이화여자대학교대학원
2015 학년도
석사학위청구논문

Physics-based and Assistive Grasping for Robust Object Manipulation in Virtual Reality

컴퓨터공학과

Kiran Nasim

2016

Physics-based and Assistive Grasping for Robust Object Manipulation in Virtual Reality

이논문을석사학위논문으로제출함

2016 년 6 월

이화여자대학교대학원

컴퓨터공학과 Kiran Nasim

Kiran Nasim 의석사학위논문을인준함

지도교수	**김영준**	_______
심사위원	**김명희**	_______
	김영준	_______
	Lien Jyh-Ming	_______

이화여자대학교대학원

Acknowledgments

First of all, I would like to express my deepest gratitude to project advisor Prof. Young J. Kim for his guidance and supervision throughout my study and research. I feel lucky to be a part of his research team as he has provided a comfortable working environment. I also thank HDI^{2}4D joint research collaboration, funded as part of the New Zealand - Korea Strategic Research Partnership, for sponsoring this research, and providing an international experience and friendly working environment.

Moreover, my sincere appreciation goes to Ewha Computer Graphics Laboratory members for their help and motivation in difficult times, and to make me feel at home in Korea. Also, I wish to thank the Ewha Womans University students who participated in the user study for this work. Finally, I thank my husband for his endless support and encouragement.

TABLE OF CONTENTS

List of Figures .... **vii**

List of Tables .... **ix**

Abstract .... **x**

I. Introduction **1**

1.1 Dissertation Goals 2

1.2 Challenges 3

1.3 Main Contributions 4

1.4 Dissertation Overview 5

II. Related Work **6**

2.1 Grasp Planning 6

2.1.1 Grasp Classifications 7

2.1.2 Hand-Object Interpenetrations 8

2.2 Interaction in Virtual Reality 9

2.2.1 Gesture-based Interaction 9

2.2.2 Physics-based Interaction 10

2.3 Hand Tracking Devices 11

III. System Overview **13**

3.1 System Architecture 13

3.2 Grasp Statics 14

3.2.1 Friction Point Contact 15

3.3 Leap Motion Controller 17

3.3.1 Coordinates Mapping 19

IV. Virtual Grasping Algorithms **20**

4.1 Visual Feedback 20

4.1.1 Proxy Hand 21

4.2 Collision States 23

4.3 Physics-based Grasping 24

4.3.1 Algorithm..25

4.3.2 Results..26

4.4 Assistive Grasping 27

4.4.1 Algorithm..28

4.4.2 Results..30

V. Implementation and Experiments 32

5.1 Implementation Platform 32

5.1.1 Unity3D Game Engine...32

5.2 Benchmarks..33

5.3 User Study..34

5.3.1 Experimental Setup..35

5.3.2 Analysis 36

VI. Conclusion...41

Bibliography...43

Abstract (Korean)..49

List of Figures

Figure 1: *Grasp Categories* *7*

Figure 2: *Grasping patterns of a child* *8*

Figure 3: *Pinch Gesture Sequence* *10*

Figure 4: *System Architecture* *13*

Figure 5: *Contact and object forces Coordinate Frames* *14*

Figure 6: *Geometric interpretation of Coulomb's friction model* *16*

Figure 7: *Leap Motion hand models* *17*

Figure 8: *Leap Motion Diagnostic Visualizer* *18*

Figure 9: *Leap Motion Controller* *19*

Figure 10*: Physics hand and graphical hand alignment* *20*

Figure 11: *Penetration situations for virtual hand and proxy hand* *21*

Figure 12: *Leap Motion skeletal hand taxonomy* *22*

Figure 13: *Leap hands and corresponding proxy hands* *22*

Figure 14: *System Collision States* *23*

Figure 15: *Collision states for proxy hand* *24*

Figure 16: *Physics-based grasping algorithm for proxy hand* *25*

Figure 17*: Grasping sequence without proxy hand, showing penetrations..........26*

Figure 18: *Grasping sequence with proxy hand, resolving penetrations.............27*

Figure 19: *Overlap sphere with hand palm as center..29*

Figure 20: *Assistive grasping algorithm..30*

Figure 21: *Assistive grasping sequence...31*

Figure 22: *Assistive grasp results for a Lego model 31*

Figure 23: *3D Benchmarking Models...33*

Figure 24: *A subject of our user study trying the setup.......................................35*

Figure 25: *Images of a user achieving the goal..36*

Figure 26: *Comparison of three approaches based on survey questions.............37*

Figure 27: *Comparison of three approached based on user data 38*

Figure 28: *Learning curve for assistive grasp...40*

List of Tables

Table 1: *Model complexities of benchmarking models …………………………… 33*

Table 2: *Results of the user study ……………………………………………………. 39*

Table 3: *User data of learning curve for assistive grasp ………………………… 40*

Abstract

Interaction is the key to a decent real world Virtual Reality (VR) experience but there exists a significant gap between the two domains as major interaction methods in VR include voice, gaze, tools and gestures instead of using the virtual hand as a counterpart of a real hand. In this dissertation, an interactive virtual grasping system is proposed, enabling physically realistic interaction between a user and virtual objects, using kinematic hand models whose motions are tracked and controlled in real-time by the user. Since hand tracking devices provide only one-way coupling between the user and kinematic hands resulting in penetrations and frictionless interaction, our method implements the contacts as Coulomb's friction model and includes a dynamic proxy hand in the system which follows the kinematic virtual hand to provide a two-way control loop for the physics interactions. The introduced proxy hand not only provides visual feedback but also is used to simulate dynamics between the hand and virtual objects. We propose two grasping methods; physics-based grasp and assistive palmar grasp where the former ensures friction contact and avoid penetrations while the later makes system help the user in achieving a grasp by identifying user intention. We have implemented and evaluated each of these techniques with various benchmarking settings and conducted a user study to compare our methods against the existing pinch-based grasping mechanism. The experimental results show how our assistive grasp provides fast and effective solutions by performing better in unknown virtual environments for a naïve user and physics-based method helps to provide friction contacts resulting in physically real interactions in the virtual world. Moreover, survey questions show that users prefer assistive grasp in terms of its practicality and effectiveness. Our system, taking advantage of the graphics and physics engine of used game engine, allows us to interact with 3D objects in the virtual space in real-time with no visible latency. Moreover, to visually assist the user's grasping we provide visual markers at the contact locations and rendered proxy hand as principle visual grasping hand, as it avoids penetrations as compared to the actual virtual tracking hand and provides system stability. We demonstrate how the use of these tools helps us to perform real-time grasping tasks in complicated virtual environments.

I. INTRODUCTION

With the recent advancements in virtual reality (VR), 3D computer graphics and animations, the efforts to create and simulate virtual humans are becoming more and more popular, resulting in the emergence of new interaction devices that enables us to precisely sense the environment and gives us a new dimensional experience. These devices aiming to improve human-computer interaction (HCI) have changed and expanded the ways in which we can interact with a virtual environment [1] [2].

In the approaching times, we believe that everyone will be able to experience the virtual world combined with the actual world, as many types of research are being done to simulate physical realism in the virtual world. This dissertation explores how a simple system can allow the user not only to freely interact with the virtual world in real-time but also able to manipulate virtual object making the interactions more close to the real world.

One of the main focuses of HCI is to make interaction easier and more intuitive allowing a large spectrum of people to better use the system [3]. This makes grasping the most natural tool for human-computer interaction as humans are accustomed to grasping for performing daily life tasks. Thus, grasping as a natural instinct of human intellect is expected to be the same in an immersed virtual environment. Hence, grasping in human-computer interaction draws recent research interests with broad applications in computer graphics, artificial intelligence, industrial design, surgery and much more [4] [5] [6] [7] [8].

Fully autonomous grasping has proven to be a very challenging problem in robotics due to a huge number of possible hand configurations. Moreover, multi-finger end-effector increases the kinematic and dynamic complexity of the system [9]. Modern sensors and technological advancements have provided us with sophisticated sensors that are able to precisely sense the environment and its changes. Although, including the human as a part of the environment in the system (i.e. human in the loop) makes the grasping problem relatively simple as humans can choose appropriate feasible posture to perform the interaction task [2]. However, the great number of degrees of freedom of the human hand coupled with the space of grasping parameters and the geometry of the object creates a high dimensional problem in configuration space [10].

Creating physically realistic animation for human grasping is also a non-trivial task [11]. Dealing with the virtual world in parallel with the real world always results in discrepancies, as the virtual world cannot be expected to obey the laws of physics of the real world. As the user can not actually feel the object, i.e. *one-way coupling* between the user and the kinematic hand without proper haptic feedback, the touch can result in hand-object inter-penetrations or the contact may seem frictionless.

1.1 Dissertation Goals

As grasping is expected to be the same in an immersed virtual environment, the ultimate objective is to provide grasp as real to the physical word as possible in VR. Data from available tracking devices, when provided as input to the virtual hand results in driving the kinematic virtual hand but for a feasible grasp, it is necessary to add dynamics to the system, as this may result in undesired and unstable situations in the virtual world because of this one-way control.

Our goal, therefore, is to generate real-time motion control for grasping a wide variety of objects in a virtually simulated environment, while providing control solutions for the kinematic hand model. Moreover, the way we address the *one-way coupling problem* is to receive tracking data as input from the tracking device and transfer it to another hand called "proxy hand" in our system so that physics simulation will be carried through this proxy hand and dynamics can be simulated between hand and objects.

We aimed to improve grasping capability of existing pinch-based magnetic grasping technique [12] [13], as it is purely control based where a specific control gesture (pinch) is provided by the user, resulting in the corresponding action to make it possible to pick and move the virtual objects. We planned our approach in such a way to make this type of control as realistic as possible by making use of palmar grasp instead of pinch grasp as palmar grasp is the most basic type of grasp and natural instinct in infants to grasp objects [14].

We have provided two methods, physics-based grasping and assistive palmar grasping against the existing state-of-art pinch-based grasping. The physics-based method ensures contact based on Coulomb's friction model and introduces proxy hand in the system as the main interaction hand while the assistive grasping method includes the system as a grasping aid to identify various collision states and grasp objects accordingly. This algorithm not only achieves grasp more close to the real world but also provides control solutions for one-way coupling limitation of hand-tracking devices.

To achieve the real-time interactive performance we used '*colliders*' with each object. Collider is a simplified shape of the object attached with it and all collision detection will be carried out between these colliders. This way we avoid mesh-mesh collision and achieve better performance. The proposed system's performance, validity, and reliability are checked by conducting a user study against pinch-based magnetic grasp technique.

1.2 Challenges

Simulating interactions in the virtual world is a complex problem due to lack of physical constraints and anatomical complexity of the human hands, making it challenging to achieve physically realistic interactions in VR. Moreover, interaction in the virtual world is a new task to a human so it is required to make it as close to the real world as possible, also to make the user familiarize with the new world by training. Main challenges identified, to achieve physically realistic interactions in virtual domain, are as follows;

High DOF of human hand: Challenges to achieve the goal includes first of all dealing with the complex structure of human hand as it has more than 20 degrees of freedom (DOF), making kinematic and dynamic analyses of the system complex and, increasing the difficulty of achieving a feasible grasp.

Virtual environment ambiguity: Dealing with the virtual world in parallel with the real world always results in discrepancies, as the virtual world cannot be expected to obey the laws of physics of the real world resulting in penetrations and frictionless grasp.

One-way device-user coupling: Hand tracking device do not provide any control for the virtual hand, as the only driving factor for the hand is the raw data from real hand, resulting in unstable situations in the virtual world.

Real world to virtual world transformations: Moreover, we need to perform certain transformations to convert data from real life environment to the virtual one so that interactions can be simulated.

Real-time collision resolution: Collision resolution algorithms are required in real-time to make feasible interactions to achieve interactive performance.

1.3 Main Contributions

Our work’s main contributions include:

- a hand-object contact performance interface that allows the user to create a desired grasping action by acting out the motion in front of a single, off-the-shelf sensor
- online physics simulation, with one to one mapping to the physical world, resolving collision situations and providing visual aids
- virtual grasping without any need to have pre-planned grasp pose database
- a control solution for hand tracking devices which only provide one-way coupling between the device and user, by including a dynamic hand called “proxy hand” in the system.

Pinch-based grasping method, a gesture-based method, also provides real-time solutions but 0by keep track of collision states and resolving the hand-object interpenetrations to provide more stable interactions. Moreover, pinch-based method is an artificial control method, same like pressing a button to have a stored result, assuming hand as a controller and providing a reaction against an action performed by hand.

1.4 Dissertation Overview

The organization of this dissertation is as follows: chapter 1 provides an introduction to the work including research goal and main challenges to achieve the goal. Chapter 2 gives a detailed background of different fields involved and studied to establish this research. Chapter 3 discusses our system overview including basic grasp statics involved, followed by the system architecture. In chapter 4, we introduce our proposed grasping methods, discussed the algorithms and showed basic results for each method. Chapter 5 describes implementation platform and system evaluation by the conducted user study. This chapter compares our methods against existing pinch based online grasping and provides a detailed analysis of the differences and improvements. Finally, chapter 6 concludes the dissertation by providing a summary of the work and discussing future improvements possible.

II. RELATED WORK

This chapter gives a background of the existing research on grasp planning and interactions in virtual reality and the related issues to be considered.

2.1 Grasp Planning

Finding stable grasps for 3D virtual objects is considered to be one of the hardest problems in robotics and Virtual Reality since many parameters such as object geometry, hand kinematics, and material properties have to be considered. This result in a high dimensional space of possible grasp, therefore it is required to plan possible grasp pose and sequence before head to reduce the complexity of the problem [15].

Grasp planning and optimization have been hot topics in robotics and computer graphics since last decade. Most of the grasp planning algorithms are either using offline shape matching [16] or active learning database to store the set of feasible grasps [16] [11]. Thus, mostly the available systems can operate offline to pre-compute configurations for hand poses to create stable grasps [17] [11].

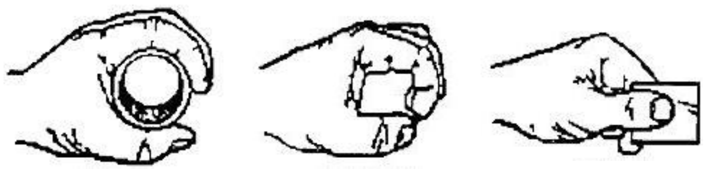
For robotic grasp simulations, very few open source simulators are available. Miller [18] proposed a grasp planning dynamic engine called GraspIt to perform grasp gesture and evaluate the grasp postures for different objects. However, the system is not for real-time interactions, also this work does not include any rich dimensional variations of the hand models [19].

Very few researchers are working on real-time physics-based interactions without any prior knowledge of the environment because prior synthesized grasp poses are usually natural looking and consistent with the real world data [11]. Kyota [20] combined pre-recorded grasp poses and grasp taxonomy for interactive grasp synthesis. But such systems always encounter delays and cannot deal with the un-predictive situations.

Many methods use supervised or unsupervised learning techniques [21] [22] [7] to predict the stability of a grasp but the shortcoming of these methods is that they require a sufficiently huge set of coherent labeled data [23]. Moreover, most of the research in this field has focused on kinematic systems to select a suitable pose. In computer graphics, Shapiro [24] combined simulation and motion capture playback to shift between kinematics and dynamics. Kitamura et al. [25] [26] proposed a system for manipulation of objects using virtual chopsticks monitoring the established contact between object and chopsticks and adjusting the finger angles accordingly unless chopsticks penetrate the object.

2.1.1 Grasp Classifications

Grasp synthesis is often classified into analytical and empirical methods, as described by Sahbani et al. [27] [8] , where the former involves kinematics and dynamics of the hand pre-shapes and force impacts while the latter focuses on learning and classification methods. According to Sclesinger [8] [28], there can be six types of grasps i.e. cylindrical, tip, hook, spherical, palmar and lateral. A more compact form of the above classification for grasp analysis in both real and virtual worlds is shown in figure 1.

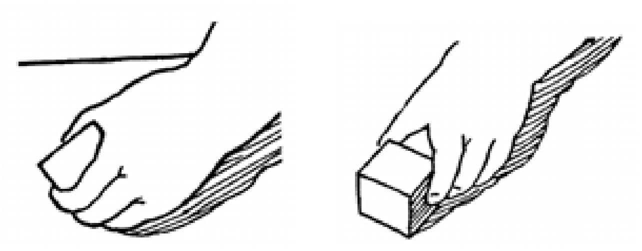


a. Palmar (Cylindrical) *b. Pinch (Tip)* *c. Lateral*

Figure 1: *Grasp Categories*
Figure is taken from Taylor and Schwarz [8] [60]

A more recent classification involves power and precision grasp by Napier [29]. “Power grasps” include *palmar* and *spherical* types, as for these grasps palm and fingers hold an object whereas the “precision grasp” can be subcategorized as a *tip, hook,* and *lateral* types, as in these only fingers are in contact with the target object.

As virtual grasping is relatively a new field of study, so we compared it to a child grasp in real life following Castner [14] study. Castner research on infant prehension categorized two main types of grasps; *palmar* and *pincer* (figure 2). The study reflects that palmar grasp is the most initial successful pose used by infants to perform new grasping actions.



(a) Palmar Grasp *(b) Pincer Grasp*
(Figures reproduced from Castner [14]*)*

Figure 2: *Grasping patterns of a child*

2.1.2 Hand-Object Interpenetrations

Interaction in the virtual world is often accompanied by interpenetrations between the hand model and virtual objects, due to lack of real physical constraints. This problem is usually resolved by making use of a "ghost hand" [26] [30] that is a duplicate of tracked hand but remains outside the grasped object at all times.

Zachmann [26] [31] explained heuristic contact based grasping system, having a minimization process to find a joint angle that keeps fingers from penetrating an object. Borst [32] proposed a physically realistic approach to grasping, addressing the penetrations problem visually. None of the above-mentioned methods take dynamics and control issues of the ghost hand into consideration. Our system provides visual feedback to avoid penetrations and also simulates dynamics between ghost hand and objects, thereby yielding stable control of the virtual hand.

Prachyabrued [33] suggested that the inert-penetrations are helpful in grasping and adding a constrained hand causes the user to misunderstand the grasp but Lindeman [34] negates Prachyabrued proposal showing how these penetrations result in instabilities in their system.

Our system goes along with Lindeman [34] proposal, as the penetrations affected our system performance negatively.

2.2 Interaction in Virtual Reality

Human-Computer Interaction (HCI) is a relatively mature research field but is one that continually benefits from the emergence of new interaction technologies [1] [5].

The need for realistic grasping and manipulation in virtual reality has always been obvious. Much research is in progress to apply digital hand to the virtual reality (VR) environment. The main purpose of this research is to have a more intuitive interaction possible in the virtual world so that user can directly manipulate virtual objects as naturally as possible [35].

2.2.1 Gesture-based Interaction

Gesture recognition is defined as the mathematical interpretation of a human motion by a computing device [4]. The most common way to manipulate objects and perform actions in the virtual world is through gestures.

Gesture interfaces are being implemented on a wide range of technical systems classified mainly into 2D and 3D gestures, where the former operates through hand movements on interactive surfaces while the latter operates through free-form movements in space [36].

Buchmann [37] introduced a gesture-based system called FingARtips, to directly manipulate objects by tracking marker fingers and identifying the gestures. Radkowski [38] presented an Augmented Reality (AR) assembly system using Kinect to interact with 3D models using finger gestures. Lee et al. [39] used a feature-based tracking system to identify hand pose and move the object around relative to hand movement. However, the above mentioned methods lack physical realism.

Pinch gesture is currently the most common gesture for interactions in the virtual world as it is relatively easy to implement. The pinch gesture sequence is shown in figure 3. The general idea is that if a fingertip and thumb are in contact, a pinch is detected, following this pinch an action is simulated for the nearest object until the pinch is released.

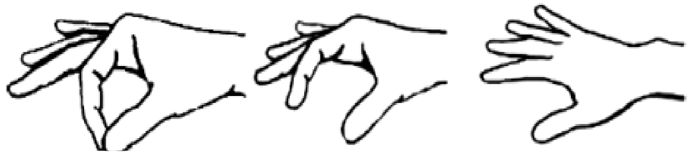


Figure 3: *Pinch Gesture Sequence*

Figure is taken from Jiménez [44]

Recently, Leap Motion [40] has announced the release of an Interaction Engine to establish the user intent, based on the stored information of hand movements, and help the user interact freely in the virtual environment. But the point to ponder is how it will reduce the gap between real world and virtual world interactions, thus, we have compared our methods with the available Leap Motion Pinch Gesture [13] [12] based grasping for this purpose. The method is used by Hyung-il [13], where an object is moved toward the hand once pinch gesture is detected for that object, but the possibility of making this grasp natural is still in discussion. Leap Motion's Interaction Engine offers to provide control based methods mostly and involves avatar forces to move objects around, like in Oculus wireless touch controller [41].

2.2.2 Physics-based Interaction

Physically-based simulation has been taking on an important role in various graphical applications such as computer animation, feature films, computer games, and virtual reality. Physics-based interaction considers hand as any other object in the physics simulation, including some constraints for odd hand deformations, so that hand can affect the physical objects in the virtual environment. Karen [42] has done noticeable work about physics-based interactions, including a physics-based optimization approach to synthesize feasible grasp using an initial hand grasping posture and a prescribed object motion whereas Zhao [11] build their system on physics-based simulation using pre-recorded motion sequences and transformed synthesized kinematic motion into physically realistic one [11]. Our approach is different because we build our system on physics-based simulation rather than physics-based optimization.

The identified issues with the physics-based approach are that due to differences in both environments and the lack of physical constraints in the virtual world, grabbing an object tightly can cause it to fly out of the hand as the fingers will penetrate through the object. The hand-object interpenetrations are explained in section 2.1.2. Therefore, we claim that physics alone is not sufficient to simulate stable interactions in the virtual world.

2.3 Hand Tracking Devices

Motion tracking methods can be divided into two types, i.e. incorporating a device that user must wear or hold and body-free methods [43]. The latter becomes more and more popular as it involves user as a controller instead of an operator. Jimenez [44] proposed a virtual classroom system and classified AR interaction as;

- Using markers
- Using wearable devices, like data gloves
- Using bare hands

One of the first widespread devices of the body-free kind was the Nintendo WiiMote Controller released in 2006 [43] [45]. Kinect by Microsoft is another milestone in the fields of robotics and human-computer interaction [46]. Several tracking options are possible using Kinect: skeleton, face and hand tracking within an accuracy of 1.3mm [47]. Recently, Oculus has announced the release of Oculus wireless touch controller [41], to be held in the hand and perform grasping tasks by pressing control buttons on the device, thus providing hand the control functionality.

Leap Motion controller was introduced, in 2012, as an affordable new technology claiming to be 200 times more sensitive than existing touch-free technologies. Its basic use is aimed towards hand tracking and gesture recognition. The smaller observation area and higher resolution of the device differentiate the product from Kinect, which is more suitable for whole-body tracking and requires relatively large space [48]. Sabir et al. [49] combined Leap Motion with Kinect in their research and were able to provide six degrees of freedom enhanced tracking. We selected Leap Motion as the starting point for our study because of its promising accuracy, mobility and distance interaction range. Initial tests conducted under regular interior lighting conditions proved the device efficient to be used as the tracking device for this research work.

III. SYSTEM OVERVIEW

This section briefly describes the commonly used grasp concepts in literature and the used contact model, assuming the particularly simple case in which all contacts between the fingers and objects are idealized as point contacts, followed by our design and approach.

3.1 System Architecture

Our system takes a human hand as input from the user for the simulated environment. The system block diagram is shown in figure 4, where simulated environment consists of virtual hand, proxy hand and a few virtual objects of various complexities. The system maps the input hand to a virtual hand model, which behaves only kinematically, and then creates a proxy hand that is coupled with the virtual hand and can provide dynamics to the virtual environment.

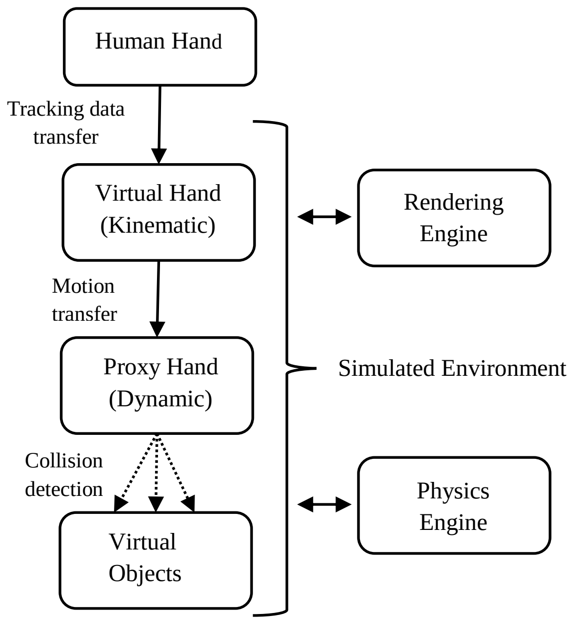


Figure 4: *System Architecture*

The human hand is the main driving factor in the system, and tracking device supplies driven data to the virtual hand in the simulated environment, thus the user can see his avatar hand motion in the virtual system. We have created another controlled proxy hand in the simulated environment which will be our main interactive hand as we will simulate dynamics for this proxy hand. This conceptual design is reminiscent of virtual coupling between a haptic handle and a virtual probe in haptic rendering [50]. We have defined an algorithm to control the motion of proxy hand depending on the collision states, as described in section 4.2. Moreover, in our virtual environment, we have many objects imported as 3D models having different complexities. The collision detection will be carried out between these virtual objects and the proxy hand.

Game engine Unity 3D is used for simulating virtual world because of its easy to use interface with the used tracking device; Leap Motion. Leap Motion provides well-established support for Unity 3D, allowing us to speed up the development and focus on improving the algorithms.

3.2 Grasp Statics

Typically, a Grasp is described using "friction contact" type and a "grasp matrix". The contact location is expressed by its relative position and orientation with respect to the object reference frame, i.e. $g_{oc} = (p_{oc}, R_{oc})$ where g_{oc} takes points from contact coordinates frame and return the coordinates in the object frame, shown geometrically in figure 5.

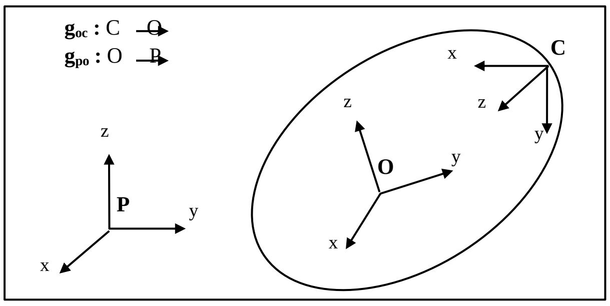


Figure 5: *Contact and object forces Coordinate Frames*

Grasp matrix is the mapping between the contact forces and the total object force. If we have k fingers in contact with an object, the total wrench on the object is the sum of the object wrenches due to each finger [9],

$$F_o = G_1 f_{c_1} + \cdots + G_k f_{c_k} = [G_1 \cdots G_k] \begin{bmatrix} f_{c_1} \\ \vdots \\ f_{c_k} \end{bmatrix} \qquad (1)$$

Where grasp map ‘G’ is the product of wrench transformation matrix, to map contact wrenches to object wrenches, and wrench basis ‘B_c’

$$G_i = Ad^T_{g^{-1}_{oc_i}} B_{c_i}$$

The “grasp pose” is a combination of the position and orientation of each vector i.e. a vector P = (x, y, z, α, β, γ) ∈ R. (x, y, z) is the object position and (α, β, γ) is the parameterization of its orientation.

We implemented our idea assuming friction point contacts, then calculated wrenches due to each finger in contact and the appropriate finger forces exerted on the grasped object to balance the external forces, using PhysX engine [51] [52] in our simulated environment.

3.2.1 Friction Point Contact

In simulation engine, by default, there exists frictionless point contact, where, contact forces can only be applied in the normal direction to the surface of the object. This type of contact can push an object but it cannot pull it.

The applied wrench can be represented as [9],

$$\mathrm{F}_{c_i} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} f_{c_i} \qquad f_{c_i} \geq 0$$

We can see the need for friction contact analysis here, thus, we implemented our contact as Coulomb Friction Model. The model states that "the allowed tangential force is proportional to the applied normal force" [9],

$$|f_t| \leq \mu f_n \qquad f_t \in R,\ f_n \in R \qquad (2)$$

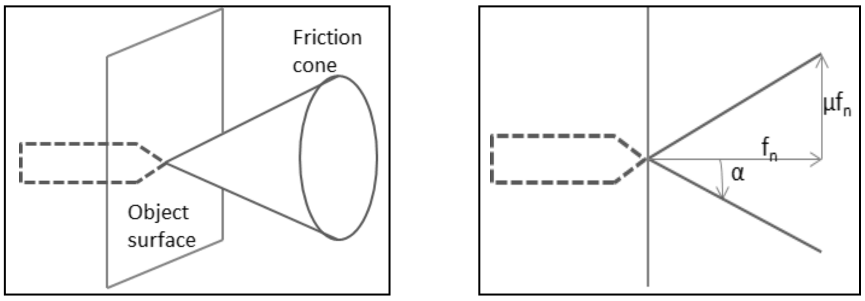
where, f_t is the magnitude of the tangential force, f_n is the magnitude of the normal force, and μ is the coefficient of friction and its value depends on the materials in contact. In this case, applied wrench can be represented as [9],

$$\mathrm{F}_{c_i} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} f_{c_i} \qquad f_{c_i} \in FC_{c_i}$$

where

$$FC_{c_i} = \left\{ f \in R^3 : \sqrt{f_1^2 + f_2^2} \leq \mu f_3, \qquad f_3 \geq 0 \right\}$$

This condition ensures that there will be no slipping as far as the applied forces lie in the friction cones.



(a) Front view (b) Side view

Figure 6: *Geometric interpretation of Coulomb's friction model*

Therefore, our model efficiently solves the frictionless problem by using friction cones around the contact points transmitting contact forces in normal and tangential directions making it possible to grasp the object firmly.

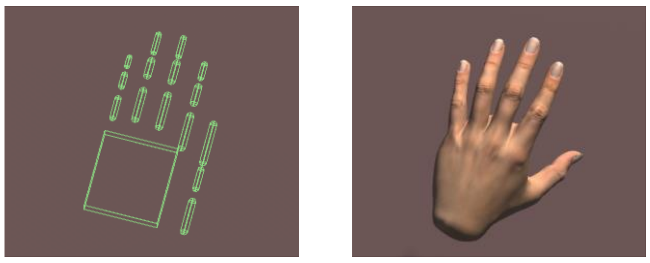
We implemented the frictional grasping by making use of the state-of-the-art physics engine, i.e. PhysX [51], to ensure that our system performs friction contact analysis for the contact points. This is achieved by enabling friction cones around the contact points for each established contact.

3.3 Leap Motion Controller

Using tracking device Leap Motion we can track physical hand in 3D space and interact with the intelligent devices in a completely new way replacing the old space taking the hardware. The hand tracking controller provides an approximately 150º field of view and uses a depth sensor to track hand features up to 1/100th of a millimeter which is sufficient enough for our requirement.

Leap Hand Controller is required to be added to the environment as an interface between the application and Leap software for the acquisition of data available through the Frame objects. Leap Motion Unity assets are equipped with a number of hand model prefabs, of two basic types as shown in figure 7,

1. Graphics hand
2. Physics hand



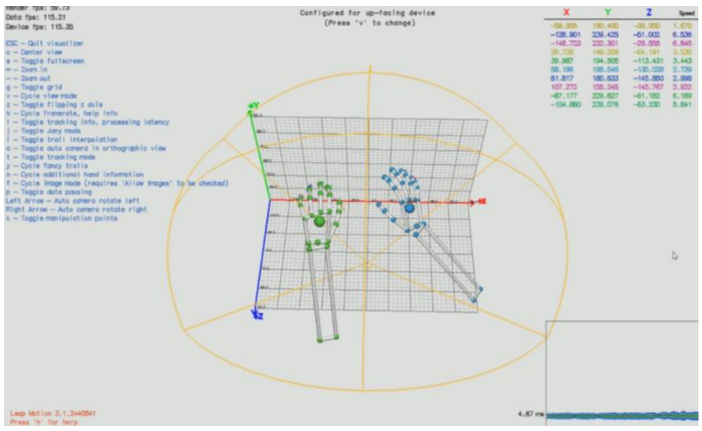
(a) Physics Hand *(b) Graphics Hand*

Figure 7: *Leap Motion hand models*

Leap Motion has provided the user with the different types of visualization of hand models including non-human and human models. The graphical hand is the virtual hand model visible on the screen while the physics hand is responsible for all the collision detections in the game engine.

The used game engine allows us to add a collider component to each object where the collision detection will be carried out between collider of the objects. Thus, we have used these colliders to simplify object's mesh and get better performance by avoiding mesh to mesh collision. Our system uses rigged-mesh based human graphical hands and capsule colliders as the base of our collision detection. There are three capsule colliders for each finger and a box collider for palm giving 16 colliders for a hand.

When the device is plugged in and turned on, it is constantly looking for any hand, finger or tool like objects within a certain area, commonly referred to as its field of view (FOV). The FOV provided is just 30 degrees short of being a perfect hemisphere, measuring in as a 150-degree area protruding directly from the device [52]. It can be seen clearly in figure 8, a screen capture of Leap Motion Diagnostic Visualizer, which allows developers to test Leap controller physical hardware.



```
Render fps: 59.73
Data fps: 115.21
Device fps: 115.35
```

ESC – Quit visualizer
c – Center view
s – Toggle fullscreen
= – Zoom in
- – Zoom out
g – Toggle grid
v – Cycle view mode
z – Toggle flipping z axis
h – Cycle framerate, help info
i – Toggle tracking info, processing latency
j – Toggle Jony mode
l – Toggle trail interpolation
o – Toggle auto camera in orthographic view
t – Toggle tracking mode
y – Cycle fancy trails
n – Cycle additional hand information
f – Cycle image mode (requires 'Allow Images' to be checked)
p – Toggle data pausing
Left Arrow – Auto camera rotate left
Right Arrow – Auto camera rotate right
k – Toggle manipulation points
Configured for up-facing device
(Press 'v' to change)
+Y
+X
+Z
Leap Motion 3.1.3+40841
Press 'h' for help
4.87 ms

```
     X          Y          Z        Speed
  -68.856    190.400    -35.980    1.670
 -128.901    239.425    -61.002    6.536
 -148.723    232.301    -29.556    6.845
   26.728    148.209    -64.191    3.526
   39.987    194.505   -113.431    3.443
   58.196    195.545   -135.008    2.739
   81.817    180.833   -145.850    2.998
  107.272    155.245   -145.787    2.922
  -87.177    229.827    -61.182    6.189
 -104.880    238.076    -63.230    5.841
```

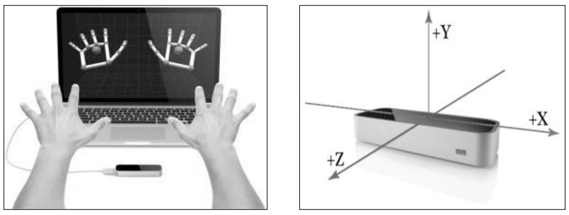
Figure 8: *Leap Motion Diagnostic Visualizer*

Interaction Area: 2 feet above the controller, by 2 feet wide on each side (150°), by 2 feet deep on each side (120°)

3.3.1 Coordinates Mapping

As we are dealing with different coordinate systems for hand tracking and physical simulation thus certain transformations are required for using such a tracking device as Leap Motion device in our game engine.

Leap Motion controller uses right-hand rule and it provides coordinates in terms of real world millimeters. As shown in figure 9(b), the Leap controller origin is located at the top, center of the hardware. However, Unity3D uses left handed coordinate system and provides data in meters. Unity itself provides transformations between world space, local space and screen space. We need to convert Leap data to one of unity coordinate systems to make the system compatible with unity frame of reference.



(a) Leap Setup *(b) Leap Coordinate frame*

Figure 9: *Leap Motion Controller*
Figure is taken from Leap Motion website [53]

Each hand bone collider (i.e. rigid-body) is parameterized using position and rotation vectors. Additionally, we have 3 colliders for each finger plus a palm collider giving 16 colliders in total to be able to perform contact dynamics. For each of these colliders, it is required to perform translations to map their coordinated in world space. The usage of basic mapping extensions is shown below to convert leap data to unity and world frame [53].

```
Leap.Vector position = finger.TipPosition;
Vector3 unityPosition = position.ToUnityScaled(false);
Vector3 worldPosition = handController.transform.TransformPoint(unityPosition);
```

```
Quaternion leapRotation = arm.Basis.Rotation(false);
Quaternion finalRotation = handcontroller.transform.rotation * leapRotation;
```

IV. VIRTUAL GRASPING ALGORITHMS

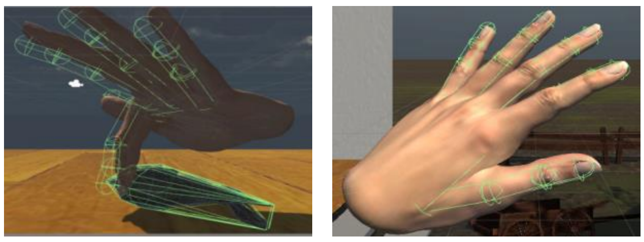
This chapter introduces basic collision states and visual feedback methods involved in our system, followed by two main virtual grasping methods proposed in this work;

1. Physics-based grasp
2. Assistive grasp

4.1 Visual Feedback

We optimize Leap Motion virtual hands to meet our visual needs that include rendering both hand models i.e. graphical and physical, as close to each other as possible and providing ways to help the user with an effective grasp possibility. Also, several tests are carried out to find best suitable physics engine settings i.e. time-step and iteration count.

To have real-time simulation performance, we do not use the rigged graphical mesh model as physics models (or colliders); instead, we use its simplified shapes (called primitives). To do so, we align physics models and rigged graphics models for the virtual hand to optimize as close to the hand mesh as possible using primitive shape colliders, by editing bones positions and rotations. Figure 10 shows the difference between before (left image) and after (right image) proper alignment.



(a) Not Aligned *(b) Aligned*

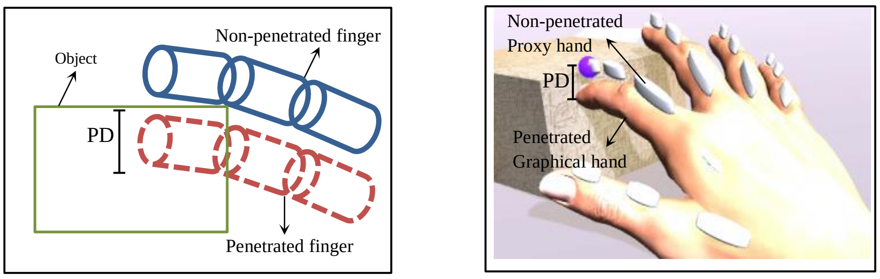
***Figure 10**: Physics hand and graphical hand alignment*

Our system generates virtual markers at run time for contact points between virtual hands and objects to give the user a visual idea of the contact locations (see figure 15). The markers are created whenever system encounters a CollisionEnter (C) event and destroyed at CollisionExit (C^{-}) event (described in section 4.2). We believe that this way assists the system with a feasible grasp possibility as the user can see the contact points clearly.

4.1.1 Proxy Hand

Tracking data used as input to the virtual hand may make it difficult to select the contacts between the object and the hand for grasping firmly. The resulting problem is that the finger colliders keep penetrating inside the object as there is no closed loop control involved between the user hand and virtual hand and the underlying physics engine cannot resolve this problem because colliders follow the data from the tracking device without considering instabilities in the simulation.

We address this issue as penetration depth (PD) problem. PD is the minimum distance required to separate two penetrating/colliding objects. Penetration resolution is itself a huge field in physically-based animation and robotics. In our case we faced penetrations because of the one-way coupling between Leap Motion and the driven kinematic hand; i.e. only Leap Motion drives the motion of the virtual hand, but not vice versa. We resolved this issue by creating another virtual proxy hand driven by the virtual hand. Thus, we resolved penetrations by adding dynamics to the proxy hand to have a feedback loop response for the system. This way we make sure that our system will not go into the deep penetration state and it will avoid instability.



(a) Geometrical *(b) Graphical*

Figure 11: *Penetration situations for virtual hand and proxy hand*

It is important to understand the provided hand model structure in detail to be able to access the used data structures. The skeletal hand used as a base for all models is same (figure 12).

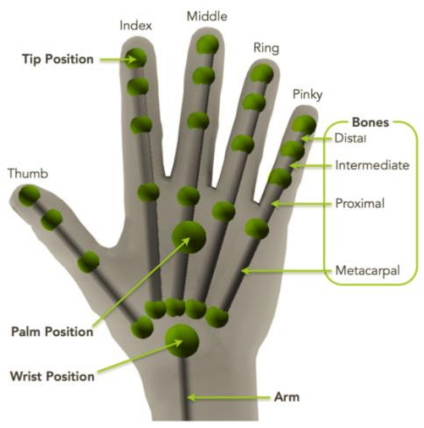
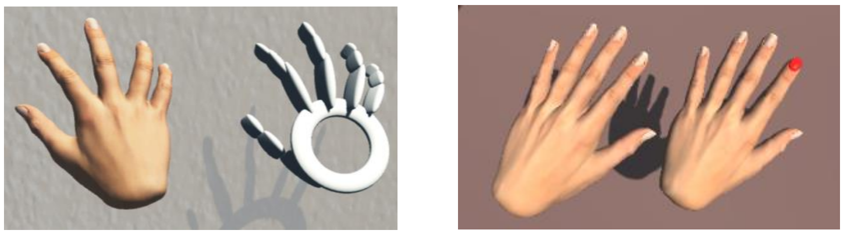


Figure 12: *Leap Motion skeletal hand taxonomy*

Figure is taken from Leap Motion official blog [54]

To provide dynamics to hand, we map the virtual hand model to our proxy hands; a rigid non-human hand and a surface human hand. Figure 13 is showing the original leap virtual hand and our proxy hand which can be controlled by leap data or as per required. The proxy hand creation process involves transferring leap hand motion to proxy hand completely by performing all the necessary coordinate transformations, as described in section 3.3.1.

In the next stage, we created a surface model of the proxy hand. It involves mapping between bones rotations only. In this case, it is not possible to deal each skeletal bone individually as doing so will deform the surface model at wrong angles and it requires inverse kinematics. In this way, we have provided a control solution, in the form of dynamic proxy hand, for hand tracking devices which only provide one-way coupling between the device and the user.



(a) Leap hand and bone proxy hand

(b) Leap hand and surface proxy hand (red nail as an indication)

Figure 13: *Leap hands and corresponding proxy hands*

4.2 Collision States

Given an object to be grasped, we have divided our kinematic motion into four basic states, as shown in figure 14 and listed below;

1. C^0: collision-free phase
2. C: collision-enter phase
3. C^+: collision-stay phase
4. C^-: collision-exit phase

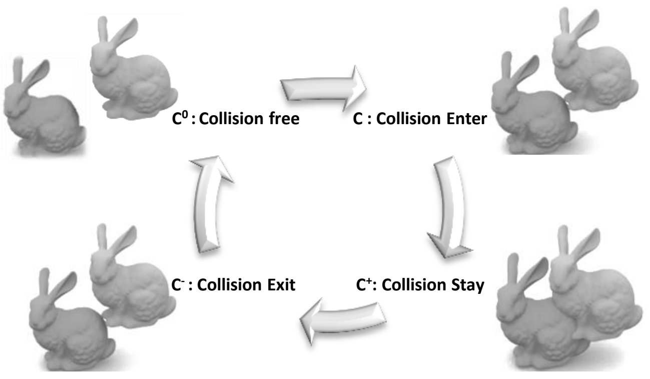
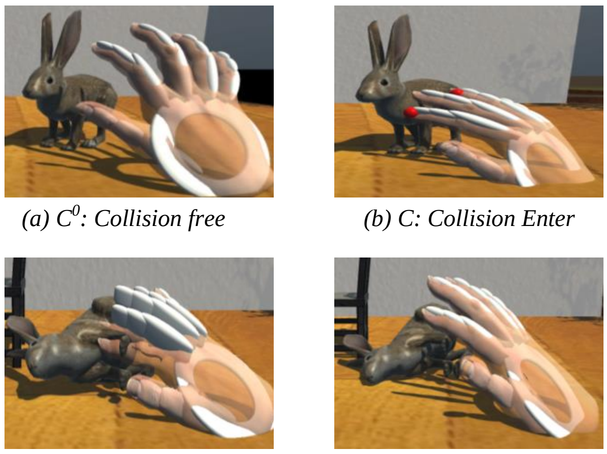


Figure 14: *System Collision States*

The collision-free state is where objects are separated. In the next frame, objects are in contact, we called this state as collision-enter state. Collision-stay phase is for all the frames in which objects are still in contact and are overlapping. The shortest translation required to separate such objects is called the *penetration depth* to have a collision-exit state.

In our designed system, during collision-free state proxy hand is following the virtual hand. In the case of a collision event, the system will report a collision-enter state and in the next frame as soon as the system goes into the collision-stay phase, the corresponding colliders of proxy hand will stop following the virtual hand to be independent and controllable so as to avoid penetrations until the collision ends and colliders separate. The captured screenshots to illustrate our system states are shown in figure 15.


(a) C^0: Collision free (b) C: Collision Enter

(c) C^+: Collision Stay (d) C^-: Collision Exit

Figure 15: *Collision states for proxy hand*

4.3 Physics-based Grasping

Our first approach uses a physics-based simulation to simulate interactions between hand and objects, considering hand as a kinematic physics object. This method enables the user to interact freely with the object and by ensuring friction point contact it is possible to achieve a grasp ensuring a careful touch.

We have named fingers as thumb, index, middle, ring and pinky. The total wrench on the object will be the sum of individual wrenches by each finger, using the equation (1),

$$F_{total} = G_1 f_{c_{thumb}} + G_2 f_{c_{index}} + G_3 f_{c_{middle}} + G_4 f_{c_{ring}} + G_5 f_{c_{pinky}}$$

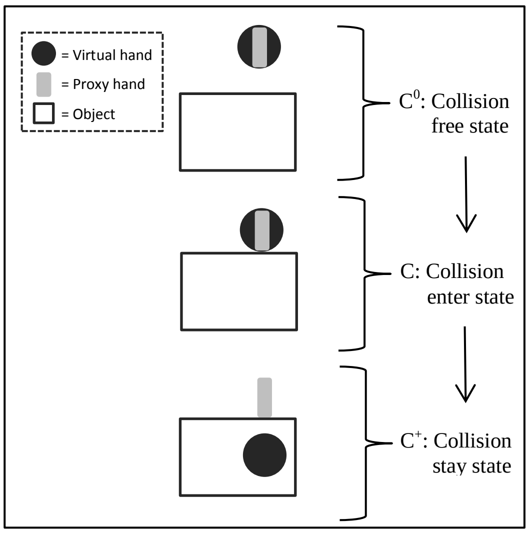
For a stable grasp, at least three fingers should be in contact with the object, depending on the object size and weight. To make sure that our system will use friction cones as explained in equation (2), we use the friction settings of physics engine in Unity3D as,

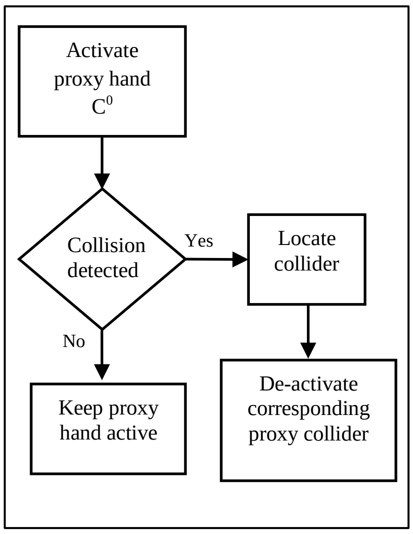
```
object.useConeFriction = true;
```

4.3.1 Algorithm

Given an object to be grasped, we have divided our kinematic motion state into C^0: no-collision phase, C: collision enter phase, C^+: collision stay phase, and C^-: collision exit phase. The usual robotics denomination of 'free finger' and 'grasping finger' is used here. A free finger is constrained to have no contact with the object whereas a grasping finger is in contact with the object surface [17].

As it is not possible to control movements of virtual hand independently from real hand, we used the already created proxy hand. Our proxy hand will follow tracking hand movements at the times when there is no collision but as soon the virtual hand collides with the object it will create a collision enter event to deactivate proxy hand, until the collision event exits. The basic flow chart of our grasping algorithm is shown in figure 16.





(a) Illustration *(b) Flow Chart*

Figure 16: *Physics-based grasping algorithm for proxy hand*

Figure 16(a) shows an illustration of the difference between proxy and real hand situations. The algorithm will keep proxy hand away from penetrating inside the object at all times in the simulation. But there is a limitation because of game engine physics which require hand motion to be slow, otherwise, the system might miss collision event because of discrete collision detection method.

Unity 3D allows continuous collision detection against static mesh colliders only, meaning the object, with static mesh collider, lacks rigid body component and will not be affected by physics. However, for our simulated environment we have used all objects as rigid bodies and are affected by physics, thus, the physics engine is using discrete collision detection against these objects. Collision event time steps can be decreased at the cost of system performance.

4.3.2 Results

We believe that our proposed algorithm can provide physically realistic grasping dynamically using the hand tracking device. We experimented the grasping at first without proxy hand and it results in penetrations making the system unstable.

Figure 17 is showing results without resolving penetrations. It can be seen that hand penetrates deeply into the object which results in instability. The object either sticks to the hand and user have to jerk it off or in some case object slips from the hand.

.

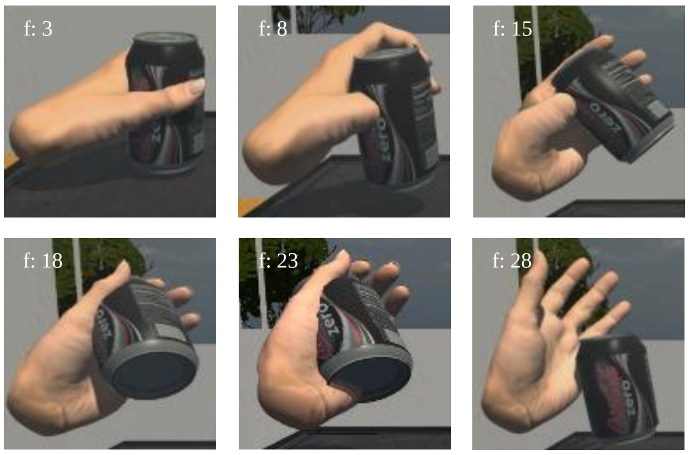


***Figure 17**: Grasping sequence without proxy hand, showing penetrations*

Captured at different fames showing holding, moving and releasing the object (top left to bottom right)

Thus, it becomes necessary to resolve deep penetrations to have stable interactions. As in this system, the used physics engine failed to spot and resolve deep penetration, so we have used the concept of proxy object for this purpose. Figure 18 is showing the results using our proxy hand approach; the difference is visible clearly as this approach resolves penetrations making the results more stable and visibly correct.

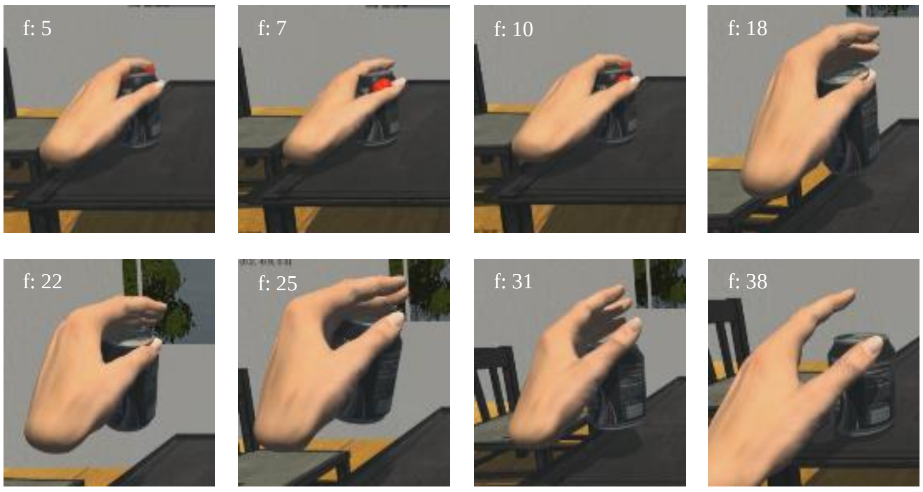


Figure 18: *Grasping sequence with proxy hand, resolving penetrations*

Captured at different fames showing holding, moving and releasing the object (top left to bottom right)

4.4 Assistive Grasping

This section provides our assistive palmar grasp algorithm where we have planned our system in a way to predict user's grasping intuition and assist with the grasp. Our assistive algorithm is an improvement to the basic pinch mechanism provided by Leap Motion [55] [13] and also to the physics-based method introduced in the earlier section as physics simulation alone is still not sufficient to have an efficient grasp.

As compared to pinch gesture grasp, our method provides a more realistic grasp pose. Depending on the distance between thumb and fingers, we defined our grasp strength and accordingly once the grasping gesture is detected, the object will move a relatively small distance towards the palm of the hand by applying a force in the direction of the hand with enough magnitude to move the distance between the hand and the object. This makes the positional vector of the object same as of hand palm.

4.4.1 Algorithm

Among the basic requirements in a virtual-human grasp system, is the ability to produce natural-looking grasps with sufficient user control. This method is based on power grasp, where grasp pose involves the palm and inner surface of the fingers. We have provided assistive planning, as planning refers to testing with different hand positions and orientations relative to the object, once a contact is established between hand and object, the fingers of the hand close around the object until they cannot move anymore.

We used this concept to automate the grasping process in our system by detecting the first step that is when an object is detected as grab-able and then allowing the user to close the fingers around the object to grab it. We utilize proxy hand to force fingers not to pass through the object. A few terminologies used in the algorithm are explained below [52] [56];

Grab Strength (GS): A float value in the range of 0 to 1, representing 0 for an open hand and 1 for a closed hand. It indicates how close is the hand to being a fist, any finger not curled will reduce its value

Overlap Sphere: An imaginary sphere centered at a *position (hand palm)* with a specific *radius* and returns an array with all the objects touching or inside the sphere.

Grab-able: A variable to store the closest object to the hand palm returned by Overlap Sphere.

We have divided the algorithm into grabbing and releasing stage explained below;

Grabbing:

The first step is to initialize an empty object called 'grab-able' to keep track of all the objects possible to grab. Next, we need to find the closest object to the hand, for this purpose we cast an imaginary overlap sphere around the hand with the hand palm as a center and a specific radius and identified the object inside this sphere and closest to the hand as 'grab-able', shown in figure 19. The next step is to identify the user grasp pose, to do so we checked the distance between all fingers to the thumb, called as "grab strength". If the grab strength is sufficient enough it means the user is trying to close the fingers around the grab-able object. Here, we made system to help the user pick up the object by applying gentle forces to the object to move it to the hand palm and be attached there unless the releasing sequence is detected.

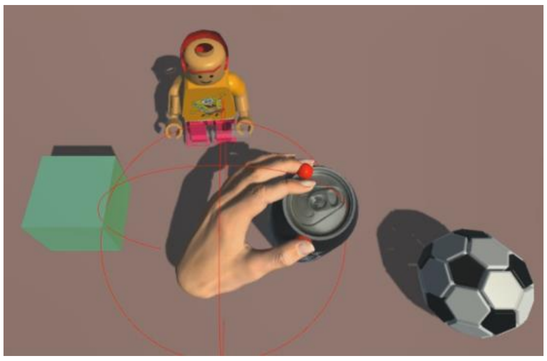


Figure 19: *Overlap sphere with hand palm as center*

Releasing:

To detect if the user is releasing an object, we kept track of the previous condition, that is, if the grab-able object is not null, meaning something is already in hand and if grab strength reduces; meaning the user is opening his hand. At such a situation the object will be de-attached from the hand palm and hence affected by gravity will move freely.

Figure 20 shows the algorithm in the form of a flowchart.

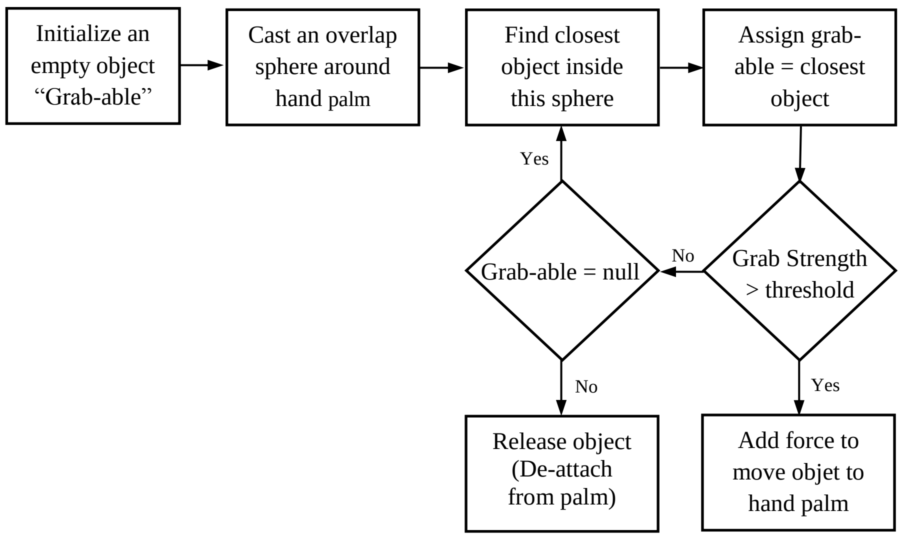
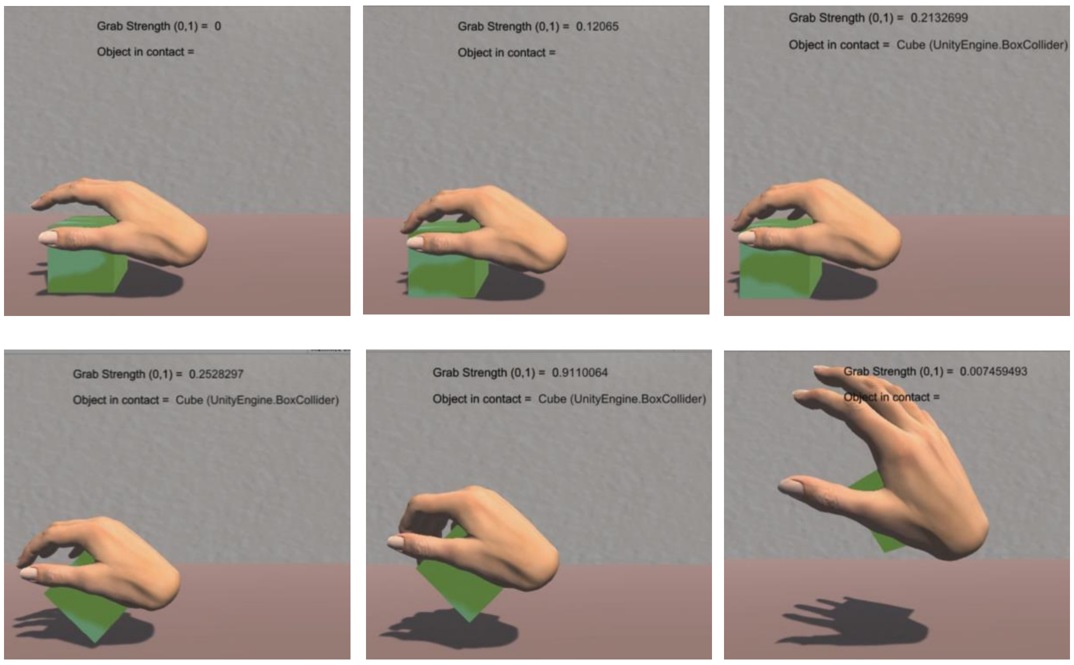

Initialize an empty object "Grab-able"
Cast an overlap sphere around hand palm
Find closest object inside this sphere
Assign grab-able = closest object
Yes
Grab-able = null
No
Grab Strength > threshold
No
Yes
Release object (De-attach from palm)
Add force to move objet to hand palm

Figure 20: *Assistive grasping algorithm*

4.4.2 Results

Figure 21 shows the real-time experiment of algorithm steps as described above. The hand model has always an imaginary sphere surrounding it (figure 19) with hand palm as a center, once an object is detected inside this sphere grab strength value is monitored to know the user intention. On detecting user hand pose as a grasping pose, the object is moved a small distance toward the hand, this distance is assumed to be un-noticeable in the simulation after that user can close the fingers to achieve a properly visible grasp. The virtual hand visible in the simulated environment is the hand as discussed before.

The grasping result for a 3D Lego model is shown in figure 22. The model consists of 15 separate convex colliders to simplify the shape of the object, and in total 2913 triangles, each collider had less than 256 triangles. The assigned goal here is to pick up the model and move it into the box; the grasp sequence is shown in figure 22 with the calculated grasp strength (GS) values.



```
Grab Strength (0,1) = 0
Object in contact =
```

```
Grab Strength (0,1) = 0.12065
Object in contact =
```

```
Grab Strength (0,1) = 0.2132699
Object in contact = Cube (UnityEngine.BoxCollider)
```

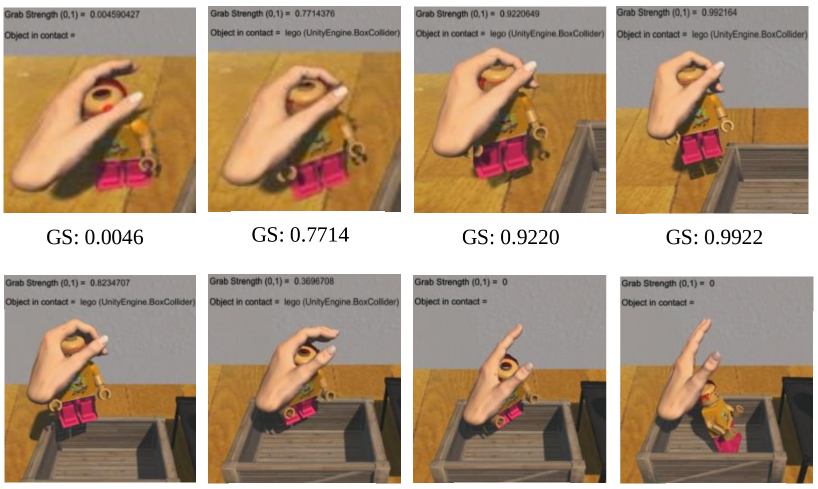
```
Grab Strength (0,1) = 0.2528297
Object in contact = Cube (UnityEngine.BoxCollider)
```

```
Grab Strength (0,1) = 0.9110064
Object in contact = Cube (UnityEngine.BoxCollider)
```

```
Grab Strength (0,1) = 0.007459493
Object in contact =
```

Figure 21: *Assistive grasping sequence*

In order from top left to bottom right, notice that in step 3 grab strength meets the threshold and object is detected as grabble. In step 4 object changes its orientation to move into the palm of the hand, followed by step 5 where user is moving the object until grab strength decreases and object is released from the hand in step 6.



```
Grab Strength (0,1) = 0.004590427
Object in contact =
```

```
Grab Strength (0,1) = 0.7714376
Object in contact = lego (UnityEngine.BoxCollider)
```

```
Grab Strength (0,1) = 0.9220649
Object in contact = lego (UnityEngine.BoxCollider)
```

```
Grab Strength (0,1) = 0.992164
Object in contact = lego (UnityEngine.BoxCollider)
```

```
Grab Strength (0,1) = 0.8234707
Object in contact = lego (UnityEngine.BoxCollider)
```

```
Grab Strength (0,1) = 0.3696708
Object in contact = lego (UnityEngine.BoxCollider)
```

```
Grab Strength (0,1) = 0
Object in contact =
```

```
Grab Strength (0,1) = 0
Object in contact =
```

Figure 22: *Assistive grasp results for a Lego model*

V. IMPLEMENTATION AND EXPERIMENTS

In previous chapters, we have introduced two relevant approaches to simulate grasping in virtual reality. In this chapter, we present used benchmarks and the results achieved by a conducted user study.

5.1 Implementation Platform

We have implemented our methods and conducted experiments on x64 Intel TM i7-4790 CPU @ 3.60 GHz with 16.0 GB RAM. The display screen used has 13.3” x 8.3” (width x height) dimensions and 1280 x 800 resolution.

Two main tools used are the hand tracking device; Leap Motion and game development engine; Unity3D. CPU utilization by the system is less than 25% at all times. The algorithms are written in C# as scripts in the used game engine.

5.1.1 Unity3D Game Engine

We have chosen the unity game engine for design, rendering and implementing physics in our system. Unity allows game designers, artists, and developers to create games and simulators with little difficulty. The physics engine used by unity is NVIDIA PhysX. Leap Motion has provided its plugin [40] for Unity thus making the integration process relatively simple. Once the controller is inside the unity scene, the next task is to make interactions possible between the physics hand and scene game objects.

Unity3D allows the user to add rigid body and assign suitable collider explicitly to any game object enabling the collision detection possible within the virtual world. Our method runs well at solver iteration count of 50 while contact offset and sleep threshold for rigid bodies are set to default PhysX values.

5.2 Benchmarks

We have used different 3D objects for experiment purposes with different complexities, as shown in figure 23. Also, we have tested the system with primitive shapes like cube, sphere, and cylinder.

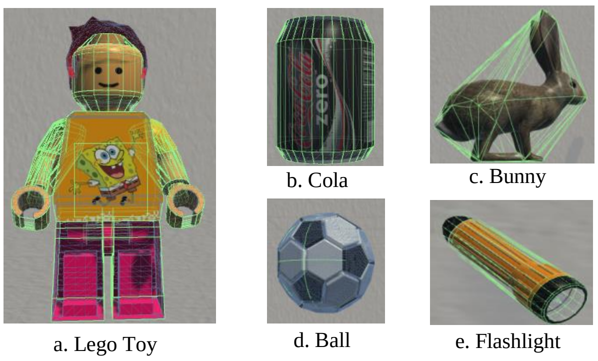


Figure 23: *3D Benchmarking Models*

Different models have different type of complexities to check system behavior. The ball is a spherical object having less friction and more torque and thus requires precise contacts to avoid slipping. Lego model is a combination of different objects and can be treated as separate colliders thus, even grabbing a part of the model like arm or leg can give a stable grasp. Bunny has a complex mesh so its collider is reduced to 255 triangles, but it is an important benchmark to check system behavior with complex meshes. The flashlight also has separate colliders linked together and is relatively thinner than other objects. The complexities of the benchmarking models are summarized in Table 1.

Table 1: *Model complexities of benchmarking models*

Models	**Tris**	**Links**	**Collider**	
			Type	**Tris**
Lego	9114	15	Mesh	2113
Cola	1020	1	Mesh	255
Bunny	1910	1	Mesh	255
Ball	32220	1	Sphere	-
Flashlight	1508	5	Mesh	798

Some of the models have mesh colliders attached but the used game engine limitation for a rigid body mesh collider is that the mesh will be created as convex and with only 255 triangles, thus, the used collider is an approximation of the original shape of the 3D model. We believe that an exact formulation of mesh collider will result in significant improvement with respect to the collision detection but at the cost of system performance.

5.3 User Study

A user study is conducted to check system improvement against the existing pinch gesture-based grabbing. Also, different aspects of the system are analyzed by asking survey questions from the users about system efficiency in terms of achieving the assigned task and personal like/dislike. The user study is conducted with 14 female subjects, consisting of 8 VR experienced users and 6 naïve users. All the users are Ewha Womans University female students, between 20 to 38 years old. 4 users have tried the used tracking device; Leap Motion, before to interact with the virtual world but none of the users were experienced enough to plan interaction before head.

We have used three objects from the benchmarks for user testing having different complexities; Lego toy model, Ball, and Cola can. The three systems tested are;

1) System A: Physics-based grasp
2) System B: Assistive grasp
3) System C: Magnetic pinch grasp

System A and B are compared against system C which is the official grabbing system introduced by Leap Motion.

5.3.1 Experimental Setup

Experiments are carried out in normal internal lighting conditions and sitting position for the user. Prior to interactions, participants are briefed about the goal of the study and each participant is given 90 seconds to try out each system before recording the results. In this training session, each subject is told to try both left and right hands to pick up the objects near each hand and move the objects into the box one by one, also to be slow and try to be precise in the hands movement.

Once the subject is comfortable with the environment, the time is noted once an object is in contact with the hand till it is moved into the box for each object in each system. Maximum of 30 seconds is given for each object, after which it is counted as a failed grasp. The order of systems is randomized for each participant to avoid any previous knowledge of the methods. The objects are reset for the user if it goes at an unreachable distance as a result of interactions, assuming it as a limitation of tracking device and is not counted as a fail in our experiment. Moreover, if the system loses tracking, it is also counted as tracking device limitation and will not affect the data collected.

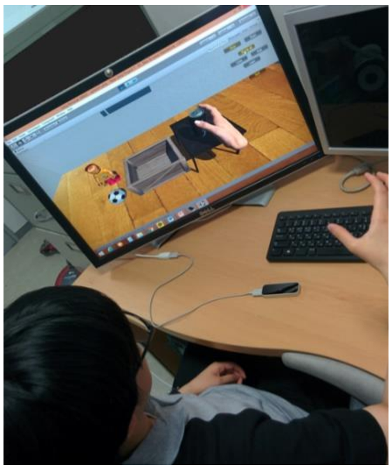


Figure 24: *A subject of our user study trying the setup*

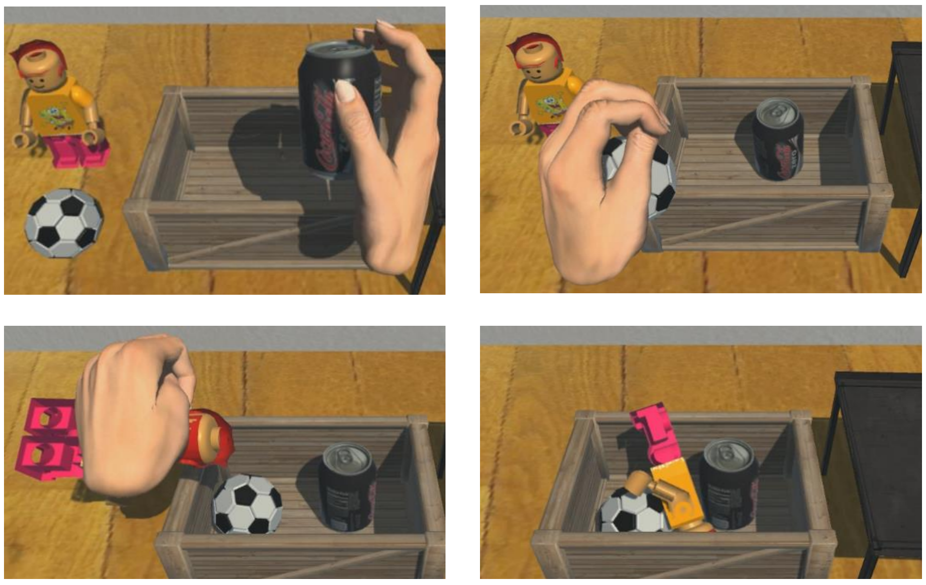


Figure 25: *Images of a user achieving the goal*

Moving cola, ball and lego model into the box (in order from top left to bottom right)

5.3.2 Analysis

Different aspects of the system are analyzed including time spent to achieve the assigned task, personal like/ dislike, learning rate and system efficiency in terms of success/fail to achieve the task. Participants are asked to rank the three systems based on four criteria;

1) Liking: Which one did you prefer?
2) Naturalness: Which one looked more natural?
3) Practicality: Which one was easier to take?
4) Efficiency: Which one was efficient?

Liking refers to the personal taste of each user that what type of system they would prefer to see in the future virtual reality. Naturalness is defined in comparison to real world grasping. Practicality is the system difficulty level and efficiency is identified in terms of achieving the assigned task. figure 26 shows the results in the form of the pie chart for each criterion. Our proposed assistive palmar grasp came out as better than the existing system, but the exception can be seen in the figure below that user preferred our physics-based system in terms of naturalness as it is most close to the real life actions.

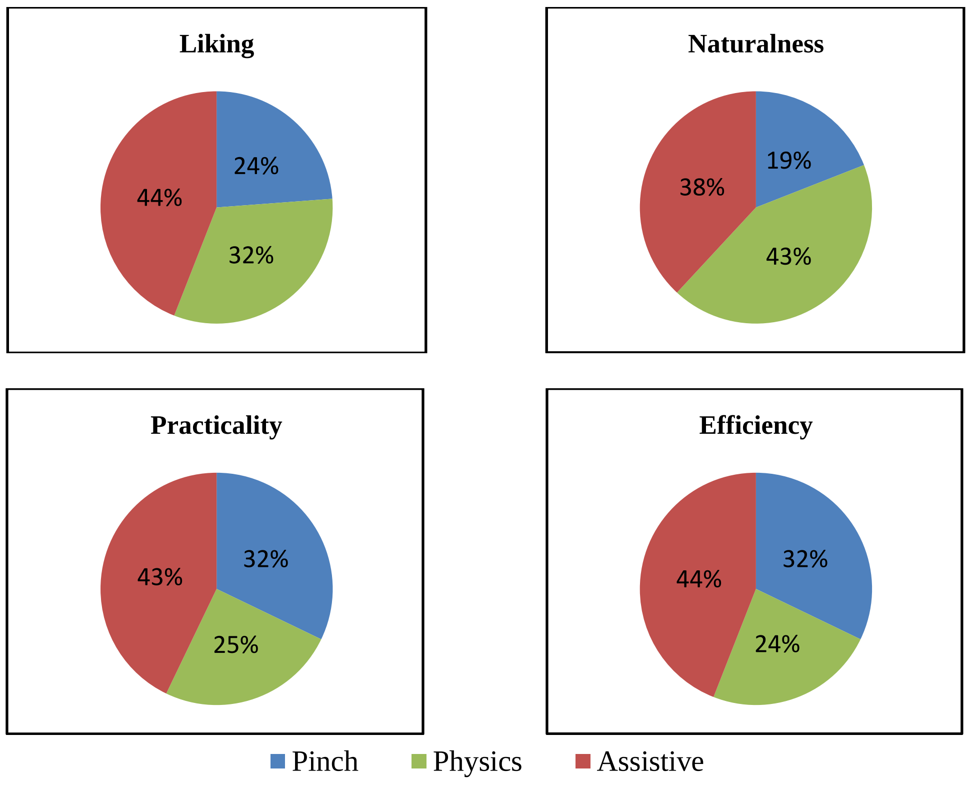


Figure 26: *Comparison of three approaches based on survey questions*

Grasping is carried out in an environment unknown to the user so we compare it to an infant grasp as infants are trying out grasp in an environment unknown to them and the earliest research by Castner [14] showed that palmar grasp gives best results for them. Our system performed in support of this research. The data collected from each individual subject is shown in table 2. The experiment result showed that the assistive palmar grasp works better than both of other methods, as expected.

Physics-based approach performed better in terms of being closest to real life grasp sequence but as virtual environment is not expected to work the same as the real environment, thus although it is possible to grasp objects only by physics rules in the virtual world but it takes more time and training to be able to grasp properly.

Magnetic pinch grasp is an un-natural grasp as once a pinch gesture is detected, the object is moved towards the pinch position, and object seems to be hanging to the thumb. It is not actual grasp pose, but a gesture based pick. We noticed that none of the subjects tried to grasp objects like this way as it is completely unnatural to the human brain.

Assistive grasp gave most stable and efficient results as the system here helps the user by attaching the object to the hand palm, once a feasible grasp pose is detected.

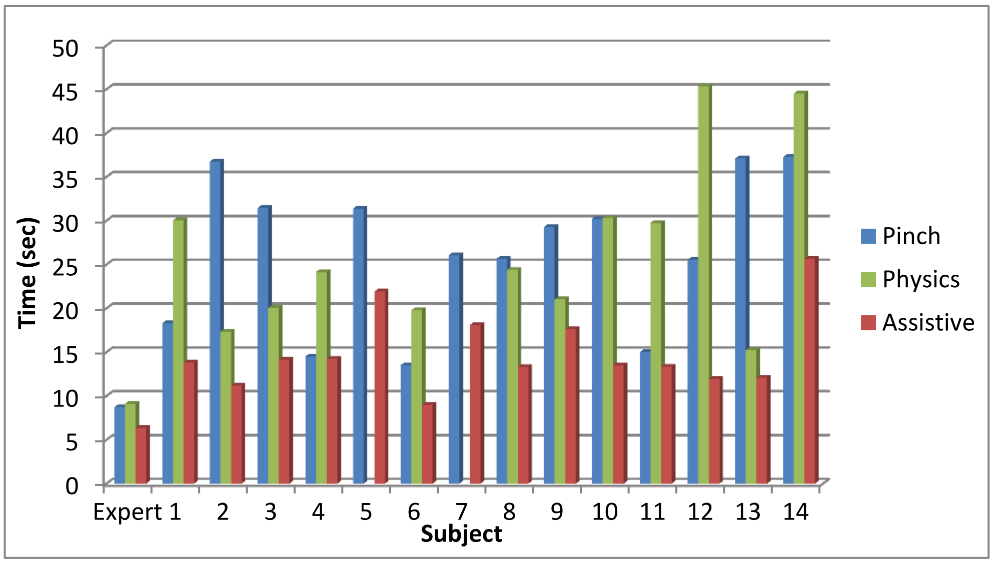


Figure 27: *Comparison of three approached based on user data showing time spent to achieve the assigned goal of moving three objects into a box*

Table 2: *Results of the user study*

The data is collected from a group of 14 students. The subjects are asked to move three benchmarking models into a box and the time spent is recorded in seconds

Subject	**Magnetic Pinch Grasp**				**Physics-based Grasp**				**Assistive Palmar Grasp**			
	Lego	**Ball**	**Cola**	**Total**	**Lego**	**Ball**	**Cola**	**Total**	**Lego**	**Ball**	**Cola**	**Total**
Expert	*2.82*	*3.02*	*2.89*	*8.73*	*2.28*	*3.78*	*3.05*	*9.11*	*1.15*	*2.87*	*2.35*	*6.37*
1	7.07	6.18	5.05	18.3	14.85	9.18	6	30.03	5.05	4.28	4.46	13.79
2	9.82	11.73	15.2	36.75	8.76	4.7	3.89	17.35	3.45	2.73	5.01	11.19
3	8.28	9.12	14.06	31.46	4.73	6.21	9.12	20.06	3.03	5	6.12	14.15
4	5.57	5.07	3.87	14.51	9.57	3.63	10.95	24.15	5.22	5.43	3.57	14.22
5	15.81	10.63	4.95	31.39	-	4.9	1.63	-	11.12	6.97	3.82	21.91
6	3.98	4.27	5.28	13.53	4.3	8.91	6.59	19.8	2.99	2.84	3.14	8.97
7	8.71	7.52	9.82	26.05	-	6.57	8.03	-	7.51	5.25	5.33	18.09
8	11.62	7.74	6.29	25.65	10.23	7.65	6.52	24.4	5.28	4.36	3.68	13.32
9	12.08	8.54	8.68	29.3	8.23	7.58	5.22	21.03	7.13	7.05	3.44	17.62
10	11.95	11.21	7.05	30.21	10.97	11.68	7.66	30.31	6.89	3.65	2.99	13.53
11	6.23	4.56	4.21	15	10.14	8.07	11.54	29.75	7.29	3.78	2.26	13.33
12	13.51	6.6	5.43	25.54	18.41	15.46	11.47	45.34	4.77	1.59	5.61	11.97
13	5.94	9.35	21.83	37.12	3.35	4.27	7.62	15.24	4.63	1.77	5.71	12.11
14	20.08	10.93	6.25	37.26	26.29	9.35	8.87	44.51	15.32	6.45	3.87	25.64
Average				25.38667				25.46769				**14.414**

We asked three random subjects to try the assistive grasp multiple times to check the method learning rate. The goal was to check how fast the user can be familiarized with the method without any prior knowledge of the algorithm at the back. The results were encouraging, as each user happens to complete the task in less time than the previous attempt. The learning curve is shown in figure 28; the time shown is the total computed time to move three objects into the box. The detailed timings are shown in table 3. The result shows that user can learn the system and adapt grasp gesture better with experience.

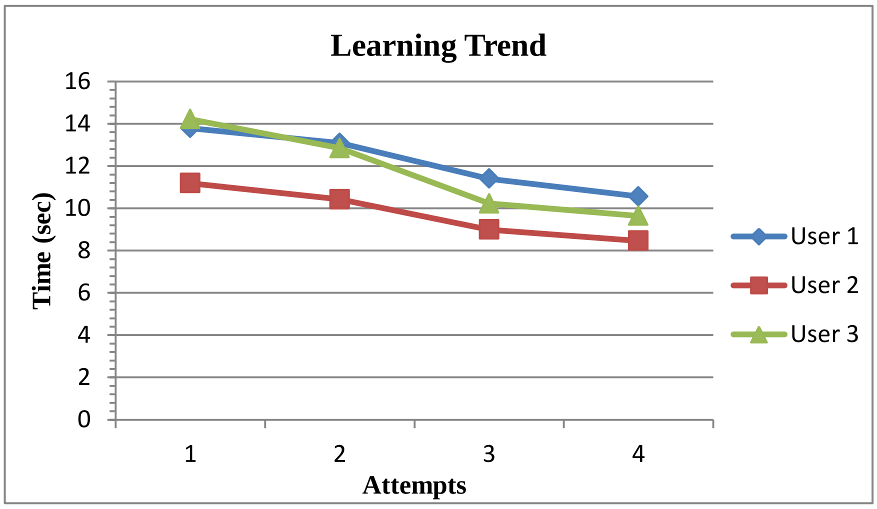


Figure 28: *Learning curve for assistive grasp*

Table 3: *User data of learning curve for assistive grasp*
(Time spent to complete the task in seconds)

User	Attempt	Lego	Ball	Cola	Total
1	1	3.45	2.73	5.01	11.19
	2	3.4	2.56	4.46	10.42
	3	3.12	1.72	4.15	8.99
	4	2.87	1.7	3.89	8.46
2	1	5.05	4.28	4.46	13.79
	2	4.85	3.89	4.35	13.09
	3	4.09	3.49	3.82	11.4
	4	3.71	3.33	3.52	10.56
3	1	5.22	5.43	3.57	14.22
	2	5.19	4.62	3.04	12.85
	3	4.92	3.02	2.29	10.23
	4	4.65	2.76	2.23	9.64

VI. CONCLUSION

In this dissertation, we have addressed the issues to achieve real-time grasp possibilities in virtual reality. We have implemented and demonstrated two methods in challenging benchmarking scenarios and compared our method with existing state-of-art pinch grasping. With these approaches, we give new and effective interaction possibilities in a simulated environment. The resulting system is an interactive grasping system, which provides visual feedback and control solutions for one-way coupling between hands and tracking devices. We believe that our proposed algorithm is a step towards providing physically realistic grasping using kinematic hand models provided by usual tracking devices.

The past research in the related fields of grasp planning, virtual reality, and motion tracking methods mostly relies on motion sequence databases with huge software and hardware requirements. Very little research exists about real-time physically realistic interactions in the virtual domain. Our goal was to achieve real-time interactive performance in virtual reality, as close to the real world as possible, and to be able to grasp digital objects.

We address two important issues encountered in simulations i.e. frictionless contact and hand-object interpenetrations. We resolved the frictionless problem by including Coulomb's friction point contacts in our system and introduced how to use a proxy hand to resolve hand-object inter-penetrations and simulate dynamics in the system. To do so we mapped virtual hand’s joints velocities and positions to a proxy hand and carried the simulations between proxy hand and objects so we can detect collision events and force the proxy hand to avoid penetrations.

Our system consists of an off-the-shelf hand tracking device and a game development engine, performing simulations in a regular desktop environment. We have presented two grasping algorithms; physics-based grasp and assistive palmar grasp. The physics-based approach makes sure to provide friction point contacts to be able to grip object firmly, and assistive approach makes system aid in carrying out the grasp by detecting user intention and adjusting object’s position and orientation in hand palm.

We conducted user experiments to test the algorithms against existing magnetic-pinch grasp. We tested the system with both VR experienced and un-experienced users to see how naïve users reach out for virtual objects. Our assistive grasping method works well for all users, in terms of achieving the goal in less time and aiding the user in achieving a feasible grasp.

The ultimate goal would be for people to interact with the virtual domain as easily as with the real world but simulating physically realistic dynamics in the virtual domain is still a complex problem. We have provided a control solution for one-way hand-object coupling but our system has certain limitations. The user is required to perform light touch on the object as the user can not feel the object, providing force analysis is not possible without physical feedback to the user. We have provided virtual feedback by contact markers but it is needed to utilize haptic response to have better interactions. Moreover, the system is dependent on off-the-shelf game engine and thus accompanied by certain limitations as continuous collision detection is only supported for static objects in the used game engine and in discrete collision detection case, the system can miss collision events and goes into penetrations. Also, mesh-mesh collision can improve penetration situations at the cost of system performance.

As future work, we would like to improve the system by utilizing haptic response to give the user more realistic experience and integrate our current work with the HDI^{2}4D [6] project in process. The HDI^{2}4D project promises to not only provide an immersive Virtual Reality environment but also allows the user to physically interact with it, enabling them to use their hands to touch, feel and move objects in the virtual space. Thus, our work certainly fulfills the project demands as using our method the user can interact with virtual object freely in an efficient way with minimum hardware requirements. Our algorithm can grasp object parts smaller than the hand size only, so two hands interaction is another possible extension of the method as it can allow interactions for a wide variety of objects. Moreover, we will investigate the possibility of using user image hand as a virtual hand model, instead of creating a hand model from tracked point-set; we believe it can result in improved interaction as it will trick user’s mind to realize virtual hand as his real hand.

BIBLIOGRAPHY

1. *"Meaningful Touch and Gestural Interactions with Simulations Interfacing via the Dart Programming Language".* **Hawick, T.H. McMullen and K.A.** s.l. : Computational Science Investigations **-** Technical Note CSI-0009, 2014, Computer Science Investigations-Technical Note CSI-0009.

2. *"Physics-based Interactive Virtual Grasping".* **Kiran Nasim, Kim Young J.** Seoul : Human Computer Interaction (HCI), Korea, 2016.

3. *"Human-Computer Interaction, Advanced Interaction, Modalities, and Techniques".* **Masaaki, Kurosu and.** s.l. : 16th International Conference HCI 2014, Vol. Cham: Springer International Publishing : Imprint: Springer, p. 24.

4. *"Simulation of Gesture Recognition for Physical Impairments Peoples".* **Dinesh, R. Jayashree and L.** 3, s.l. : International Journal of Advanced Research in Computer and Communication Engineering, 2015, Vol. 4. ISSN (online) 2278-1021.

5. *"Trends in human-computer interaction to support future intelligence analysis capabilities".* **Gouin, D., and Lavigue, V.** s.l. : Proc. 16th Int. Command and Control Research and Technology Symposium (Paper 130), 2010.

6. Human-Digital Content Interaction for Immersive 4D Home Entertainment. *A collaboration between New Zealand and Korea.* [Online] http://computergraphics.ac.nz/hdi4d/.

7. *"Deep learning for detecting robotic grasps".* **I. Lenz, H. Lee, and A. Saxena.** s.l. : The Int. Jour. of Robotics Research (IJRR), 2014.

8. **Iris, Kyranou.** *" Robotic Prosthetic Hand Performance Through Grasp Preshaping".* The University of Edinburgh, Edinburgh. Lackner, United Kingdom. 2014. pp. 7-8, M.S, Thesis.

9. **Richard M. Murray, S. Shankar Sastry, and Li Zexiang.** *"A Mathematical Introduction to Robotic Manipulation".* s.l. : CRC Press, 1994. pp. 212-236.

10. *"Online Grasp Synthesis".* **Grupen, J. A. Coelho Jr. and R. A.** s.l. : IEEE Int. Conf. on Robotics and Automation, 1996.

11. *"Robust Realtime Physics-based Motion Control for Human Grasping".* **Wenping Zhao, Jianjie Zhang, Jianyuan Min and Jinxiang Chai.** 6, s.l. : ACM Transactions on Graphics (TOG) **-** Proceedings of ACM SIGGRAPH Asia 2013, Vol. 32, pp. 32-6.

12. Magnetic Pinch. [Online] https://developer.leapmotion.com/documentation/unity/unity/Unity.MagneticPinch.html.

13. *"Smartwatch-assisted Robust 6-DOF Hand Tracking System for Object Manipulation in HMD-based Augmented Reality".* **Woo, Hyung-il Kim and Woontack.** Greenville,SC, USA : IEEE Symposium on 3D User Interfaces, March 19-23, 2016, pp. 249-250.

14. **Castner, B. M.** *"The development of fine prehension in infancy.".* s.l. : Genetic Psychology Monographs.

15. **Goussous, Faisal Amer.** *"Grasp Planning for Digital Humans".* Iowa : Mater's thesis, Graduate College of The University of Iowa, 2007.

16. *"Controlling Physics-Based Characters Using Soft Contacts".* **Liu, Sumit Jain and C. Karen.** s.l. : ACM SIGGRAPH Asia, 2011.

17. *"A Global Approach for Dexterous Manipulation Planning using paths in n-fingers grasp subspace".* **Jean P. Saut, Anis Sahbani, and Veronique Perdereau.** Singapore : 9th International Conference on Control, Automation, Robotics and Vision, 2006, pp. 1-6.

18. *"GraspIt!: A Versatile Simulator for Robotic Grasping".* **A.Miller.** s.l. : IEEE Robotics and Automation Magazine, 2004, pp. 110-122.

19. *"Virtual Grasping Assessment Using 3D Digital Hand Model".* **Yui Endo, Satoshi Kanai and Takeshi Kishinami.** s.l. : 10th Annual Applied Ergonomics Conference: Celebrating the Past: Shaping the future, 2007.

20. *"Fast grasp synthesis for various".* **Kyota, F., and Saito, S.** s.l. : Comp. Graph. Forum 31, 2pt3 (May), 2012, pp. 765-774.

21. *"Refining grasp affordance models by experience".* **R. Detry, D. Kraft, A. G. Buch, N. Kr¨uger, and J. Piater.** s.l. : In IEEE Int. Conf. on Robotics and Automation (ICRA), 2010, pp. 2287-2293.

22. *"Generalizing grasps across partly similar objects".* **R. Detry, C. H. Ek, M. Madry, J. Piater, and D. Kragic.** s.l. : In IEEE Int. Conf. on Robotics and Automation (ICRA), 2012.

23. *"Leveraging big data for grasp".* **D. Kappler, J. Bohg, and S. Schaal.** s.l. : in Proc. IEEE Int. Conf. Robot. Autom. (ICRA), May 2015, pp. 4304–4311.

24. *"Hybrid control for interactive character animation".* **Shapiro, Pighin, and Faloutsos.** s.l. : Pacific Graphics, 2003, pp. 455-461.

25. *"Virtual Chopsticks: Object Manipulation using Multiple Exact Interactions".* **Y. Kitamura, T. Higashi, T. Masaki, and F. Kishino.** s.l. : IEEE Virtual Reality, 1999.

26. *"Realistic Virtual Grasping".* **Indugula, Christoph W. Borst and Arun P.** Bonn. Germany : IEEE Virtual Reality, 2005.

27. *"An overview of 3d object grasp synthesis algorithms".* **Sahbani, A., El-Khoury, S., and Bidaud, P.** s.l. : Robotics and Autonomous Systems, 60(3), Autonomous Grasping, 2012, pp. 326-336.

28. *"Der mechanische Aufbau der künstlichen Glieder".* **Schlesinger, G.** [ed.] Hartmann D-I e h K, Leymann D, Radike DR, Schlesinger D-I, Schwiening D (éd.). Ersatzglieder und Arbeitshilfen. In : Borchardt DM. s.l. : Springer Berlin Heidelberg, 1919, pp. 321-661. ISBN: 978-3-662-32182-9.

29. **Napier, J. R.** *"The prehensile movements of the human hand".* s.l. : Hand and Brain. The Neurophysiology Soechting, J.F. and Flanders, M. Flexibility and repeatability of finger and Psychology of Hand Movements, 1956.

30. *"Intuitive Virtual Grasping for non Haptic Environments".* **Sauer, T. Ullmann and J.** s.l. : Pacific Graphics, 2000.

31. *"Natural and Robust Interaction in Virtual Assembly Simulation".* **Rettig, G. Zachmann and A.** Anaheim : Eighth ISPE International Conference on Concurrent Engineering: Research and Applications, 2001.

32. **Volz, C. W. Borst and R. A.** "Observations on and Modifications to the Rutgers Master to Support a Mixture of Passive Haptics and Active Force Feedback". Los Angeles : poster presented at Eleventh Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2003.

33. *"Visual Interpenetration Tradeoffs in Whole-Hand Virtual Grasping".* **Borst, M. Prachyabrued and C. W.** s.l. : Proc. of IEEE 3DUI, 2012. pp. 39-42.

34. *"The Effect of 3D Widget Representation and Simulated Surface Constraints on Interaction in Virtual Environments".* **R. W. Lindeman, J. L Sibert, and J. N. Templeman.** s.l. : IEEE Virtual Reality, 2001. pp. 141-148.

35. *"Virtual grasping assessment using 3D digital hand model".* **Endo Y, Kanai S, Kishinami T, Miyata N, Kouchi M, Mochimaru M.** s.l. : 10th Annual Applied Ergonomics Conference: Celebrating the Past- Shaping the Future.

36. **Saffer, D.** *"Designing Gestural Interfaces".* first edition. s.l. : O'Reilly Media, Sebastopol, CA, 2008.

37. *"FingARtips: gesture based direct manipulation in Augmented Reality".* **Buchmann, V., Violich, S., Billinghurst, M., and Cockburn, A.** s.l. : Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and South East Asia (GRAPHITE '04), 2004, pp. 212-221.

38. *"Interactive Hand Gesture-based Assembly for Augmented Reality Applications".* **Stritzke, Rafael Radkowski and Christian.** s.l. : The Fifth International Conference on Advances in Computer-Human Interactions(ACHI), 2012.

39. *"Hybrid Feature Tracking and User Interaction for Markerless Augmented Reality,".* **Lee, T. and Höllerer, T.** Reno, Nevada, USA : IEEE Virtual Reality, March 2008, pp. 145-152.

40. Leap Motion Unity Assets and Plugin. [Online] https://developer.leapmotion.com/documentation/unity/index.html.

41. Wireless Touch. [Online] Oculus. https://www.oculus.com/en-us/touch/.

42. *"Synthesis of interactive hand manipulation".* **Liu, C. Karen.** s.l. : Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA, 2008. pp. 163-171.

43. *"An Analysis of the Precision and Reliability of the Leap Motion Sensor and Its Suitability for Static and Dynamic Tracking".* **J. Guna, G. Jakus, M. Pogačnik, S. Tomažič, and J. Sodnik,.** s.l. : Sensors 2014. 3702-3720.

44. **Jiménez, Samuel.** *"Physical interaction in augmented environments".* s.l. : Gjøvik University College.

45. *"The wiimote and beyond: Spatially convenient devices for 3D user interfaces".* **C.A. Wingrave, B. Williamson, P.D. Varcholik, J. Rose and A. Miller.** s.l. : IEEE Computer Graphics Applications, 2010, pp. 71-85.

46. *"Approach to Hand Tracking and Gesture Recognition Based on Depth Sensing Cameras and EMG Monitoring".* **Jakab, Ondrej Kainz and František.** s.l. : Acta Informatica Pragensia 3(1), 2014, pp. 104-112.

47. **A., Jana.** *"Kinect for Windows SDK Programming Guide".* s.l. : Birmingham: Packt Publishing Ltd, 2012.

48. **White, Ryen W.** *"Interactions with search systems".* s.l. : Cambridge University Press, 2016. ISBN 978-1-107-03422-8.

49. *"Controlling 3D molecular graphics via gesture and voice".* **Sabir, K., Stolte, C., Tabor, B. and O'Donoghue S.I.** s.l. : IEEE Symposium on Biological Data Visualization, 2013, pp. 49-56.

50. *"Issues in the Haptic Display of Tool Use".* **J. Colgate, M. Stanley, and J. Brown.** s.l. : Proc. Int'l Conf. Intelligent Robots and Systems, IEEE CS Press, 1995, pp. 140-145.

51. **Anthony.** High Performance physics in Unity 5. [Online] 8 2014. http://blogs.unity3d.com/2014/07/08/high-performance-physics-in-unity-5/.

52. **Sanders, Brandon.** *"Mastering Leap Motion: Design robust and responsive Leap Motion applications for real-world use".* s.l. : Packt Publishing Ltd, UK, 2014. pp. 24, 28. ISBN 978-1-78355-139-2.

53. Leap Motion C# API Reference. [Online] 2015. https://developer.leapmotion.com/documentation/unity/api/Leap_Classes.html.

54. Getting Started with the Leap Motion SDK. [Online] http://blog.leapmotion.com/getting-started-leap-motion-sdk/.

55. **Tytel, Matt.** Grabbing and Throwing Small Objects, Ragdoll Style. [Online] June 05, 2014. http://blog.leapmotion.com/grabbing-and-throwing-small-objects-ragdoll-style/.

56. Unity Manual. [Online] http://docs.unity3d.com/Manual/index.html.

57. *Robust Realtime Physics-based Motion Control for Human Grasping.* **Zhao, Wenping.**

58. **Sanders, Brandon.** *Mastering Leap Motion: Design robust and responsive Leap Motion applications for real-world use.* s.l. : Packt Publishing Ltd, UK, 2014. p. 28. ISBN 978-1-78355-139-2.

59. *"Human Preferences for Robot-Human Hand-over Configurations".* **Maya Cakmak, Siddhartha S. Srinivasa, Min Kyung Lee, Jodi Forlizzi and Sara Kiesler.** San Francisco, USA : 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2011.

60. *"The anatomy and mechanics of the human hand. Artif Limbs", 2(2).* **Taylor, C. L. and Schwarz, R. J.** 1955, pp. 22-35.

ABSTRACT (KOREAN)
국문초록

가상현실에서의 효과적 객체조작을 위한 물리기반 및 보조형 파지법

Kiran Nasim

컴퓨터공학과

이화여자대학교 대학원

가상현실 환경에서 사용자의 몰입감을 향상시키기 위한 핵심 요소중 하나는 인간과 객체상호작용 방법이다. 하지만 기존에 제안된 목소리, 시선, 도구, 제스처 등 가상현실과 현실을 잇는 상호작용 방법들은 물리적 현실의 그것과 큰 차이가 있다. 본 논문에서는 기구학적 손 모델을 이용하여 사용자의 손의 위치를 실시간에 추적하고 제어함으로써 가상의 물체와 물리적으로 현실감있게 상호작용하는 가상 파지 시스템을 제안한다. 기존의 연구에서 사용되는 손 추적 방법은 가상 물체와의 중첩과 마찰력을 제대로 고려하지 않고 물리적 손과 가상의 손 모델 사이를 단방향으로 제어하여 사용자의 몰입 감을 크게 방해하므로, 본 논문에서는 다음과 같은 두 가지 해결 방법을 제안한다. 첫 번째 방법은 가상 손 모델과 가상 물체간의 접촉 반력을 계산하기 위하여 쿨롱의 마찰 모델을 이용한 마찰력 계산방법이고, 두번째 방법은 기구학적 손 모델을 추적하는 동시에 동역학적 특성을 고려한 프록시 손을 모델 하여 물리기반 시뮬레이션 엔진과의 상호작용이 쌍방향 제어가 가능하도록 하였다. 이러한 프록시 손 모델은 사용자에게 보다 실감 있는 시각적인 피드백을 제공할 뿐만 아니라

가상의 물체와 손 모델 사이에 동역학을 좀 더 강건하게 시뮬레이션한다. 또한 본 논문에서는 효과적인 상호작용을 위해 다음과 같은 두 가지 파지제어법을 제안한다. 하나는 손 모델이 가상 물체과 중첩되지 않도록 마찰력이 적용된 물리 기반 파지법이며, 다른 하나는 제안하는 시스템이 동역학적 특성을 고려할 뿐만 아니라 사용자의 의도를 파악하여 쥐는 제스처를 취할 때 가상 물체를 파지하는 것을 돕는 보조형 파지법이다. 두 가지 파지법은 사용자 연구와 다양한 벤치마킹을 통해 기존의 집기 기반 파지법과 비교하고 계량화하였다. 수행된 실험의 결과 제안된 방법중 보조형 파지법이 가상 환경에 대한 사전 지식이 없는 사용자가 객체 조작을 수행할 때 가장 빠르고 효과적인 방법을 제공하는 것으로 나타났으며, 물리 기반 파지법도 또한 가상 세계에서 현실감 있는 상호작용을 돕는 것으로 입증되었다.