International Conference on Artificial Reality and Telexistence
Eurographics Symposium on Virtual Environments (2023)
J.-M. Normand, M. Sugimoto, and V. Sundstedt (Editors)

Ownership Estimation for Tracked Hands in a Colocated VR Environment

Dennis Reimer[1,2], Daniel Scherzer[2] and Hannes Kaufmann[1]

[1]TU Wien, Research Unit Virtual and Augmented Reality, Austria
[2]Ravensburg-Weingarten University, Germany

Abstract

Hand tracking systems play a crucial role in virtual reality (VR) applications, typically focusing on tracking the hands of the user who is using the system. Consequently, most existing systems are designed to track a maximum of two hands simultaneously. However, in certain colocated multi-user VR scenarios, it becomes necessary to track more than two hands simultaneously, such as to eliminate blind spots in individual tracking systems. In such scenarios, accurately assigning the tracked hands to the corresponding users using only the hand locations relative to the users becomes essential.
This paper introduces and evaluates various methods for efficiently assigning hands to users in such scenarios. Additionally, we propose an algorithm that leverages past assignments to enhance the robustness and effectiveness of future assignments. Our experimental results demonstrate that this algorithm significantly improves upon existing methods. Furthermore, when combined with an assignment algorithm based on reinforcement learning AI agents, we achieve a remarkable 99% accuracy in hand assignments. As a result, we present an assignment algorithm specifically tailored for colocated VR scenarios, utilizing only the hand and user locations within the scene, making it directly applicable in the aforementioned contexts.

CCS Concepts
• ***Computing methodologies*** $\rightarrow$ *Virtual reality; Mixed / augmented reality;* • ***Human-centered computing*** $\rightarrow$ *Systems and tools for interaction design;*

1. Introduction

Multi-user virtual reality (VR) experiences allow multiple users to engage in the same virtual environment (VE) simultaneously. In colocated multi-user scenarios, where users share both the VE and physical space, interactions with the environment and other users are crucial. Traditionally, these interactions are facilitated using controllers or hand tracking mechanisms. Optical markerless hand tracking presents an opportunity to achieve more natural interactions in virtual environments without additional hardware or hand-worn fabrics, eliminating the need for calibration.
While some commercial VR systems, such as Meta Quest and HTC Vive Focus, integrate hand tracking as a built-in feature, others, like LeapMotion, require an additional sensor attached to the head-mounted display (HMD). However, these systems are primarily designed for single-user scenarios and can only detect two hands at once, assuming they belong to the user. In colocated VR scenarios, there is a need to recognize and assign hands to individual users within the shared physical space. For example, it may be necessary to relay information about a user's hands even when they are not visible, such as when held behind their back.
However, a challenge arises as the recognized hands may not necessarily belong to the user from whom the tracking originated. Current research addressing this problem focuses either on image tracking [TFC20] [NNHH22] to extract additional information, such as overlapping tracking boxes or hand color, or is limited to 2-dimensional images [LBC*14]. To our knowledge, there is currently no solution available that addresses this assignment problem in a 3-dimensional virtual environment where only location information is available. This is especially difficult for hands in close proximity. Consequently, the objective of our work is to assign all virtual hands in 3D space to their respective users solely based on the location of the tracked hands relative to all present virtual users and therefore enabling targeted interactions within the virtual environment.
In this paper, we present an algorithm that computes the probability of a virtual hand belonging to a specific user. We also address the challenging task of accurately assigning hands when they overlap in the virtual scene. To achieve this, we demonstrate various methods for computing this probability and compare their effectiveness. Additionally, we introduce a history-based algorithm that utilizes previous assignments and probabilities to adjust future assignments. Finally, we evaluate our methods within a colocated virtual reality simulation, considering different user formations involving overlapping hand interactions.
Our evaluation focuses on assignment accuracy and the runtime of seven method combinations to identify the most reliable and ro-

DOI: 10.2312/egve.20231318

bust approach for assigning hands in all possible colocated virtual reality scenarios. We also assess the effectiveness of the history algorithm in improving assignment accuracy. The results reveal two methods with a correct assignment rate of 99%. One method utilized machine learning for hand assignment, while the other dynamically selected between the machine learning approach and a simpler distance algorithm. This dynamic selection not only improved the assignment runtime but also maintained a high accuracy rate. This allows for the use of computationally intensive AI methods when users are close together with overlapping hands while utilizing a faster algorithm for situations where users are farther apart. This approach ensures high accuracy with a runtime suitable for real-time applications. Furthermore, the results demonstrate the efficacy of the history algorithm, showcasing a significant increase in assignment correctness by an average of 20%, even at the expense of slightly longer runtimes.
By providing these results, we offer an algorithm for accurately assigning hands to virtual users without relying on additional image recognition methods, solely utilizing hand and user locations. Additionally, we solve the assignment problem that arises when tracked in the virtual scene are in close proximity, which is also particularly useful when a user's hands are obscured.

2. Related Work

2.1. Hand tracking

Several optical markerless hand tracking methods with diverse technological focuses are available [ZP13] [WMB*20] [SHS21]. However, these methods are confined to 2D screen space. As our approach necessitates 3D hand transformations in a virtual environment to estimate ownership, we need to explore alternative approaches.
Frati et al. used the Kinect, a Microsoft tracking camera, along with a wearable haptic device for hand tracking and rendering [FP11]. Their approach involves body tracking to assign hands to users. However, this method isn't suitable for us as our solution doesn't demand comprehensive body tracking.
The commercial solution MEgATrack, which is used in Meta Quest headsets by Han et al. [HLC*20], utilizes four monochronous cameras to track hands in a 3D space. As the system only tracks two hands at most, our solution is more appropriate for systems that are tracking more than two hands in colocated VR setups, where hand ownership estimation is needed.
Our proposed solution could integrate MediaPipe, which is offering perception pipelines for object detection in RGB cameras via machine learning [LTN*19]. Although Zhang et al.'s hand tracking within MediaPipe [ZBV*20] offers accurate 3D joint recognition for visible hands, it primarily emphasizes hand recognition rather than associating these hands with virtual users.
Other solutions include those that use RGB input [ZB17, POA17, SHS21] or cameras with additional depth input [SKR*15, HZG*21, MEN*18]. For example, Malik et al. use segmentation masks and mesh files of depth maps for neural network recognition with a processing time from depth image to mesh of 3.7ms and joint location error <15mm [MEN*18]. Zimmermann et al. published a method of 3D hand pose estimation based on RGB input, which was tested on a synthetic dataset for neural network hand recognition, with performance comparable to existing depth approaches [ZB17].

2.2. Hand interactions in multiuser scenarios

Prior research illustrates hand tracking applications in multiuser scenarios, emphasizing the importance of correctly assigning hands to users during interactions. Dohse et al. for example created a tabletop that uses hand tracking to enable multiuser hand interactions [DDSP08]. Something similar was investigated by Del Bimbo et al. They created a computer vision-based system and algorithms which uses hand recognition to enable multiuser interactions at a display table [DBLV06].
Multiuser VR scenarios have also been the target of previous work to investigate virtual interactions. User testing regarding the impact of interactions in colocated multiuser scenarios for cultural heritage was conducted by Li et al. They concluded that social influence has positive effects on performance expectancy and effort expectancy [LCCS18]. Streuber et al. conducted user tests in multiuser scenarios and found that lack of haptic and tactile feedback can be compensated if the user is immersed in the virtual environment [SC07]. A case study for interaction systems in multiuser VR was conducted by Gong et al. [GSB*20]. They underscore the significance of a robust interaction system in multiuser scenarios. This highlights the crucial role of accurate hand assignment in revealing which user is engaged in the interaction process. This is crucial, particularly in scenarios where recognized hands need to work closely together, possibly with only one hand recognition system available, such as in surgical settings.

2.3. Hand ownership in 2D and egocentric views

To assign detected hands to users, various existing research studies have explored hand ownership in egocentric views and hands detected in 2D space. Narasimhaswamy et al. employed neural networks to associate hands with bodies by utilizing overlapping tracking bounding boxes from the image detection, as well as the locations of heads and hands, to perform hand assignments [NNHH22]. They show that hand contact estimation can be improved with a hand-body association.
Tsutsui et al. also utilized visual cues provided by image tracking, such as color, depth, skin texture, and shape, for hand identification in egocentric views [TFC20]. However, their approach only focused on identifying the user's own hands regardless of their position, without considering other users and hands in the environment. Additionally, their approach demonstrates a verification error rate of 36%, making it impractical for real-world scenarios. Hence, this approach is not applicable to our scenario.
Tabletop solutions that enable multi-user interaction also require the correct assignment of interactions to the respective users. Del Bimbo et al. focused on distinguishing between two overlapping hands using a predictor-based method [DBLV06]. Conversely, Dohse et al. employed a combination of computer vision for hand recognition and touch detection to differentiate between hands in close proximity [DDSP08]. Both works highlighted the significance of distinguishing and assigning hands to different users for interactions in multi-user and close proximity scenarios. However,

as our focus is on assignment based solely on the location of users and their hands in the virtual environment, these approaches are not directly applicable to our solution.
Lee et al. distinguished multiple hands in an egocentric view using 2D hand locations, although their method didn't encompass tracked hands in the 3-dimensional world [LBC*14]. Lin et al. introduced a technique detecting hand raising in classrooms with a 90% mean Average Precision [LJS18]. Zhou et al. expanded on this by improving the detection and pose algorithms, achieving an average recognition precision of 83% by using hand location and keypoints for hand and hand raiser matching [ZJS18]. Since all the mentioned research either confines hand assignment to the 2-dimensional space [LBC*14] [LJS18] [ZJS18], or relies on additional tracking information such as color or 2D tracking boundaries [NNHH22] [TFC20], they are not applicable to our use case. Our scenario involves assigning hands in a colocated virtual environment in 3D space, with limited information available, specifically the location of the hands and user heads. Therefore, our methods are based on the existing methods in the 2D domain and extend them with our own ideas for the 3D domain.

3. Methodology

Our algorithm prioritizes robust hand-to-user assignments, addressing challenges in colocated multiuser scenarios. Traditional hand-tracking systems often assume tracking only the primary user's hands due to their two-hand limitation [HLC*20] [WMB*20] [SHS21]. However, in colocated scenarios, each user's hands must be assigned to their respective tracking systems.
A naive approach is to assign each virtual hand to its closest user, but this is error-prone in close user formations with spatially intermixed virtual hands. Our algorithm evaluates various factors, such as distance and rotation, to compute assignment probabilities for virtual hands.
Additionally, we maintain a history cache of prior hand-user assignments to enhance future accuracy. Leveraging this historical data, our algorithm predicts virtual hand ownership more reliably, especially in dynamic colocated settings.

3.1. Methods

We developed four approaches to determine the affiliation of a hand to a user, each of which computes a probability factor p.

Distance: We calculate the distance of the hand to the user's head and assume that the closer the hand is, the more likely it is to belong to the user. If the distance exceeds a threshold value, we no longer assume that the hand belongs to the user. To calculate probability values between 0 and 1, we use a logistic curve, as shown in Equation 1. We chose a distance range of [0.4m;1.3m] resulting in $a = 10.22$ and $b = 0.85$. Figure 1 shows the resulting logistic curve. This method was selected based on its simplicity and anticipated high accuracy in numerous colocated scenarios, where users are sufficiently separated from each other.

$$p = 1 - \frac{1}{1 + e^{-a(x-b)}} \tag{1}$$

Rotation: We calculate the angle between the hand and the direction vector between the user's head and the hand. We assume that a hand turned away from the user is more likely to belong to the user than a hand turned towards the user (which would require an uncomfortable hand position). We calculate p using the logistic curve in Equation 1 with values in the angle range [90°;180°], which results in the values $a = 0.1$ and $b = 135$. The logistic curve can be seen in Figure 1.

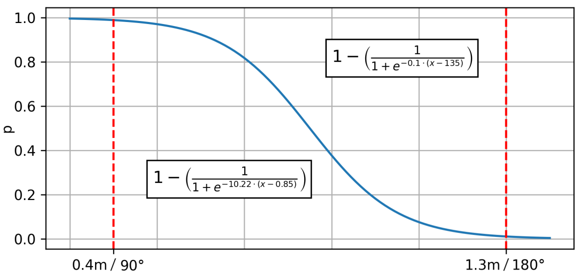

Figure 1: *Logistic curve for distance (formula at bottom left)) and rotation (formula at bottom right) calculations. The X-axis represents the input position/rotation, while the Y-axis denotes the resulting probability. The red lines indicate the selected threshold values.*

Prerecorded area: In a preliminary setup, we asked the user to stretch out their arms and perform defined movements in front, to the side, and behind their body, resulting in a total of 459 points. A convex hull is then created from the points at runtime, in which it is checked whether the user's hand is in this hull or not. Our focus was on capturing the essential movements rather than the precise number of points to ensure comprehensive coverage of all relevant sides of the body. Such a convex hull can be seen in the supplemental material. The fundamental idea is that although this approach is more complex than using the distance method, it offers a more accurate representation of the hand's radius. For instance, it acknowledges the limitation that hands cannot be positioned as far behind the user as they can be in front of the user.

Machine Learning: In this approach, we used Unity3D's machine learning agents [JBT*20, CTB*22] for hand assignment using reinforcement learning. During training, an agent received input data, including hand and user locations, and received rewards for correct assignments. To enrich the learning experience, we introduced random user placements within a 2-meter radius, with locations changing every frame. Additionally, we enforced specific scenarios with overlapping hands when users were close to train the agent for challenges. For varying user counts, we created dedicated agents (e.g., one for two users, one for three users) to handle increased decision complexity. Training involved 1.5 million steps, and reward progression can be found in the supplemental material, showing smoothed reward curves.
This training data served as training input for Unity's ML system, enabling the creation of an agent capable of runtime hand assignment using the same type of input data (positions, rotations). In the assignment process, each visible hand in the scene is assigned to a specific user using its own agent, depending on the number of concurrent users. Agents were created for 2-5 concurrent users, but

more concurrent user agents can be easily added. A decision for hand assignment is made in every render frame. We anticipate that this method can leverage multiple inputs during the learning process to autonomously make real-time decisions. Consequently, we expect it to offer broader coverage of various colocated situations.

Dynamic: This method combines the other methods to dynamically determine which method to use in the current situation. The idea is that if all users to be tested are far apart, the distance methods can give a very reliable indication of the hand's affiliation. However, if at least two users are spatially very close to each other, the assignment is no longer unambiguous, and therefore a method is used which is more reliable when hands are overlapping. In our test implementations, we use the combination of "Prerecorded area" when far away and "Distance" when near, as well as the combination of "Distance" when far away and "Machine Learning" when near. We expect the strengths of both methods used to come into their own depending on the distance. Especially the second combination is expected to improve the runtime compared to the singular use of the complex algorithm while maintaining a high correctness. As threshold distance, we preliminarily determined a near distance of 0.8m as effective.

3.2. History Algorithm

In a naive mapping of a hand to a user, the probability that a hand belongs to a particular user is computed independently for each hand and user in isolation. However, the prior assignment of a hand could significantly influence future assignments. To tackle this problem, we're exploring if integrating historical data can enhance hand assignments. The concept is that if a hand was previously assigned securely to a user, it is probable that it will belong to that user in subsequent instances. The idea is similar to the basic idea of dead reckoning, where for example in games previous player states are used to predict next states to counteract network latency [Mur11].
In our implementation, we cache the calculated probabilities of each hand for each user for each frame. We limit the number of entries to 500 for memory and runtime efficiency, with new entries replacing the oldest ones. From all cached entries, we compute the mean probability value and use it for the final probability calculations. However, we only incorporate the history information if the value of the history exceeds a certain threshold. Pilot tests indicate that a threshold of $p >= 0.9$ is appropriate.
Furthermore, we compute a weighting for the history so that high prior probabilities are weighted more heavily in the overall computation. As a result, the history probability can be multiplied by a weighting factor, depending on the previous probabilities. A maximum wight of 2 was chosen to expedite that the history algorithm's return to lower probabilities after consecutive frames with low input raw probabilities, a weighting factor of 2 was selected. The probability p of the history is calculated using the following formula:

$$p_h = \frac{\sum_{n=0}^{N} p_n}{N} * \frac{\sum_{n=0}^{N} p_n * w_f}{N_{max}} = \frac{w_f * (\sum_{n=0}^{N} p_n)^2}{N * N_{max}} \qquad (2)$$

where:

N	=	number of entries in the history
N_{max}	=	maximum number of entries in the history, set to 500 in our implementation
w_f	=	maximum weight, set to 2 in our implementation

An example visualization, in which different raw probabilities are given, can be found in Figure 2.

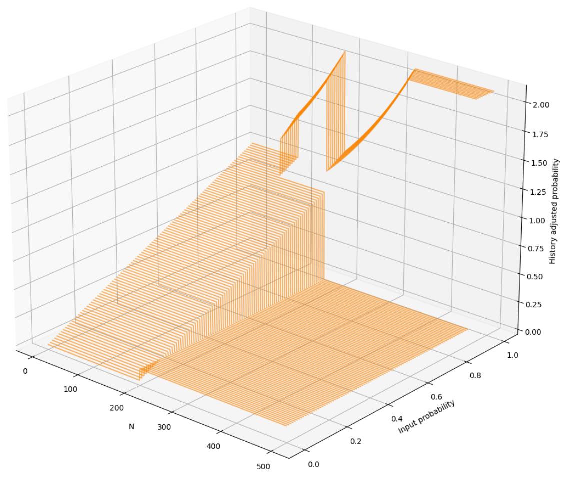

Figure 2: *Visualization of the history algorithm from Equation 2. The input probability is represented within the range of [0:1]. The history adjustment is applied once the number of entries reaches N=100. As the number of entries increases to N=200, the input probability decreases to 0.1, indicating that probabilities in the history above the threshold of 0.9 remain high, as they had a consistently high probability in the past for accurate assignments.*

Therefore, a previous reliable assignment of a hand can also be critical for future calculations if an assignment is no longer straightforward. For instance, in dynamic scenarios where users start far apart but later converge to interact with the same virtual object, relying solely on distance for hand assignment becomes less effective. In such situations, utilizing historical data can assist in making more accurate hand assignments. This underscores the value of incorporating historical information in dynamic contexts.

4. Evaluation

To assess the effectiveness of various hand ownership estimation methods and examine the impact of the history algorithm, we developed an evaluation framework that simulates diverse test scenarios within a colocated VR environment. Each scenario comprises specific user formations and the choice of user movement or stationary positioning. The objective of this evaluation is to offer comparable scenarios for different combinations and conduct a quantitative analysis of the accuracy and runtime performance of the distinct methods.
We created a simulated environment that maps a colocated VR application, with 2-5 simulated users as needed. For each user, hand movements were recorded over a period of 1500 consecutive frames, representing an activity such as assembly work. These

recordings can then be played back for different test scenarios, making these scenarios similar in their hand movements and thus comparable. Each simulated user had different recordings to cover different tracking scenarios while maintaining repeatability and comparability.
Each test scenario is defined by the following factors:

- **Formation:** A predefined formation of how the virtual users are positioned with respect to each other in the virtual space and at different distances.
- **Method:** An assignment method, as described in subsection 3.1.
- **History:** Whether history is used as in subsection 3.2.

Specifically, we use the methods described in subsection 3.1 in isolation (Distance **(D)**, Rotation **(R)**, Prerecorded Area **(PA)**, and Machine Learning **(ML)**). In addition, we use a combination of Prerecorded Area and Distance **(D-PA-C)** to see if this combination improves the individual results. The selection of the methods was made in a preliminary evaluation run. Two constellations were used for the dynamic method: Area Prerecorded as a base method with Distance as an additional method when users are close to each other **(PA-D-DYN)**; and Distance as a base method and Machine Learning when users are close to each other **(D-ML-DYN)**. The idea is to use a simple algorithm for straightforward scenarios and switch to a more complex, potentially slower one when necessary, such as when users are close or hands overlap. This approach aims to improve average runtime while maintaining good results. Therefore, a total of 7 method constellations were evaluated.
In the simulation, we assume all virtual hands are detected by a hand detection system but remain unassigned to users. With ground truth information about hand ownership, we can compare assignment results. We calculate a **correctness ratio**, representing the percentage of accurate assignments during evaluation. A ratio of 1 signifies perfect assignment accuracy, serving as a key metric for method comparison in our study.
In addition to assessing assignment accuracy, we evaluate the runtime (in ms) of each method to determine its applicability in a real-time scenario and the computational time required. In our implementation, we included a method to measure the execution time of the assignment process. This system tracks the time taken for execution by capturing the start and end times. To find the overall runtime per frame, we sum up the measured times for all frames and divide by the total frames collected. Anticipating increased runtime with more users, we calculate an adjusted runtime per user by dividing the frame's runtime by the number of concurrent users, as elaborated in subsection 5.3.
The evaluation covers a total of 73 formations, comprising 14 formations with 2 users, 16 with 3 users, 20 with 4 users, and 23 with 5 users. The variation in the number of formations is attributed to the increasing number of concurrent users, which in turn leads to a greater diversity of potential arrangements. We aimed to represent realistic formations where users stood both farther apart and close together, resulting in overlapping hands. Additionally, we selected formations in which users remained fixed in one place and dynamic formations where users moved along a fixed path. Each of the seven methods mentioned earlier was applied to each formation for the duration of 1500 frames to determine the percentage of correctly assigned hands. This evaluation was conducted both with and without applying the history algorithm. In total, we obtained 1022 results for comparative analysis.
The evaluation was performed on a Windows notebook equipped with an Intel Core i7-12700H CPU and an NVIDIA GeForce RTX 3070 Ti Laptop GPU. We utilized Unity3D 2021.3.9 as the software platform, coupled with an in-house developed hand tracking framework for Unity. This framework served as a layer for hand tracking, and was responsible for recording tracked hands in a preliminary setup, playing back these recordings, and rendering the hands during the evaluation.
By conducting these evaluations, we gain insights into the accuracy of hand assignment and the computational efficiency of the methods, ensuring their feasibility in real-time applications.

5. Results

We conducted an analysis of various metrics to determine the optimal method for a real colocated VR scenario. Firstly, we evaluated the effectiveness of our history algorithm in improving hand assignment results. Additionally, we identified the method or combination of methods that yielded the best outcomes. To evaluate the algorithm and methods, we measured the accuracy of hand assignment for each frame, enabling us to calculate a correctness ratio for each combination of method, formation, and utilization of the history algorithm. This resulted in a total of 1022 values, each representing the percentage of correctly assigned hands throughout the evaluation.
The third metric we examined was the runtime of the algorithm. As our objective is to use this estimation method in real-time colocated VR applications, runtime is a critical factor in determining the method's feasibility. The runtime was measured in milliseconds per frame for each run.
Based on our implementation and the complexity of the methods, we formulated the following hypotheses for our expected results:

- **H1**: The history algorithm significantly improves hand assignments.
- **H2**: When used individually, the machine learning algorithm achieves the highest correctness ratio.
- **H3**: The machine learning algorithm has the longest runtime.
- **H4**: A dynamic combination of two methods can yield results with a correctness rate comparable to that of machine learning, while also achieving a reduced runtime.

By analyzing these metrics and testing our hypotheses, we aim to provide valuable insights into the performance and suitability of different methods for real-time colocated VR scenarios. To select the appropriate test for the comparative analysis in the subsequent sections, we performed a Shapiro-Wilk normality test [SW65] on each analysis. The test results suggested that the examined data did not adhere to a normal distribution ($p < 0.001$). We presume that the non-normal distribution might be attributed to the varying complexity of different methods, leading to distinct runtimes and diverse accuracy distributions across different scenarios.

5.1. History analysis

To evaluate the impact of the history algorithm, we analyzed all available test results, resulting in a sample size of 1022 entries. Half of the entries (n=511) were executed with the history algorithm enabled, while the other half was run without it. The objective was to determine if the utilization of the history algorithm leads to a significant improvement in hand assignment rates. The results are visualized in Figure 3.

Without employing the history algorithm, an average correctness rate of **0.658** ($\sigma = 0.26$) was observed across all data. However, when the history algorithm was included, the rate increased to **0.793** ($\sigma = 0.29$). We conducted an Independent-Samples Mann-Whitney U test [Nac08] for comparative analysis. The resulting p-value (p<0.001; U=79317), together with a rank-biserial correlation coefficient of *r=0.393* (indicating a positive effect size), demonstrate significant differences with a notable improvement in hand assignment correctness rates when employing the history algorithm. These findings confirm **H1**, as the utilization of our history algorithm led to significantly enhanced assignment results, showcasing an average improvement of **20%** in our evaluations.

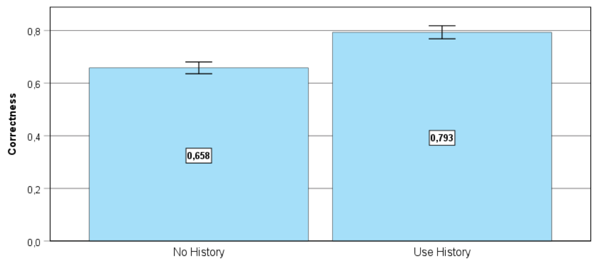

Figure 3: *Accumulated correctness across all methods with and without the history algorithm. The utilization of the history algorithm leads to an overall higher level of correctness.*

5.2. Method comparison

In this analysis, the methods listed in section 4 were compared based on their correctness ratio. Since the previous section demonstrated the superior performance of the history algorithm, we focused on runs where the history algorithm was applied to achieve the highest possible correctness in our evaluations. This resulted in a sample size of 511 cases, corresponding to N=73 per method with 7 applied methods.

Initially, we calculated the means and standard deviations (σ) of the correctness ratio for all methods, which can be found in detail in the supplemental material. From the table, it is evident that the methods that include machine learning exhibit the highest correctness ratio with the lowest standard deviation. Conversely, the rotation method yielded the lowest average correctness, with approximately half of the assigned hands being incorrect on average. The corresponding bar graph can be seen in Figure 4.

We conducted an Independent-Samples Kruskal-Wallis test [KW52] to assess the differences in correctness among all methods. The analysis yielded a significant result (p<0.001; H=92.9; df=6) and a medium effect size, as measured by Eta-squared ($\eta^2 = 0.172$), indicating substantial differences in the distribution of correctness across the methods.

To further investigate these differences, we performed a post-hoc test for pairwise comparisons between the methods. We applied

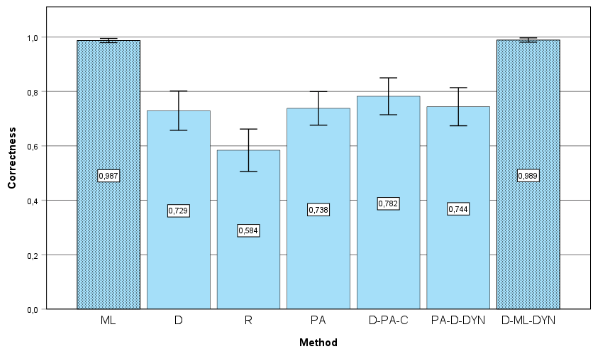

Figure 4: *Mean correctness results for the methods. The two most accurate methods are highlighted.*

Bonferroni correction (with k=21) to adjust the resulting p-values. The detailed p-values can be found in the supplemental material.

The rotation method exhibited significantly different results compared to all other methods, indicating it has the lowest correctness ratio. The machine learning method demonstrated significant differences when compared to the distance and rotation methods. Although significance was initially observed for the dynamic methods before Bonferroni correction, it was no longer present after adjusting for multiple comparisons. A larger sample size might help reduce this variance. However, it is worth noting that the machine learning method exhibited the least variance and the most consistent correctness across all runs, as evidenced by its low standard deviation and consistent data distribution.

The dynamic combination of distance and machine learning shows an even more favorable outcome, showing significant differences compared to all other methods except for the standalone Machine Learning method. It also displayed high consistency with a low standard deviation.

With a correctness ratio of **99%**, both the machine learning algorithm and the dynamic combination of machine learning and distance emerged as significantly superior algorithms for hand assignments. Therefore, these findings partially support Hypothesis **H2**, with the exception that the dynamic combination yielded similar results.

The remaining methods fell within a similar range of average correctness, ranging from **72%** to **78%**, and did not produce significant differences from each other. This suggests that the choice of method in this range should be based on other factors, such as runtime. However, given that machine learning and the dynamic combination of distance and machine learning achieved higher average correctness with significant differences compared to other methods, they are preferred over the remaining methods if runtime is acceptable (more details on runtime results are provided in the next chapter).

5.3. Runtime

In our final decision regarding the optimal hand assignment algorithm, it is crucial to consider the runtime to assess the method's feasibility in a real-time scenario. Since we have already estab-

lished in subsection 5.1 that the history algorithm significantly improves hand assignments, we also need to investigate if it introduces a substantial increase in runtime. In addition to correctness, runtime plays a vital role in selecting the best method.
To obtain the final runtime for the most effective hand assignment method, we calculated and compared the runtime for each method when the history algorithm was employed. As the allocation methods are applied to active hands and users in each frame, the computational load increases with more users and hands. To account for this, we used a user-adjusted runtime per computation frame for the analysis by dividing the measured frame runtime by the number of users.
The findings in the runtime analysis will aid in selecting the most suitable algorithm for hand assignment, taking both correctness and runtime into consideration.

5.3.0.1. History runtime To assess the impact of the history algorithm on runtime, we evaluated the runtime across all available runs, resulting in a sample size of 1022, evenly split between runs with and without the history algorithm.
When utilizing the history algorithm, we observed an average user-adjusted runtime of **0.465 ms** ($\sigma = 0.22$). In contrast, the average user-adjusted runtime without the history algorithm was **0.147 ms** ($\sigma = 0.24$). It is important to note that these mean values do not solely represent the runtime of the algorithm, as they incorporate the values from different assignment methods. However, they do indicate that the history algorithm generally results in a higher runtime.
We conducted an independent-samples Mann-Whitney U test and calculated the rank-biserial correlation coefficient to determine the effect size. The test yielded a significant result ($p<0.001$; U=40603), and the coefficient (*r=0.689*) indicated a positive effect size, demonstrating significant differences between using and not using the history algorithm.

5.3.0.2. Method runtime Considering the significant improvement in the correctness ratio achieved by the methods, we proceeded to analyze the runtime of each method when combined with the history algorithm. The mean runtime values per number of users, along with the user-corrected value, are visually depicted in Figure 5. Detailed values can be found in the supplemental material.
Again, we conducted an Independent-Samples Kruskal-Wallis test. The analysis revealed significant differences among the user-corrected runtimes of the methods, with a p-value of less than 0.001 (H=249.74; df=6) and a positive effect size (calculated using Eta-squared) of $\eta^2 = 0.484$. To determine the specific differences, post-hoc pairwise comparisons were performed with Bonferroni correction (with k=21).
The results indicate significant differences between the Machine Learning (ML) method and all other methods, as well as between the Dynamic Method with Machine Learning and Distance (D-ML-DYN) and all other methods, each with a p-value of less than 0.001. As both of these methods involve reinforcement learning agents, which demand more CPU time, it was anticipated that they would exhibit longer runtimes. These findings confirm our hypothesis **H3**. However, it is worth noting that there was also a significant difference between the two methods that utilize Machine Learning

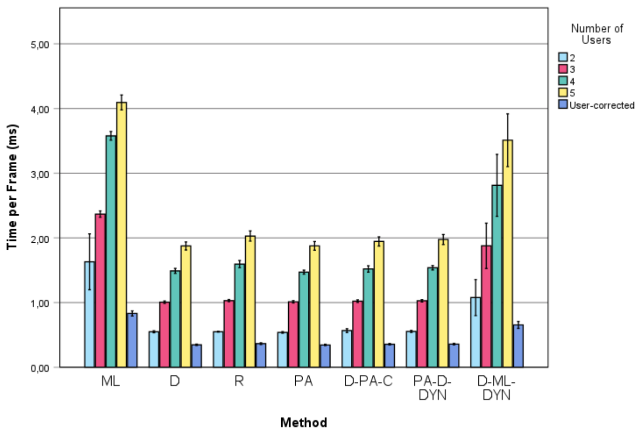

Figure 5: *Mean runtime results for the methods at various user counts. One bar illustrates the mean runtime adjusted for the number of users.*

(p=0.047). This indicates that the dynamic combination, with a mean runtime of 0.65ms, is significantly faster than the Machine Learning method alone, which has an average runtime of **0.83ms**, by **27.7%**. When considering the results from subsection 5.2, hypothesis **H4** can also be confirmed. The dynamic combination of the machine learning and distance algorithms exhibits a significantly better runtime while achieving the best correctness results among all the applied methods.
No significant differences were observed among the remaining methods that do not utilize machine learning agents, suggesting similar runtimes for these methods. With an average runtime of approximately **0.35ms**, these methods are **85.7%** faster than the dynamic method with machine learning and **137.1%** faster than the isolated machine learning method. A more comprehensive discussion on the applicability of the methods based on these results, along with the previous findings, is presented in section 6.

6. Discussion

The results demonstrate significant differences in the effectiveness of different algorithms for assigning recognized hands to users in colocated VR scenarios. As anticipated, the rotation method performs the worst, with approximately half of the virtual hands being incorrectly assigned. This outcome is logical since hands typically have a specific orientation to their owners but can be freely rotated. Assigning hands becomes challenging when two users have similar orientations in the scene, resulting in ambiguous assignments. Although a combination of the rotation method with other methods could potentially yield improvements, initial tests did not reveal any visible enhancement over other methods. Furthermore, since the rotation method does not offer significantly better runtime, it is not suitable for reliable hand assignment.
The distance-based methods, whether utilizing the distance or a prerecorded area, deliver similar results across all examined aspects. With an accuracy rate of approximately three out of four hands correctly assigned, the performance is better than the rotation method but still insufficient for ensuring reliable assignment. Con-

trary to our assumptions, the precise definition of the hand's area of action (prerecorded area) does not yield significantly better results compared to the pure distance method, which essentially represents a radius around the user. Additionally, combining these two methods did not confirm the notion that they could mutually support each other. Given the similar correctness rates and runtimes, the distance algorithm is preferable to the prerecorded area algorithm in terms of simplicity.
Taking into account Figure 6, it becomes evident again why the distance methods do not exhibit high overall correctness rates. When examining user information where users are far apart, the assignment of hands can be achieved with a high probability of success. However, if users are standing close to each other and their hands overlap, assigning hands based solely on distance becomes significantly more challenging. Consequently, the distance methods are more suitable for scenarios where users in colocated spaces are not in close proximity to each other.

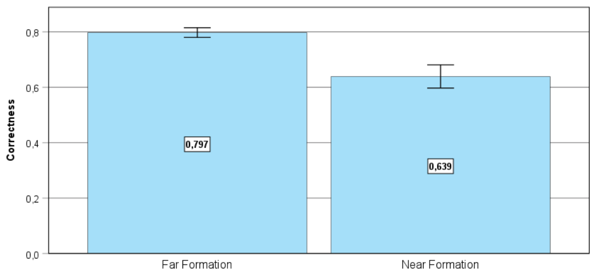

Figure 6: *Effect of user proximity on the accuracy for the distance method. Reduced correctness and increased variance can be seen when users are near each other.*

When examining the methods involving machine learning, the results demonstrate a significantly improved assignment rate. The prior reinforcement learning of an AI agent was crucial in effectively determining the assignment factors. However, the one-time learning effort is no longer necessary in subsequent applications, highlighting its capability to produce good results. With a correctness rate of approximately 99%, these machine learning methods exhibit the highest accuracy in hand assignment, regardless of the proximity of users to each other. This distinguishes them clearly from the distance methods and scores an even higher accuracy than existing algorithms [LJS18].
Nonetheless, as expected, the machine learning methods also have the highest runtime due to the computational complexity involved, with a user-corrected runtime ranging from 0.65 to 0.83 ms. The runtime increases linearly with the number of users and hands present in the scene. Its applicability is thus contingent on the number of users, as well as the complexity and rendering demands of the virtual VR scene. Nevertheless, the runtime remains sufficiently low to enable real-time applications.
The dynamic combination of machine learning and distance algorithms proves to be the optimal approach among the presented methods. The results indicate that the weaknesses of the distance method in assigning hands to users standing close to each other can be effectively compensated for by the machine learning agent. As a result, a significantly improved runtime (of 21%) can be achieved while maintaining a high correctness rate. This combination is particularly suitable for complex applications with high computational complexity. When users are consistently in close proximity to each other, the worst-case runtime is comparable to that of using machine learning alone. Based on these findings, this combination is recommended for real-time applications.
Finally, the results highlight the significant improvement in the correctness rate of hand assignments achieved by the history algorithm we developed. As expected, applying the algorithm also leads to higher runtime. However, the runtime remains at a reasonable level even with the algorithm, ensuring its applicability. The 20% improvement in assignment correctness, coupled with an acceptable increase in runtime, establishes the algorithm as a substantial enhancement to the methods. Consequently, we incorporate it into our methods for optimal results.

7. Conclusion

This research aimed to present and compare algorithms for assigning tracked hands to virtual users in a colocated virtual environment, using limited information from the tracking system. Unlike other solutions that rely on image detection information [TFC20] [LBC*14] [LJS18] [ZJS18], we focused on utilizing the hand location in the virtual scene. We introduced and evaluated various methods for determining hand assignments, including a history algorithm that enhances assignment robustness by leveraging past assignment information.
Our evaluation revealed that assignment methods employing prelearned AI agents achieved the highest accuracy with the lowest variance. Specifically, the dynamic combination of an AI agent for close user proximity and overlapping hands, combined with a simple distance calculation for greater distances, improved the runtime while maintaining a high correctness rate of 99%. This dynamic combination effectively addressed the challenge of assigning overlapping hands. Although other methods exhibited better general runtimes, their accuracy fell short of being suitable for real colocated VR scenarios. In scenarios involving hand tracking systems for tracking multiple hands, such as those utilizing camera tracking meshes, our solution provides developers with a high-accuracy and real-time hand assignment tool.
The computational load in a colocated VR environment increases with each additional user, which leads us to anticipate that this approach will eventually be constrained by the computational power of the system, ultimately limiting the maximum number of concurrent users before the application becomes impractical. Future applications should consider investigating the limitations concerning the maximum number of users and hands that can be simultaneously present in the virtual scene. Furthermore, it would be worthwhile to explore potential enhancements to the algorithm's runtime. For instance, enhancing the performance of hand assignments can be achieved by optimizing the algorithm's execution frequency. This can be accomplished by avoiding the execution of the algorithm during every frame. One avenue for improvement is to investigate the integration of image detection cues, such as hand color, shape, and classification, to augment both accuracy and runtime in hand assignment. Such explorations have the potential to significantly advance the effectiveness and efficiency of hand ownership estimation techniques.

References

[CTB*22] COHEN A., TENG E., BERGES V.-P., DONG R.-P., HENRY H., MATTAR M., ZOOK A., GANGULY S.: On the use and misuse of abosrbing states in multi-agent reinforcement learning. *RL in Games Workshop AAAI 2022* (2022). 3

[DBLV06] DEL BIMBO A., LANDUCCI L., VALLI A.: Multi-user natural interaction system based on real-time hand tracking and gesture recognition. In *18th International Conference on Pattern Recognition (ICPR'06)* (2006), vol. 3, pp. 55–58. doi:10.1109/ICPR.2006.833. 2

[DDSP08] DOHSE K. C., DOHSE T., STILL J. D., PARKHURST D. J.: Enhancing multi-user interaction with multi-touch tabletop displays using hand tracking. In *First International Conference on Advances in Computer-Human Interaction* (2008), pp. 297–302. doi:10.1109/ACHI.2008.11. 2

[FP11] FRATI V., PRATTICHIZZO D.: Using kinect for hand tracking and rendering in wearable haptics. In *2011 IEEE World Haptics Conference* (2011), pp. 317–321. doi:10.1109/WHC.2011.5945505. 2

[GSB*20] GONG L., SÖDERLUND H., BOGOJEVIC L., CHEN X., BERCE A., ÅSA FAST-BERGLUND, JOHANSSON B.: Interaction design for multi-user virtual reality systems: An automotive case study. *Procedia CIRP 93* (2020), 1259–1264. 53rd CIRP Conference on Manufacturing Systems 2020. URL: https://www.sciencedirect.com/science/article/pii/S2212827120305965, doi:https://doi.org/10.1016/j.procir.2020.04.036. 2

[HLC*20] HAN S., LIU B., CABEZAS R., TWIGG C. D., ZHANG P., PETKAU J., YU T.-H., TAI C.-J., AKBAY M., WANG Z., NITZAN A., DONG G., YE Y., TAO L., WAN C., WANG R.: Megatrack: Monochrome egocentric articulated hand-tracking for virtual reality. *ACM Trans. Graph. 39*, 4 (aug 2020). URL: https://doi.org/10.1145/3386569.3392452, doi:10.1145/3386569.3392452. 2, 3

[HZG*21] HUANG L., ZHANG B., GUO Z., XIAO Y., CAO Z., YUAN J.: Survey on depth and rgb image-based 3d hand shape and pose estimation. *Virtual Reality & Intelligent Hardware 3*, 3 (2021), 207–234. URL: https://www.sciencedirect.com/science/article/pii/S2096579621000280, doi:https://doi.org/10.1016/j.vrih.2021.05.002. 2

[JBT*20] JULIANI A., BERGES V.-P., TENG E., COHEN A., HARPER J., ELION C., GOY C., GAO Y., HENRY H., MATTAR M., LANGE D.: Unity: A general platform for intelligent agents. *arXiv preprint arXiv:1809.02627* (2020). 3

[KW52] KRUSKAL W. H., WALLIS W. A.: Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association 47*, 260 (1952), 583–621. URL: http://www.jstor.org/stable/2280779. 6

[LBC*14] LEE S., BAMBACH S., CRANDALL D. J., FRANCHAK J. M., YU C.: This hand is my hand: A probabilistic approach to hand disambiguation in egocentric video. In *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops* (2014), pp. 557–564. doi:10.1109/CVPRW.2014.86. 1, 3, 8

[LCCS18] LI Y., CH'NG E., CAI S., SEE S.: Multiuser interaction with hybrid vr and ar for cultural heritage objects. In *2018 3rd Digital Heritage International Congress (DigitalHERITAGE) held jointly with 2018 24th International Conference on Virtual Systems & Multimedia (VSMM 2018)* (2018), pp. 1–8. doi:10.1109/DigitalHeritage.2018.8810126. 2

[LJS18] LIN J., JIANG F., SHEN R.: Hand-raising gesture detection in real classroom. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2018), pp. 6453–6457. doi:10.1109/ICASSP.2018.8461733. 3, 8

[LTN*19] LUGARESI C., TANG J., NASH H., MCCLANAHAN C., UBOWEJA E., HAYS M., ZHANG F., CHANG C., YONG M. G., LEE J., CHANG W., HUA W., GEORG M., GRUNDMANN M.: Mediapipe: A framework for building perception pipelines. *CoRR abs/1906.08172* (2019). URL: http://arxiv.org/abs/1906.08172, arXiv:1906.08172. 2

[MEN*18] MALIK J., ELHAYEK A., NUNNARI F., VARANASI K., TAMADDON K., HÉLOIR A., STRICKER D.: Deephps: End-to-end estimation of 3d hand pose and shape by learning from synthetic depth. *CoRR abs/1808.09208* (2018). URL: http://arxiv.org/abs/1808.09208, arXiv:1808.09208. 2

[Mur11] MURPHY C.: *Believable Dead Reckoning for Networked Games*. 02 2011, pp. 307–328. doi:10.1201/b11333-21. 4

[Nac08] NACHAR N.: The mann-whitney u: A test for assessing whether two independent samples come from the same distribution. *Tutorials in Quantitative Methods for Psychology 4* (03 2008). doi:10.20982/tqmp.04.1.p013. 6

[NNHH22] NARASIMHASWAMY S., NGUYEN T., HUANG M., HOAI M.: Whose hands are these? hand detection and hand-body association in the wild. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022), pp. 4879–4889. doi:10.1109/CVPR52688.2022.00484. 1, 2, 3

[POA17] PANTELERIS P., OIKONOMIDIS I., ARGYROS A. A.: Using a single RGB frame for real time 3d hand pose estimation in the wild. *CoRR abs/1712.03866* (2017). URL: http://arxiv.org/abs/1712.03866, arXiv:1712.03866. 2

[SC07] STREUBER S., CHATZIASTROS A.: Human interaction in multi-user virtual reality. *Proceedings of the 10th International Conference on Humans and Computers (HC 2007), 1-7 (2007)* (01 2007). 2

[SHS21] SUN Z., HU Y., SHEN X.: Two-hand pose estimation from the non-cropped rgb image with self-attention based network. In *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)* (Los Alamitos, CA, USA, oct 2021), IEEE Computer Society, pp. 248–255. URL: https://doi.ieeecomputersociety.org/10.1109/ISMAR52148.2021.00040, doi:10.1109/ISMAR52148.2021.00040. 2, 3

[SKR*15] SHARP T., KESKIN C., ROBERTSON D., TAYLOR J., SHOTTON J., KIM D., RHEMANN C., LEICHTER I., VINNIKOV A., WEI Y., FREEDMAN D., KOHLI P., KRUPKA E., FITZGIBBON A., IZADI S.: Accurate, robust, and flexible real-time hand tracking. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (New York, NY, USA, 2015), CHI '15, Association for Computing Machinery, p. 3633–3642. doi:10.1145/2702123.2702179. 2

[SW65] SHAPIRO S. S., WILK M. B.: An analysis of variance test for normality (complete samples)†. *Biometrika 52*, 3-4 (12 1965), 591–611. URL: https://doi.org/10.1093/biomet/52.3-4.591, arXiv:https://academic.oup.com/biomet/article-pdf/52/3-4/591/962907/52-3-4-591.pdf, doi:10.1093/biomet/52.3-4.591. 5

[TFC20] TSUTSUI S., FU Y., CRANDALL D.: Whose hand is this? person identification from egocentric hand gestures, 2020. URL: https://arxiv.org/abs/2011.08900, doi:10.48550/ARXIV.2011.08900. 1, 2, 3, 8

[WMB*20] WANG J., MUELLER F., BERNARD F., SORLI S., SOTNYCHENKO O., QIAN N., OTADUY M. A., CASAS D., THEOBALT C.: RGB2Hands: Real-Time Tracking of 3D Hand Interactions from Monocular RGB Video. *ACM Transactions on Graphics (TOG) 39*, 6 (12 2020). 2, 3

[ZB17] ZIMMERMANN C., BROX T.: Learning to estimate 3d hand pose from single rgb images. In *IEEE International Conference on Computer Vision (ICCV)* (2017). https://arxiv.org/abs/1705.01389. URL: https://lmb.informatik.uni-freiburg.de/projects/hand3d/. 2

[ZBV*20] ZHANG F., BAZAREVSKY V., VAKUNOV A., TKACHENKA A., SUNG G., CHANG C.-L., GRUNDMANN M.: Mediapipe hands: On-device real-time hand tracking. *ArXiv abs/2006.10214* (2020). 2

[ZJS18] ZHOU H., JIANG F., SHEN R.: Who are raising their hands? hand-raiser seeking based on object detection and pose estimation. In

Proceedings of The 10th Asian Conference on Machine Learning (14–16 Nov 2018), Zhu J., Takeuchi I., (Eds.), vol. 95 of *Proceedings of Machine Learning Research*, PMLR, pp. 470–485. URL: https://proceedings.mlr.press/v95/zhou18a.html. 3, 8

[ZP13] ZARIFFA J., POPOVIC M.: Hand contour detection in wearable camera video using an adaptive histogram region of interest. *Journal of neuroengineering and rehabilitation 10* (12 2013), 114. doi:10.1186/1743-0003-10-114. 2