaching and guiding task, respectively. The blue line is the linear regression plot. The scores are effectively fit by a linear model. It indicate that there is a significant negative association between the overall values and the cost values.

were higher than 4.00, followed by the traditional approach with an average score about 2.72. However, the scores of the traditional approach varied in different interaction tasks (e.g., 3.20 in the teaching task and only 2.23 in the guiding task). The worst was the direct approach, with an average score ranging from 1.40 to 2.50.

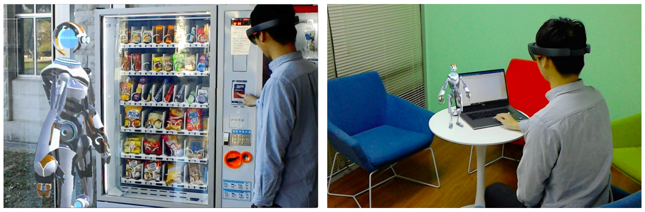
Since the traditional approach and the direct approach tended to place the virtual agent with a random orientation within a range, in some scenarios, the user may feel uncomfortable. For example, in the guiding task, when the virtual agent needed to introduce the foods in a refrigerator to the user. With the random orientation, the virtual agent did not see the food nor the user. Therefore, the orientation scores of the other two approaches were lower than that of our approach.

It is interesting to observe in Figure 9(b). Although both the traditional approach and the direct approach used the same strategy to generate the orientation for the virtual agent, the average scores of the orientation were different. The traditional approach got a higher score, compared with the direct approach. The reason may be that the user's evaluation on the orientation depended on the location partially. That is, the influence of the location and the orientation on the interaction experiences were coupled together. So our approach learned the preferences from crowds without decomposing these two factors, which resulted in stable performance across tasks.

We also did an ANONA test on the orientation scores. The results showed statistically significant between our approach and the traditional approach (communication:($F_{[2,58]} = 130.44$, $p = 1.80E-16 < .05$), teaching: ($F_{[2,58]} = 111.85$, $p = 3.70E-15 < .05$), guiding: ($F_{[2,58]} = 97.11$, $p = 5.30E-14 < .05$)), our approach and the direct approach(communication: ($F_{[2,58]} = 229.30$, $p = 7.90E-22 < .05$), teaching($F_{[2,58]} = 236.97$, $p = 3.80E-22 < .05$), guiding:($F_{[2,58]} = 179.56$, $p = 2.10E-19 < .05$)).

Overall Score. Considering the overall scores, the average score of our results was the highest. The overall average score for the communication task (4.40) and the guiding task (4.37) were higher than that of the two individual scores (communication-location 4.23, communication-orientation 4.20, guiding-location 4.13, guiding-orientation 4.00). We also did an AVOVA test on the overall scores. The results showed that our approach outperformed the other two approaches with statistical significance. Please refer to the supplementary for more details.

Through the statistical analysis, we believe that our positioning approach based on the semantics is more effective than the other two approaches. Specifically, the average scores in three tasks was higher than the other two approaches, and the standard deviation was smaller. It showed the statistical significance in the experiments.



(a) Near a Vending Machine. (b) In a Meeting Room

Figure 11: Optimizing results in other scenes. (a) The virtual agent briefs the user about how to use the vending machine. (b) The virtual agent guides the user to use a computer in a meeting room.

8.2 Regression Analysis

To investigate the validity of our cost function, we did a regression analysis based on the overall scores for the configurations of virtual agents in the user study and the corresponding cost values of the configurations. The average overall scores of 150 configurations used in the user study and the corresponding cost values of the configuration are plot in Figure 10. The green, red and purple points represent the results from the communication, teaching and guiding task, respectively.

A linear regression was calculated to predict the overall score based on the cost value. A significant regression equation was found ($F_{[2,148]} = 304.81$, $p = 9.16E-38 < .05$) with an R^2 of .673. It indicates that there is a significant negative association between the overall values and the cost values.The configurations which obtain higher overall scores in the user study have lower cost values, vice versa.

9 Summary

To interact with a virtual agent in the mixed reality naturally and realistically, it is necessary to place the virtual agent at a proper location and with an orientation, which is based on the well understanding of the semantics of the real world. We propose an approach to solve the problem of positioning a virtual agent based on the scene semantics in mixed reality. To understand the semantics of the scene, we reconstruct the 3D model of the scene and detect key objects in it. Then we devise a cost function and optimize it by an MCMC approach to obtain a proper position for the virtual agent.

There are some benefits of using our semantics-based positioning approach. First, our approach allows the user to interact with the virtual agent more naturally and realistically. Second, the cost function design and the optimization process enable the user to extend our framework conveniently. For example, the user may add specific cost to constrain the appearance of the virtual agent,e.g., changing its size or pose. In addition, our approach could be applied in many applications and scenarios (e.g., a library, a meeting room), as Figure 11 shows.

Limitations. Due to the performance limitation of the Mask R-CNN approach, we can only detect a limited number of objects (15 categories of key objects in our work). We believe that better detection performance will help to improve the semantics understanding and realism of virtual agent positioning, especially if more datasets for object detection become available. In addition, we ran the optimization process on a PC rather than on the Hololens due to the limited computing power of the Hololens. With further advancement in the computing capabilities of mixed reality devices, we may be able to run our whole approach on a mixed reality device efficiently.

Future Work. In our work, we fixed the height of the virtual agent at 30cm or 175cm, which varies according to the height of the surrounding objects. We consider to add the factor of size of the virtual agent into cost function which is more convenient to interact with people of different heights. Besides, the existing 3D object detection methods are not efficient enough, and the detection accuracy is significantly lower than that of 2D methods. If better 3D object detection methods are proposed in the future, we will consider to use them to enhance the overall process.

References

[1] H. Akiyama and I. Noda. Multi-agent positioning mechanism in the dynamic environment. In *Robot Soccer World Cup*, pages 377–384. Springer, 2007.

[2] M. Anabuki, H. Kakuta, H. Yamamoto, and H. Tamura. Welbo: An embodied conversational agent living in mixed reality space. In *CHI'00 extended abstracts on Human factors in computing systems*, pages 10–11. ACM, 2000.

[3] S. Balcisoy, M. Kallman, R. Torre, P. Fua, and D. Thalman. Interaction techniques with virtual humans in mixed environments. In *Biomedical Imaging, 2002. 5th IEEE EMBS International Summer School on*, pages 6–pp. IEEE, 2002.

[4] I. Barakonyi and D. Schmalstieg. Augmented reality agents in the development pipeline of computer entertainment. In *International Conference on Entertainment Computing*, pages 345–356. Springer, 2005.

[5] W. L. Brandão and M. S. Pinho. Using augmented reality to improve dismounted operators' situation awareness. In *IEEE VR*, pages 297–298, 2017.

[6] J. W. S. Chong, S. Ong, A. Y. Nee, and K. Youcef-Youmi. Robot programming using augmented reality: An interactive method for planning collision-free paths. *Robotics and Computer-Integrated Manufacturing*, 25(3):689–701, 2009.

[7] A. P. Dempster. Maximum likelihood estimation from incomplete data via the em algorithm (with discussion. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[8] S. Dong and T. Höllerer. Real-time re-textured geometry modeling using microsoft hololens. In *IEEE VR*, 2018.

[9] G. Elber, M. Gertelman, O. Shaked, and O. Shmueli. Object positioning and display in virtual environments, May 9 2006. US Patent 7,043,695.

[10] Y. Fadil, S. Mega, A. Horie, and K. Uehara. Mixed reality cooking support system using content-free recipe selection. In *IEEE International Symposium on Multimedia*, pages 845–850, 2006.

[11] M. Fisher, M. Savva, Y. Li, P. Hanrahan, and M. Nießner. Activity-centric scene synthesis for functional 3d scene modeling. *TOG*, 34(6):179, 2015.

[12] Q. Fu, X. Chen, X. Wang, S. Wen, B. Zhou, and H. Fu. Adaptive synthesis of indoor scenes via activity-associated object relation graphs. *ACM TOG*, 36(6):201, 2017.

[13] J. J. Gibson. The theory of affordances, in perceiving, acting and knowing, eds. re shaw and j. bransford, 1977.

[14] K. He, G. Gkioxari, P. Dollar, and R. Girshick. Mask r-cnn. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP(99):1–1, 2017.

[15] S. Henderson and S. Feiner. Exploring the benefits of augmented reality documentation for maintenance and repair. *IEEE TVCG*, 17(10):1355–1368, 2011.

[16] T. Höllerer and S. Feiner. Mobile augmented reality. *Telegeoinformatics: Location-Based Computing and Services. Taylor and Francis Books Ltd., London, UK*, 21:00533, 2004.

[17] T. Holz, A. G. Campbell, G. M. OHare, J. W. Stafford, A. Martin, and M. Dragone. Miramixed reality agents. *International journal of human-computer studies*, 69(4):251–268, 2011.

[18] T. Holz, M. Dragone, G. M. Ohare, A. Martin, and B. R. Duffy. Mixed reality agents as museum guides. In *ABSHL06: Agent-Based Systems for Human Learning, AAMAS 2006 Workshop*. ACM Press New York, NY, USA, 2006.

[19] Y. Horry, K. I. Anjyo, and K. Arai. Tour into the picture:using a spidery mesh interface to make animation from a single image. In *Conference on Computer Graphics and Interactive Techniques*, pages 225–232, 1997.

[20] T. Inoue and M. Nakanishi. Physical task learning support system visualizing a virtual teacher by mixed reality. In *CSEDU (1)*, pages 276–281. Citeseer, 2010.

[21] C. Jiang, S. Qi, Y. Zhu, S. Huang, J. Lin, L. F. Yu, D. Terzopoulos, and S. C. Zhu. Configurable 3d scene synthesis and 2d image rendering with per-pixel ground truth using stochastic grammars. 2018.

[22] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana. Virtual object manipulation on a table-top ar environment. In *ISAR 2000)*, pages 111–119. IEEE.

[23] H. Kim, J. L. Gabbard, A. M. Anon, and T. Misu. Driver behavior and performance with augmented reality pedestrian collision warning: An outdoor user study. *IEEE TVCG*, 24(4):1515–1524, 2018.

[24] V. G. Kim, S. Chaudhuri, L. Guibas, and T. Funkhouser. Shape2pose: Human-centric shape analysis. *ACM TOG*, 33(4):120, 2014.

[25] M. M. Korjani, A. Afshar, M. Norouzitallab, and M. B. Menhaj. Dynamic autonomous agent positioning based on computational intelligence. In *IAENG Conferences-IMECS*. Citeseer, 2009.

[26] R. V. Krevelen and R. Poelman. A survey of augmented reality technologies, applications and limitations. *Int.j.of Virtual Reality*, 9, 2010.

[27] K. Lee. Augmented reality in education and training. *TechTrends*, 56(2):13–21, 2012.

[28] H. Liu, Y. Zhang, W. Si, X. Xie, Y. Zhu, and S.-C. Zhu. Interactive robot knowledge patching using augmented reality. In *International Conference on Robotics and Automation (ICRA)*, pages 1947–1954, 2018.

[29] Microsoft. Holotour. `https://www.microsoft.com/en-us/p/holotour/9nblggh5pj87`.

[30] Microsoft. Young conker. `https://www.microsoft.com/en-us/p/young-conker/9nblggh5ggk1?activetab=pivot%3aoverviewtab`.

[31] K. Miyawaki and M. Sano. A virtual agent for a cooking navigation system using augmented reality. In *International Workshop on Intelligent Virtual Agents*, pages 97–103. Springer, 2008.

[32] S. Narang, A. Best, and D. Manocha. Simulating movement interactions between avatars & agents in virtual worlds using human motion constraints. In *IEEE VR*, pages 9–16, 2018.

[33] W. Narzt, G. Pomberger, A. Ferscha, D. Kolb, R. Mller, J. Wieghardt, H. Hrtner, and C. Lindinger. Augmented reality navigation systems. *Universal Access in the Information Society*, 4(3):177–187, 2006.

[34] Nintendo. Pokemon go. `https://www.pokemongo.com/en-us/`.

[35] M. Savva, A. X. Chang, P. Hanrahan, M. Fisher, and M. Nießner. Scenegrok: Inferring action maps in 3d environments. *ACM TOG*, 33(6):212, 2014.

[36] M. Savva, A. X. Chang, P. Hanrahan, M. Fisher, and M. Nießner. Pigraphs: learning interaction snapshots from observations. *ACM TOG*, 35(4):139, 2016.

[37] S. Se, D. Lowe, and J. Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *The international Journal of robotics Research*, 21(8):735–758, 2002.

[38] B. Shamir. Social distance and charisma: Theoretical notes and an exploratory study. *Leadership Quarterly*, 6(1):19–47, 1995.

[39] A. Sud, E. Andersen, S. Curtis, M. Lin, and D. Manocha. Real-time path planning for virtual agents in dynamic environments. In *IEEE VR*, pages 1–9, 2008.

[40] K. Wang, M. Savva, A. Chang, and D. Ritchie. Deep convolutional priors for indoor scene synthesis. *ACM TOG*, 37:1–14, 07 2018.