



Pure Physics-Based Hand Interaction in VR

Mohammed-Bashir Mahdi¹, Erwan Guillou¹, Alexandre Meyer¹^a, Arash Habibi²
and Saïda Bouakaz¹^b

¹LIRIS, CNRS, Univ. Lyon, Université Claude Bernard Lyon 1, 69100 Villeurbanne, France

²ICube, CNRS, Université de Strasbourg, 300 bd Sébastien Brant, 67400 Illkirch, France

Keywords: Virtual Reality, Interaction, Hooke's Spring Law, Physics, Physical Interaction, Hands, Natural Interaction, Visual Feedback.

Abstract: Interaction in Virtual Reality is still mainly done using controllers. However, since the early 2000s, there has been a desire to find a way to interact within the virtual environment using only our hands. Pinch motion detection was introduced to detect a grasping action. Since then, hands' motion capture has been highly developed until being directly integrated in VR headsets. Thus, multiple research projects were done in order to exploit this technology for better grasping techniques. Recent works tend to bring physical hand interaction to VR. However, they introduce physical heuristics to determine if an object is grasped or not, but in reality, motion is purely kinematic. In our paper, we introduce a purely physical method based on Hooke's spring law that eliminates the need for a grasping mode. Additionally, we incorporate visual feedback methods to compensate for the absence of the sense of touch. Consequently, with this approach, we can lift objects, throw them, stack them, and interact with them naturally. We carried out extensive tests with several people who had no previous experience, to validate our technique.

1 INTRODUCTION

Interacting with virtual environment has always been a hot topic. For instance, we can observe the evolution of controllers usage for interacting in video game on console or computers over the years. However, with the advent of new technologies like VR headset, this question arises again. Virtual reality gives a new way to explore the virtual environment and even to interact with it. However, most of commercial applications are made using controllers. This tends to evolve as new available interaction methodologies emerge.

Indeed, vendors themselves (*HTC, Meta*), provide SDK to exploit headset cameras in order to capture hand movements (HTC, 2023; Meta, 2022). When hands are captured, it is then possible to play with virtual object without using any wearable such as controllers. However, the methods proposed by vendors are purely kinematic, and lay on predefined hand poses to detect interactions (Buchmann et al., 2004).


Extensive research efforts have been directed towards achieving physically grounded interactions within the realm of virtual reality (Kim and Park,


2015; Höll et al., 2018; Liu et al., 2019). However, none of these approaches are fully physics-based. Instead, existing approaches predominantly establish heuristics for discerning the presence of interactions. They have two modes of interaction: a physics mode where an object moves following physical laws and a grasping mode where a grasped object follows the hand kinematics. These methods retain the grasping paradigm established by controller-based approaches, although with heuristics founded upon physical principles for determining whether an object is being held or not.

Our objective is to introduce a purely physics-based hand interaction method that avoids any form of grabbing mechanism altering the mode of interaction. Additionally, since we do not use additional wearable such as haptic gloves that provide tactile feedback, we incorporate visual feedback techniques to enhance the user's understanding of the environment.

2 RELATED WORK

This section first introduces the field of hand motion capture, with the goal of enabling natural interactions

^a <https://orcid.org/0000-0002-0249-1048>

^b <https://orcid.org/0000-0001-7091-7859>

in virtual reality through hand movements. Subsequently, various interaction techniques focusing VR domain are exposed.

2.1 Hand Capture

In order to capture hand pose data, different sorts of sensors may be used. We distinguish between approaches that capture the body and position of the hand, and those that capture hand movements including the fingers.

Complex and expensive solutions such as *OptiTrack* (Point, 2023) are capable of tracking and providing a comprehensive body and hand skeleton but at the cost of a cumbersome and inconvenient installation. Inertial measurement unit devices (IMU) can be used. The output data of these sensors are accelerations and angular velocities. IMUs are used for instance in PIP (Physical Inertial Poser) (Yi et al., 2022), coupled with a neural network to infer body joint positions. However, they reconstruct the body pose without the fingers. In the same category of capture methodology, LoBStr (Yang et al., 2021) and Quest Sim (Winkler et al., 2022) use the headset and the controllers position to extrapolate the whole body pose. Quest Sim uses a physics engine alongside a neural network in order to find the torques needed to move the different joints. With such an approach, they have a physical character which allows for physical interaction. With the increasing popularity of machine learning, multiple motion capture methods using only one color camera emerge (Cao et al., 2019; Zhang et al., 2020; Mehta et al., 2020). For example, Controller Pose (Ahuja et al., 2022) proposes putting wide angle cameras on the controllers, looking towards the user in order to capture its movement. As the method suggests, the user has to hold the controllers, so we are lacking finger pose. Geng (Geng et al., 2021) make the pipeline lighter as they use only one RGB camera, as *XNect* (Mehta et al., 2020).

Still, previous techniques do not provide the hand skeleton. Sensors such as Leap Motion (ultraleap, 2023) can provide good hand tracking. The device is equipped with several infrared camera sensors and LEDs that capture hand and finger movements in real time. UmeTrack (Han et al., 2022) and MEgATrack (Han et al., 2020) use multiple RGB cameras in order to focus on capturing hand poses. These methods make use of the four cameras on the *Oculus*. There are solutions using only one usual cameras. Liu (Liu et al., 2022) use architecture borrowed from deep learning for that. Indeed, transformer and attention network help to contextualize the articulation rotations and positions regarding the pose of the arm

and forearm. Google proposes a modern computer vision framework including real time hand capture system based on a single color camera called *Mediapipe* (Zhang et al., 2020). The implemented architecture is based on Blaze Face (Bazarevsky et al., 2019), a neural architecture developed for facial motion capture.

In our use case, the emphasis is on achieving hand placement with finger motion. We use the body skeleton only for precise hand positioning. Machine learning techniques for hand pose recognition are convenient because they often do not require the use of complex additional equipment to accomplish the task. This attribute makes them attractive because they can be easily deployed in a variety of conditions, including in small rooms or at home. Some headsets, such as the *HTV Vive Pro 2*, already feature deep learning-based hand pose reconstruction, eliminating the need for additional hardware. For the interaction, our aim is to perform hand capture without any accessory and to combine this capture with a completely physical interaction mode, with no need to switch to a kinematic mode.

2.2 Interaction in VR

In this section, we focus on full virtual interaction using hands to interact with virtual objects.

One of the most popular ways to interact in VR is done using controllers such as *Oculus Touch* and *HTV Vive Pro 2* controllers. Even if the interaction is constrained by the controllers, researchers tend to make it feel more natural by implementing grasping mechanisms (Han et al., 2023; Fernández et al., 2022; Oprea et al., 2019) that are at least visually more convincing. (Oprea et al., 2019) use physics to determine hand and finger positions when holding an object, depending on how the user approaches it in the virtual scene. (Han et al., 2023) use a deep neural network to compute torques applied on hand joints in order to move the hand more realistically and grab the targeted object using only a physical approach. This shows that even with only controllers, we can subdivide the interaction in two categories: geometry-based and physics-based interactions.

Another family of approaches relies on the use of cameras, often integrated into a modern VR headset, to capture hand movements and determine interactions within a preconfigured set of available actions. For instance, Hung in Puppeteer (Hung et al., 2022) use hand motion to control character motion in the environment of video games. Regarding hand-object interaction, Buchmann (Buchmann et al., 2004) propose an early method where they introduced the pinch

movement to trigger a grasping. We still see its legacy today, as most of the headset vendors still use this movement. However, Schafer et al. (Schäfer et al., 2022) let the user define its own grasping gesture as the pinch gesture is neither natural, nor adapted depending on the object the user wants to grab. They showed that letting the user define its own gesture increases the naturalness feeling of the interaction for that user.

Finally, pursuing the goal of making interactions more realistic and natural, many researches are made towards physics-based hand interactions (Höll et al., 2018; Kim and Park, 2015; Kaminski et al., 2022; Liu et al., 2019). *MuJoCo HAPTIX* (Kumar and Todorov, 2015) is purely physical, but at the cost of a heavy setup: haptic glove for force feedback and hand skeleton retrieval, paired with an *OptiTrack* system to position a hand in the world. The spring model from Borst (Borst and Indugula, 2006) is interesting as it performs physics-based interactions too, but still uses *CyberGloves* and works on hand joint torques. We aim at making the setup simpler with no additional wearable, and thus adapt the spring model. Regarding the other listed methods, the interaction is physical as long as the hand is not grasping any object, object movement becomes kinematic otherwise, which makes these techniques hybrid.

(Kim and Park, 2015) in his method selects 1000 vertices from a hand mesh in order to add sphere colliders on each one of them. These colliders are used to detect collisions with the objects in the scene. They realized that they were unable to have a stable system for fast paced movements, thus when grasping is detected they switch the moving object to kinematic mode to make it follow hand moves. This principle of having objects physically interact with the hand and switching to geometric interaction when grasping is detected is also found in the paper (Liu et al., 2019). In (Höll et al., 2018) physical approach is only used for the grasping heuristic. It relies on Coulomb friction model computation, not to move the object, but to determine if the applied force on the object satisfies the friction model. If so, then the object goes into a grabbed mode. Moreover, here one can see the hand being teleported as collision detection happens earlier. This is due to the increase of the object collider size, in order to prevent any physics engine instabilities. There is still ongoing research on making interaction being fully physical in VR as shown by the work of Kaminski (Kaminski et al., 2022). Unfortunately, they claim being only able to slide objects around, and grasping is not possible. When interacting in VR, one must use a proper hand representation in order to increase the user's experience

quality, regarding the set goal. Ferran (Argelaguet et al., 2016) shows that using a simple hand representation increases the feeling of embodiment. Moreover, Dewez (Dewez et al., 2023) discussed the interest of using a ghost hand representation when handling objects.

As described, current methods rely on a grasping mechanism. Indeed, most methods are hybrid, employing a physics heuristic to determine whether the object is grasped or not; if it is, they switch to a kinematic movement. Furthermore, these systems fix the position of the hand's fingers and palm, allowing only the wrist to move. Consequently, this limitation makes it impossible to perform more complex movements. This emphasizes the idea of having a pure physical interaction method similar to how interaction works in real life, in order to have a more natural feeling.

3 PHYSICAL INTERACTION

Our goal is to design a method for purely physical interaction, enabling the exploration of new movements. To accomplish this, we implement a force model based on principles of spring physics. We suggest leaving the hand bare, without the need for a controller or glove to capture it. Our method relies only on a hand motion capture system. Therefore, we must define new feedback techniques to address the absence of tactile feedback, a crucial aspect of real-life interactions with objects.

3.1 Our Approach

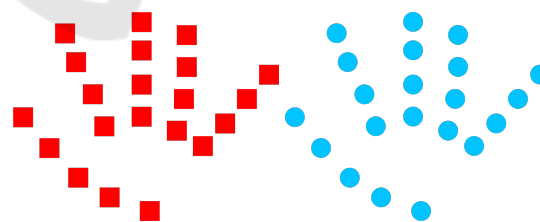


Figure 1: Scheme of both hand representations. The left one, in red, is the physical hand. The right one, in blue, is the ghost hand.

The best way to achieve completely natural interaction is to immerse the virtual world in a physical simulation, including fingers and hands. We base our approach on a dual representation of the hand: one physical hand and one ghost hand as shown in Figure 1. The physical hand interacts physically with the environment. The ghost hand takes the data received from the motion capture system, and its joints are po-

sitioned according to the position data received, regardless of the virtual environment. The ghost hand drives the physical hand. The physical hand can touch an object, move it, or if the object is too heavy for example, the hand is blocked by this object. It can not pass through objects. On the contrary, the ghost hand will act like a ghost, meaning that it goes through objects and has no physical interactions with its environment.

The key part of the model is how the joints of the physical hand are moved. The ghost joints will act like targets for the physical ones. The physical joints will try to reach their targets, but they are moved using forces and physics engine, thus they might be blocked if, for example, they encountered an obstacle on their way. Our model is related to the Proportional Derivative controller approach (Yin and Yin, 2020), except that PD controller provides usually torques whereas our model applies forces directly on joint positions. Using spring model for our interaction methods, we first refer to Hooke's law (Halliday et al., 2013) which states that the force of an elastic spring is the product of its length and a stiffness factor as shown in equation (1).

$$F_e = -k_e \times x \quad (1)$$

F_e is the resulting elastic force, k_e the rigidity constant (with $k_e > 0$) and x the displacement of the spring. The spring's rest length is zero.

This first F_e term aims to attract the joint position towards the target. In order to complete this spring model, we add a second term to ensure that the velocity of the joint targets the velocity of the target joint. We compute this force acting on the derivative by (2).

$$F_d = -k_d \times \Delta v \quad (2)$$

With F_d the damping force, k_d the gain of this force (with $k_d > 0$) and Δv the difference between the speed and a target speed.

Finally, the total spring force F_s is the sum of the two previous forces (3).

$$F_s = F_e + F_d \quad (3)$$

As described earlier, the applied force uses a spring model. The x value of equation (1) is the difference between the new ghost joint position and the physical joint position. It means that the spring seeks to get a length of zero. Δv from equation 2 is the difference between the velocities of both hand representations. Finally, we add the two previous forces and end up with the spring force like in (3).

The algorithm 1 shows how we implemented the model in order to compute the spring force applied to the physical joints. During each iteration of the physics engine, this algorithm is performed on each

Data: $p_p, p_g, k_e, \Delta_t, k_d$

Result: Added force to the physical joint.

```

 $d \leftarrow p_p - p_g;$ 
 $f_e \leftarrow -k_e \times d;$ 
 $v_g \leftarrow (p_g - p_{prev.g}) / \Delta_t;$ 
 $f_d \leftarrow -k_d \times (v_p - v_g);$ 
 $rb.AddForce(f_e + f_d);$ 
 $p_{prev.g} \leftarrow p_g;$ 

```

Algorithm 1: Computing the force.

hand joint. d is the displacement used to compute the elastic force f_e , with p_p the position of the physical joint, and p_g the position of the ghost joint. We compute the speed v_g of the ghost joint by dividing the difference between its current position and its previous position $p_{prev.g}$ by the time for one iteration of the physics engine Δ_t . After computing the derivative force (with v_p the speed of the physical joint, provided directly by the physics engine in our case), we sum the two forces to get the total spring force. Then we add the force to the rigid body rb of the physical joint in order to make it move.

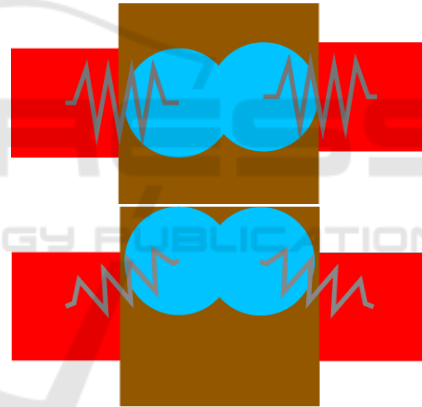


Figure 2: Scheme of how the spring forces (grey spring) are applied on the physical joint (in red) in order to move an object (in brown) with the ghost hand (in blue) not being stopped by the object.

Schemes in Figure 2 show how the forces are applied. After computing the force for each joint, it is added to the rigid body of the corresponding joint. Then, if the force is high enough, it will generate the required amount of friction between the joints and the object to make the object not slide from the fingers. The physics engine will handle the friction calculation by itself as well as the force transfer between the joints and the object. Moreover, the force direction will adapt regarding of the movement. For example, if the capture joints in the new frame are positioned higher than the current physical joints, the direction will be equal to the vector between both kind of joints.

3.2 Visual Feedback

When we grasp an object in real life, we know we are in contact with the object because of the sense of touch. If we rely only on the visual sense, it makes the task more difficult: we get the data that the fingers behind are touching the object when we see the cup moving, or at least when waves appear in the coffee. To deal with these problems, one can use a haptic glove like (Kumar and Todorov, 2015) suggested. However, our goal is to remove wearable as much as possible (controllers, gloves), thus we believe that it is possible to help these problems using visual feedback only (Prachyabrued and Borst, 2014; Vosinakis and Koutsabasis, 2018; Ahmed et al., 2021).

We introduced several visual notifications. First of all, when a joint, from the ghost hand, gets into contact with an object, the corresponding physical joint color turns into a light blue, and switch back to its original color when the contact is lost (object falling, joint no more in contact with the object). At the same time, when there is contact, the corresponding object becomes translucent. This emphasizes the information of contact, but provides also the possibility to see through the object and thus be able to know if the hand joints which were originally occluded by the object are in contact or not.

As mentioned previously, we use a double representation of the hand. We decided to let both of them visible. By being able to see the ghost hand, it will help the user to infer the amount of pressure he is applying with the joint being in contact with the object. Indeed, as described in section 3.1, the ghost joint penetrates the object and the distance between the ghost joint and the physical joint gives the amount of force to be applied. Thus, by being able to see both joints, the user can then see the distance between them and estimate how much pressure he needs to apply.

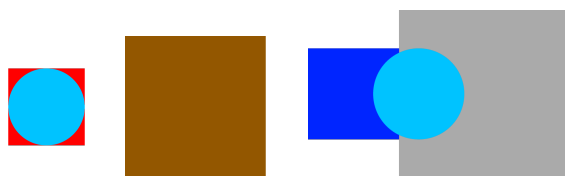


Figure 3: Scheme of the visual feedbacks. Left when there is no contact. Right when there is contact between the hand and an object. The scheme is reduced to the interaction of one joint.

The Figure 3 shows the working of the visual feedbacks. The change of color of the different elements (physical joint and the object getting transparent), and the ghost joint being visible to see its level of penetration in the object.

4 RESULTS

In order to implement our model, we used the Unity engine and its built-in physics engine *PhysicX*. The value k_e of the elastic force is set to 1000, and the value k_d of the derivative force to 20. Those values have been determined empirically, and fixed to the most stable values found. The VR headset used is the *HTV Vive Pro 2* and we use its SDK (HTC, 2023) to perform the hand motion capture. Everything was implemented in a computer equipped with an Intel Xeon CPU (Xeon E5-1603 at 2.80GHz), a GTX 1080 GPU with 8GB of VRAM and 16GB of RAM. We decided to color the joints in red for the left hand and in green for the right one.

For the testing phase, we designed four different scenes that require different skills:

- **Throw:** the user must throw a cube at a pyramid in order to destroy it.
- **Sphere:** the user must flick a sphere to make it fly into a predefined area.
- **Displacement:** the user must take a cube and place it into a tilted hole in a wall.
- **Stack:** the user must make a stack with three cubes.

To validate our model, twelve subjects tested all scenes. All of them come from different backgrounds. Most of them had never used VR or just once. If they did use VR, it was for playing video games, using controllers, or for watching a movie. All of them stated that it was their first time interacting in a virtual environment in VR by using only their hands and without any controller.

To compare users' performances, we record a reference time established by an additional experimented person. Table 1 shows the ratio of a novice subject completion time over the reference user completion time (to get the actual time of a subject i , subject ration must be multiplied by the reference completion time $R \times Si$). One can see that R was beaten by a small margin three times, in three different exercises: S3 in throw by 1s, S12 in sphere by 0.7s and S11 in placement by 0.4s. Moreover, 79% of the total number of experiments were successful, with a 100% success rate for the sphere scene. Moreover, the average time differences shows that one requires between 30 and 110 seconds in order to perform a task without previous training. Finally, the median in Table 3 is interesting as it tells that half of the users have been able to perform each task with in a minute or less, and with throw and sphere tasks completed in both in less than 30 seconds.

Table 1: Ratio of completion time by each subject over reference time ($\frac{S_i}{R}$). FAIL when the experiment was not completed by the subject. S_i for subject i . R is the reference user's time in second. T: throw, S: Sphere, P: placement, S: Stack.

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	R
T	FAIL	3.84	0.85	5.65	FAIL	FAIL	FAIL	FAIL	27.24	1.14	3.14	5.19	6.8
S	11.15	16.61	19.51	5.23	5.84	3.05	4.44	17.03	10.47	11.23	6.41	0.77	3.2
P	17.50	3.64	8.98	7.97	23.29	1.55	FAIL	8.84	2.84	FAIL	0.96	4.74	8.6
S	2,68	1.14	FAIL	8.27	FAIL	1.42	12.75	FAIL	2.19	6.02	1.84	1.01	28.3

Table 2: Table of the average times, the variance and the standard deviation, based on the measure of Table 1.

	AVG	Variance	Standard deviation
Throw	45.7	3940.9	62.8
Sphere	29.5	363.0	19.1
Placement	68.8	3859.0	62.1
Stack	110.6	13191.9	114.9

Table 3: Table of the quartiles and median of the measures from Table 1.

	1 st quartile	median	3 rd quartile
Throw	14.6	26.1	36.8
Sphere	15.9	26.7	39.8
Placement	26.1	54.5	76.6
Stack	40.0	61.9	170.1

Furthermore, the reference time for throw, sphere and placement are low (resp. 6.8s, 3.2s and 8.6s). For the stack scene, the times are higher because we had to cope with the hand motion capture which tends to freeze. This problem was pointed by every user, but they still claimed that the interaction itself feels natural.

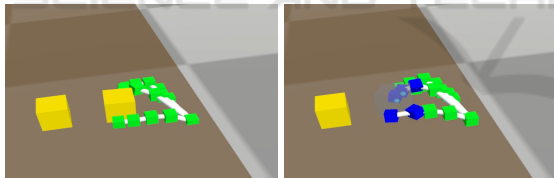


Figure 4: Joints and objects changing color when interacting with each others.

Figure 5 shows that it is possible to lift objects with our method. It is worthy to note that it is possible to lift an object such as a cube, even though it has no handle which could prevent the object from sliding. We see that we are able to cage the object with our hand to properly lift it. Finally, we can move our hand around without the cube falling, and put it wherever we want.

When holding an object, if we move fast enough and release it, we are able to throw it as it will have gained enough inertia not to fall right in front of us. One can throw an object into others as shown in Figure 6. Thus, we can also adjust the force of the thrown object by adjusting the speed of our movement, like in real life, which is only feasible when using a pure

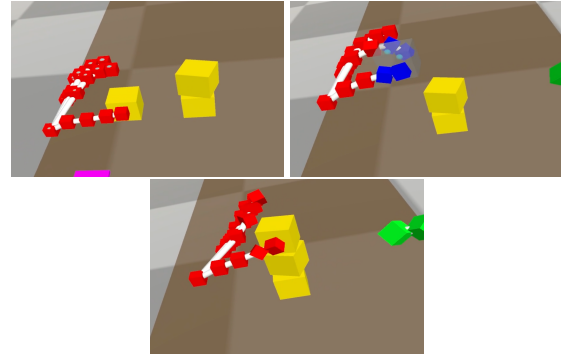


Figure 5: Lifting a cube to make a stack. Top left: preparing the hand. Top right: grabbing and lifting the cube. Bottom: opening the hand to let the cube on the stack.

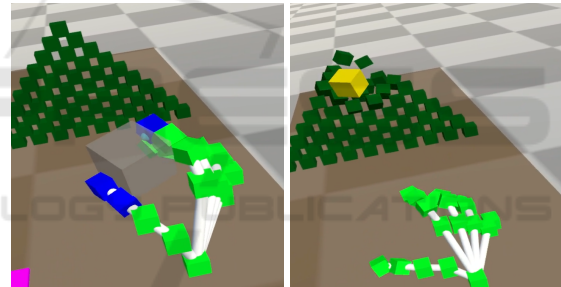


Figure 6: Throwing an object.

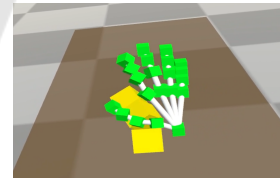


Figure 7: Punching a stack of cubes.

physical approach. Moreover, when trying to put the cube in a tilted hole, the cube will follow the rotation of the hand naturally in order to be placed correctly.

As we do not have any grasping mode and that everything is physical, it makes possible to punch objects and make them fly all over the place as shown in Figure 7. Since we are not limited to predefined actions from a controller or any grasping detection methods, we are free to interact naturally with virtual environments.

Figure 4 shows how visual feedback works. The joints in contact with the cube see their color change, as well as the object itself get transparent. This point is also illustrated in Figure 4, 5 and 6.

5 DISCUSSION

Results obtained are promising, nevertheless there are still areas where different approaches could be tried. Indeed, the actual state of the projects requires a learning phase from the user in order to be able to exploit the full potential of manipulating objects in VR. As we are lacking haptic feedback, the user must be trained to manipulate objects without touching anything with his hands. Thus work could be done in the feedbacks to try to cope with this problem and reduce the adaptation time.

To improve this model, when a user desires to interact with an object, it might be interesting to make this object transparent before it gets touched. Multiple ways can be investigated. As we do not want to add additional wearable such as gloves, we rely strongly on visual feedbacks. To maybe improve such model, it might be interesting to make the object the user wants to interact with transparent before it gets touched. For instance, by increasing the touch detection area of the object, without affecting the physics, to be able to see through it and thus be able to see the part of the hand which is behind it.

6 CONCLUSION

We proposed a method that extends the paradigm of physics-based hand manipulation of objects in VR by completely eliminating the notion of kinematic state. Indeed, we eliminate all sorts of grabbing mechanisms which switch the interaction from physics to kinematic. In addition, our pipeline allows to avoid all additional accessories such as haptic gloves. To compensate for the absence of haptic feedback, we introduced visual feedback to help the user deduce the missing tactile information. Finally, the results showed that it is possible to grab objects and even push them as we would do in real life. The object-throwing scene would not be possible with a kinematic approach. The user would have to use the button to indicate a release, but the interaction would then be different from what we do in real life. A hybrid approach would require the application to really detect the release at the right moment, at the risk of having an unrealistic trajectory. Moreover, in the scene where the cube has to be placed in a hole

in a sloping wall, the kinematic approach would require an uncomfortable turn of the wrist and would surely be less easy than with our physical approach where the cube is oriented against the surface thanks to physics.

Our test scenes show that a purely physical approach is feasible, and that novice users are able to perform tasks quite quickly, which would not be so obvious with kinematic or even hybrid interaction. Nevertheless, the model could be improved. The learning curve in order to adapt to this interaction system is steep, mainly because hand motion capture is sometimes not stable. There might still be some fine-tuning to improve grasping. Also, the proposed visual feedback needs to be explored further, or even dig into another kind of feedbacks such as audio.

For future work, it can be interesting to focus on feedback development by exploring the use of sounds. The program could emit a sound when we approach an object, similar to the way modern cars have parking sensors. However, in scenes with multiple closely located objects, reaching for one of them could result in numerous sounds playing simultaneously. Alternatively, sound could be emitted only when a hand joint comes into contact with an object. What is interesting here, is that we can make the audio feedback methods work alongside the visual feedback methods. Moreover, using audio feedback does not need any additional wearable as VR headset tend to have earphones, and if not, any headset might do the job as well as the computers built-in speakers.

REFERENCES

- Ahmed, N., Lataifeh, M., and Junejo, I. (2021). Visual pseudo haptics for a dynamic squeeze / grab gesture in immersive virtual reality. In *2021 IEEE 2nd International Conference on Human-Machine Systems (ICHMS)*, pages 1–4.
- Ahuja, K., Shen, V., Fang, C. M., Riopelle, N., Kong, A., and Harrison, C. (2022). Controllerpose: Inside-out body capture with vr controller cameras. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, New York, NY, USA. Association for Computing Machinery.
- Argelaguet, F., Hoyet, L., Trico, M., and Lécuyer, A. (2016). The role of interaction in virtual embodiment: Effects of the virtual hand representation. *IEEE Virtual Reality*.
- Bazarevsky, V., Kartynnik, Y., Vakunov, A., Raveendran, K., and Grundmann, M. (2019). Blazeface: Sub-millisecond neural face detection on mobile gpus. *ArXiv*, abs/1907.05047.
- Borst, C. W. and Indugula, A. P. (2006). A spring model for whole-hand virtual grasping. *Presence: Teleoperators & Virtual Environments*, 15:47–61.

- Buchmann, V., Violich, S., Billinghamurst, M., and Cockburn, A. (2004). Fingertips: gesture based direct manipulation in augmented reality. In *Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia*.
- Cao, Z., Hidalgo Martinez, G., Simon, T., Wei, S., and Sheikh, Y. A. (2019). Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Dewez, D., Hoyet, L., Lécuyer, A., and Argelaguet, F. (2023). Do you need another hand? investigating dual body representations during anisomorphic 3d manipulation. *IEEE Transactions on Visualization and Computer Graphics*.
- Fernández, U. J., Elizondo, S., Iriarte, N., Morales, R., Ortiz, A., Marichal, S., Ardaiz, O., and Marzo, A. (2022). A multi-object grasp technique for placement of objects in virtual reality. *Applied Sciences*, 12(9).
- Geng, Z., Sun, K., Xiao, B., Zhang, Z., and Wang, J. (2021). Bottom-up human pose estimation via disentangled keypoint regression. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14671–14681.
- Halliday, D., Resnick, R., and Walker, J. (2013). *Fundamentals of physics*. John Wiley & Sons.
- Han, D., Lee, R., Kim, K., and Kang, H. (2023). Vr-handnet: A visually and physically plausible hand manipulation system in virtual reality. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–12.
- 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., and Wang, R. (2020). Megatrack: Monochrome egocentric articulated hand-tracking for virtual reality. *ACM Trans. Graph.*, 39(4).
- Han, S., Wu, P.-C., Zhang, Y., Liu, B., Zhang, L., Wang, Z., Si, W., Zhang, P., Cai, Y., Hodan, T., Cabezas, R., Tran, L., Akbay, M., Yu, T.-H., Keskin, C., and Wang, R. (2022). Umetrack: Unified multi-view end-to-end hand tracking for vr. In *SIGGRAPH Asia 2022 Conference Papers*, SA '22, New York, NY, USA. Association for Computing Machinery.
- Höll, M., Oberweger, M., Arth, C., and Lepetit, V. (2018). Efficient physics-based implementation for realistic hand-object interaction in virtual reality. In *Proc. of Conference on Virtual Reality and 3D User Interfaces*.
- HTC (2023). Vive hand tracking sdk. Accessed: 2023-10-03.
- Hung, C.-W., Chang, R.-C., Chen, H.-S., Liang, C. H., Chan, L., and Chen, B.-Y. (2022). Puppeteer: Exploring intuitive hand gestures and upper-body postures for manipulating human avatar actions. In *Proceedings of the 28th ACM Symposium on Virtual Reality Software and Technology*, VRST '22, New York, NY, USA. Association for Computing Machinery.
- Kaminski, E., Kepplier, S., and Israel, J. H. (2022). Physics-based hand-object-interaction for immersive environments. *Mensch und Computer*.
- Kim, J.-S. and Park, J. M. (2015). Physics-based hand interaction with virtual objects. *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3814–3819.
- Kumar, V. and Todorov, E. (2015). Mujoco haptix: A virtual reality system for hand manipulation. *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 657–663.
- Liu, H., Zhang, Z., Xie, X., Zhu, Y., Liu, Y., Wan, Y., and Zhu, S.-C. (2019). Physics-based hand-object-interaction for immersive environments. *IEEE International Conference on Robotics and Automation (ICRA)*.
- Liu, S., Wu, W.-X., Wu, J., and Lin, Y. (2022). Spatial-temporal parallel transformer for arm-hand dynamic estimation. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24091–24100.
- Mehta, D., Sotnychenko, O., Mueller, F., Xu, W., Elgharib, M. A., Fua, P., Seidel, H.-P., Rhodin, H., Pons-Moll, G., and Theobalt, C. (2020). Xnect: Real-time multi-person 3d motion capture with a single rgb camera. In *International Conference on Computer Graphics and Interactive Techniques*.
- Meta (2022). Interaction sdk overview. Accessed: 2023-10-03.
- Oprea, S., Martinez-Gonzalez, P., Garcia-Garcia, A., Castro-Vargas, J. A., Orts-Escolano, S., and Garcia-Rodriguez, J. (2019). A visually realistic grasping system for object manipulation and interaction in virtual reality environments. *Comput. Graph.*, 83(C):77–86.
- Point, N. (2023). Optitrack. Accessed: 2023-10-03.
- Prachyabrued, M. and Borst, C. W. (2014). Visual feedback for virtual grasping. *2014 IEEE Symposium on 3D User Interfaces (3DUI)*, pages 19–26.
- Schäfer, A., Reis, G., and Stricker, D. (2022). The gesture authoring space: Authoring customised hand gestures for grasping virtual objects in immersive virtual environments. *Mensch und Computer*, pages 1–12.
- ultraleap (2023). ultraleap.com. Accessed: 2023-07-23.
- Vosinakis, S. and Koutsabasis, P. (2018). Evaluation of visual feedback techniques for virtual grasping with bare hands using leap motion and oculus rift. *Virtual Reality*, 22:47–62.
- Winkler, A., Won, J., and Ye, Y. (2022). Questsim: Human motion tracking from sparse sensors with simulated avatars. In *SIGGRAPH Asia 2022 Conference Papers*, SA '22, New York, NY, USA. Association for Computing Machinery.
- Yang, D., Kim, D., and Lee, S.-H. (2021). Lobstr: Real-time lower-body pose prediction from sparse upper-body tracking signals. *Computer Graphics Forum*, 40.
- Yi, X., Zhou, Y., Habermann, M., Shimada, S., Golyanik, V., Theobalt, C., and Xu, F. (2022). Physical inertial poser (pip): Physics-aware real-time human motion tracking from sparse inertial sensors. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Yin, Z. and Yin, K. (2020). Linear time stable pd controllers for physics-based character animation. *Computer Graphics Forum*, 39(8):191–200.
- Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C.-L., and Grundmann, M. (2020). Mediapipe hands: On-device real-time hand tracking. *ArXiv*, abs/2006.10214.