Virtual Human Hand: Grasping and Simulation

Esteban Peña-Pitarch[1], Jingzhou (James) Yang[2], and Karim Abdel-Malek[3]

[1] Escola Politècnica Superior d'Enginyeria de Manresa (EPSEM) UPC, Av. Bases de Manresa 61-73, 08240 Manresa, Spain
[2] Human-Centric Design Research Lab, Texas Tech University, USA
[3] Center for Computer Aided Design, The University of Iowa, USA
esteban.pena@upc.edu

Abstract. The human hand is the most complete tool, able to adapt to different surfaces and shapes and touch and grasp. It is a direct connection between the exterior world and the brain. I. Kant (German philosopher) defined how the hand is an extension of the brain. In this paper we present and develop a new algorithm for grasp any object in a virtual environment (VE). The objective is to present a novel theory for grasping in the VE any object with the virtual human (VH). The novel concepts for this application are the autonomous grasp, implementation of several types of grasp, and a new algorithm for grasp.

Keywords: Autonomous grasp, virtual environment, virtual human hand.

1 Introduction

Grasping has been an active research area for many years, and a great deal of effort has been spent on the automatic determination of grasping actions. The research activity has been oriented in different directions, ranging from robotics applications to the emulation of the human grasp actions using a VH, but the basic concepts are quite similar, and the techniques and methods used in robotics are also applied to the virtual human grasps and vice versa. Following the research line related to VH, this paper presents a novel approach to generating the grasp of different objects in a semi-intelligent VE.

1.1 Related Work

In research related to VH, [1], [2], [3], Dhaiba hand group is working on motion capture, and after being implemented in virtual environment; they study different types of grasps for applications like manipulation of a cell phone. A model of a virtual hand and its implementation in MAND3D was presented in [4]. Complete analysis and classification of the human hand was presented in [5] oriented to the design of hands for manufacturing tasks. [6] presented a design hand without grasp simulation, and hand grasping was presented in [7], based on a database. Regarding the simulation of human fingers, a deformable model based on Hertzian and similar theories was presented in [8]. Analysis of finger motion coordination (except the thumb) in the hand manipulative and gesture acts was studied in [9]. For hand and muscle simulation, a sample surfaces method was presented in [10]. The simulation of

V.G. Duffy (Ed.): Digital Human Modeling, HCII 2009, LNCS 5620, pp. 140–149, 2009.
© Springer-Verlag Berlin Heidelberg 2009

several robot hands was done in the environment called GraspIt! [11], [12]. Based on visual recognition, [13] reconstructed the hand posture using a hand model with 20 degrees of freedom (DOF). Controlling and performing activities with hands with a large number of DOF are very complex tasks; in order to reduce the complexity, [14] coupled the movements of some joints, reducing the number of DOF. Based on three functions of grasping surfaces, namely the object-supporting, pressing, and wrapping functions, [15] presented a study to assist robot designers in producing innovative robotic hand systems. The Sharmes simulator [16] is the development of a highly realistic model of the human hand and forearm; it contains 38 muscles and 24 DOF representing the joints of the system. Two of these DOF are for the wrist, two are for the arm, and 20 are for the hand.

1.2 Paper Outline

This paper is organized as follows. In Section 2, we present a grasping flowchart that was used for our virtual human to grasp. The user chooses one object from several in the VE. Each object has several inherent tasks, i.e., a mug can be moved or used for drinking. The user chooses one of these tasks, and a semi-intelligent algorithm based on previous inputs and object attributes like surface shape and weight makes a decision to grasp with power or precision. Once that decision is made, the number of hands and fingers is calculated based on the surface shape of the object and the type of grasp. In the next, we describe a novel algorithm for grasping any object with power or precision. For a power grasp, the angles for each joint are calculated based on a geometry calculation. For a precision grasp, the fingertip of each finger that grasps is calculated by geometry; to determine the angles of each joint, we apply inverse kinematics and consider each finger an independent ray. In Section 3 we present two examples, without lost generality, and implement the flowchart for these examples. Section 4 presents the conclusion.

2 Grasping Approach

Figure 1 presents a flowchart of the actions needed to perform a grasp in a given environment. First, the user chooses one object out of those entered or included as part of the scene. Each object has attached information about attributes such as surface shape, weight, temperature, fragility, and the task in which they are going to be used, among others; these attributes help the system make a decision about the type of grasp, power grasp, precision handling, pinch, pull, or push.

Once the user chooses a task to be done with the selected object, the system, helps make a decision about how to grasp the object for that purpose, using as input the known attributes of the object. The next step in the sequence is to check whether or not the object is reachable; checking the position of the object with respect to the wrist is possible implementing the row range deficiency method [20]. The row range deficiency method was implemented for each finger sweeping the workspace volume and checking for the object. Selecting the number of fingers and determining if more than one hand is necessary is a function of the hand size and the surface shape of the object to be grasped. The following subsections explain how these decisions are made.

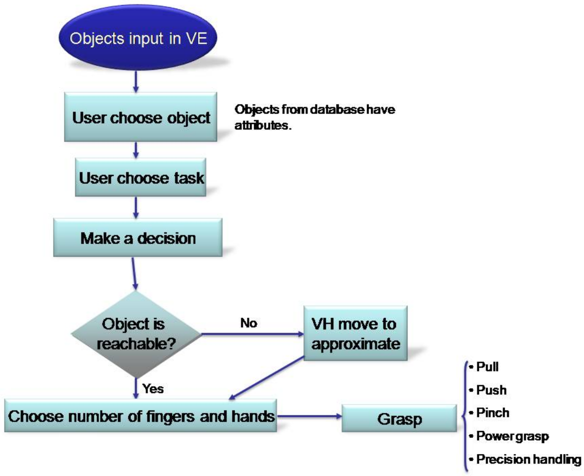

Fig. 1. Flowchart of the actions before performing a grasp

2.1 Make a Decision

The method proposed to make a decision about how a VH must grasp an object in the VE is based on a support vector machine (SVM) applied in a single perceptron. The inputs in a simple associator are the object attributes (task, weight, surface shape, temperature, object stability, and fragility) and the output pattern is the type of grasp (power grasp, precision handling, pinch, pull, and push). When the VH makes a decision, it is based in a linearly separable system.

When grasping any object, the input vector for a single perceptron can be:

$$\mathbf{z} = \begin{bmatrix} task \\ temperature \\ weight \\ \ldots \end{bmatrix}$$

Task, temperature, weight, …, are the input attributes; we can add surface shape, fragility, and object stability.

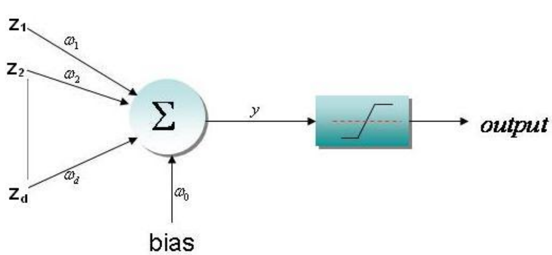

Fig. 2. Single perceptron

Figure 2 shows a single perceptron [17],[18],[19], where the input vector is $\mathbf{z} = [z_1, z_2, \ldots, z_d]^T$ described above, z_0 the bias, the weights for each input are given by the vector $\mathbf{w} = [\omega_1, \omega_2, \ldots, \omega_d]^T$, and the weight for the bias is ω_0.

In this work, these networks are used to solve a classification problem in which the inputs are binary images [1,-1] of attributes like task, weight, surface shape, temperature, object stability, and fragility. The output of the perceptron is given by

$$\mathbf{y} = g\left(\sum_{j=1}^{d} \mathbf{w}\,\boldsymbol{z}^{\boldsymbol{T}} + \omega_0\right) \tag{1}$$

where $\boldsymbol{z}^{\boldsymbol{T}}$ denotes the vector formed from the activations $z_0, \ldots, z_d$, and the function of activation in this case is a symmetric saturating linear (satlins), because this is linearly separable. The output for each of these attributes for the symmetric saturating linear (satlins) can be 1 or -1; if these values are different, the function of activation saturates to 1 or -1. In the example of grasping a mug, if the output is 1, the VH grasps the handle, and if the output is -1, the VH grasps the side of the mug.

The input/output relation is:

$$\begin{array}{cc} g(a) = -1 & a < -1 \\ g(a) = a & -1 \le a \le 1 \\ g(a) = 1 & a > 1 \end{array}$$

With a single perceptron, quite frequently the grasping output does not agree with all the input parameters; i.e., to grasp a mug, the input parameters can be weight, temperature, task, etc. If the VH needs to move a mug that is hot, if the output was "grasp side mug" (that means grasp with power), it is not compatible with the task constraint (a hot mug cannot be grasped with a power grasp). In order to solve this type of problem, new perceptrons were added, building a network with more information about the grasp.

2.2 Choosing the Number of Fingers and Hands

Whether only one hand or both hands are needed to perform the grasp is a function of the shape and size of the object. Attribute surface shape provides information about the size and the weight of the object, which determines if one hand or both hands are used or if the action is not performed. Let D be the side dimensions to grasp and HL be the hand length; if D > 0.8HL then the VH has to use two hands for the grasp.

For precision handling, the virtual human may need two, three, four, or up to five fingers. The number of fingers to be used is also a function of the shape and weight of the object.

Power grasps and simple touches do not need a predefined number of fingers; normally, a power grasp uses five fingers and a touch uses only the index finger. The numbers of fingers used in other grasp types are functions of the attributes mentioned earlier. For precision handling, a primitive sphere is a function of the shape; or better, if the radius of the equator of the sphere is ρ we can define the number of fingers as follows:

$$If\ 1 \leq \rho < 20\ mm\ then\ PH_1$$
$$If\ 20 \leq \rho < 40\ mm\ then\ PH_2$$
$$If\ 40 \leq \rho < 60\ mm\ then\ PH_3$$
$$If\ 60 \leq \rho < 90\ mm\ then\ PH_4$$

where *PH* means precision handling and the subscript indicates the number of fingers that, together with the thumb are involved in the grasp. When the output is power grasp (*PG*), the VH usually uses all the fingers, and we do not consider these conditionals in our algorithm. The following subsection shows how we calculate the angles for each joint and each finger, when the output is power grasp or precision handling.

2.3 Finger Angles

Our approximation for grasping is based on the movement of the fingers. There are two types of movements. For grasping with power, the movement described, each finger except the thumb is circular. For the second, when grasping with precision, the fingertips' position, including the thumb, approximates a circle. Based on these approximations, we can simulate all the human grasping proposed by [5]. Pinching is a particular case of grasping with precision. Pulling, pushing, and touching we considered positioning finger problems. For the first approximation of power grasping we can apply forward kinematics and calculate all the angles for every finger. For a cylinder with radius ρ, Figure 3 depicts a cross-section of the cylinder and the schematic phalanx bones. The angles θ_i and q_i are obtained from the geometry relationship. This example is considered a power grasp.

The angle is q_j for each finger, and the subscript is $j = II \ldots V$ where subscript II is for the index, III is for the middle, IV is for the ring, and V is for the small finger. These angles are for the proximal phalanx with respect to the metacarpal bones for each finger. It is similar for q_k, where $k = II \ldots V$, the subscripts indicate the same fingers as before, and the angles are between the proximal phalanx and the medial phalanx. For q_l, where the subscript is $l = II \ldots V$ the angles are between the middle phalanx and the distal phalanx. All of these angles are calculated for geometry and changed from local to global with a transformation matrix.

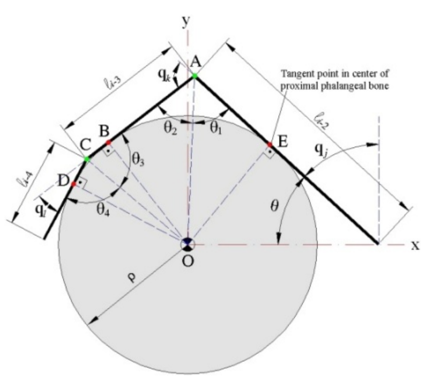

Fig. 3. The geometry relationship of finger segments

For precision grasping, the fingertip positions of each finger on the object boundary are given, and the finger joint angles for grasping the object are computed using inverse kinematics. Figures 4 and 5 depict the fingertip positions on a ball.

Figure 5, the angles β and α depend on the diameter of the ball; from the observation of real people grasping a ball for a radius ρ = 27.5 mm, the results α= 60º and $\beta = 0^o$. In addition, it is imposed that the middle finger stays in its neutral position (i.e., no abduction displacement). Then, the fingertip positions for the thumb, index, and ring fingers can be computed with respect to the wrist (global coordinates) coordinate system, while the small finger stays in the neutral position.

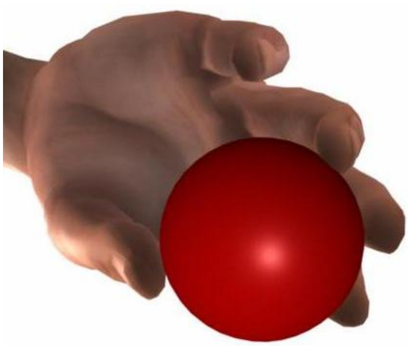

Fig. 4. Grasping a sphere

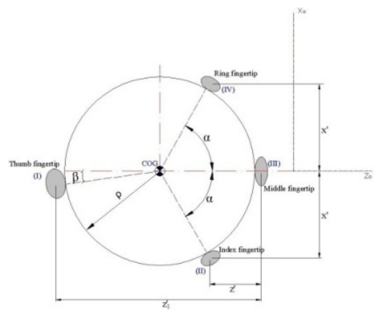

Fig. 5. Equator section with position of fingertips used

The inverse kinematic solutions depend on the initial values of the design variables (q_i) for both iterative and optimization-based methods. Table 1 presents the solutions (q_i in degrees) for the index finger with the Newton-Raphson method, where the global coordinate is $[-11.22\ 152.341\ 77.4]$ in mm, the hand length is 200 mm, and the local coordinate is $[-7.3\ 59.9887\ 77.4]$ in mm.

Table 1 shows that the convergence for the Newton-Raphson method is very fast when the initial angles are close to the solution. For the first set of initial values, the solution for the distal interphalangeal joint (DIP) (q_4) is negative and is in the range

Table 1. Index joint angles with Newton-Raphson method

Iteration	q_1	q_2	q_3	q_4
Initial	0	30	30	10
7	6.95	39.3	30	-7.95
Initial	0	0	0	0
10	6.95	42.3	10	26.6
Initial	0	10	10	0
7	6.95	42.3	10	26.6

of motion. The negative angle for this joint represents hyperextension. However, usually, we can observe that humans never grasp this sphere by DIP hyper-extension.

In practice, some joints in the fingers are coupled, i.e., the movement of one joint depends on the motion of another joint. For example, each finger except the thumb has two coupled joints. The DIP depends on the proximal interphalangeal joint (PIP) and the relationship between them is [14]

$$q_{DIP}^{i} = \frac{2}{3} q_{PIP}^{i}$$

where the superscript i identifies the finger, beginning with 1 for the index finger and ending with 4 for the small finger [21].

For the thumb, similar relationships were observed.

$$q_3 = 2(q_2 - \frac{1}{6}\pi)$$
$$q_5 = \frac{7}{5} q_4$$

3 Implementation

The implementation of the proposed approach has been done in C++. The implementation is divided into modules, following the schema shown in Figure 1. In some modules, the users interact with the virtual environment, i.e., the user chooses the object and task to be done with it; other modules run autonomously. Two examples are given below to illustrate the developed system.

3.1 Grasping a Mug

Input parameters for grasping a mug can be:

$$Task\begin{cases} Drink\ 1 \\ Move\ -1 \end{cases} \qquad Temperature\begin{cases} Hot\ 1 \\ Normal\ -1 \end{cases} \qquad Weight\begin{cases} Heave\ 1 \\ Light\ -1 \end{cases}$$

Attributes like surface shape, fragility, and object stability do not help the system make a decision in this case, and therefore they are considered 0.

To simplify the example, if we choose $z_0 = 0$ and $\omega_0 = 0$, the weight input vector $w^T = [1,0,0,0,0,0]$, and these values are implemented in Equation 1.

Output can be 1 or -1. In this example, if the output is 1, the VH grasps the mug by its handle, and if the output is -1, the VH grasps the mug by its side.

If the input is "drink" (task chosen by the user), "hot" (temperature inherent to the object), light (weight, VE knows the density and the volume and can calculate the weight), the output decision is:

$$y = hardlims\left([1\ 0\ 0\ 0\ 0\ 0] \begin{bmatrix} 1 \\ 1 \\ -1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + 0 \right) = 1$$

Fig. 6. Grasping a mug

In this case, the decision is to grasp the mug by the handle. Figure 6 shows the VH executing the action of grasping a mug by the handle.

3.2 Grasping a Joystick

A similar process and results are shown in Figures 7 and 8 for a joystick. In this case, there are only two tasks, and the attributes are inherent to the joystick.

To grasp a joystick by the side or the top is a function of the task to perform. The most important attribute for this case is the task to be performed with the joystick; for simplicity, these two tasks are to push the rear button, to unload the engine and move the joystick by the top; this movement will imply up-down the load.

$$Task \begin{cases} Push & 1 \\ Move & -1 \end{cases}$$

The output will be 1, grasping the joystick by the side, as shown in Figure 7, or -1, grasping the joystick by the top, as shown in Figure 8.

Fig. 7. Grasping a joystick; power grasp

Fig. 8. Grasping a joystick; power grasp

4 Conclusion

We have presented a novel approach for grasping based on the objects and their functionality. When the object is selected for the user, it is associated with more attributes, which we describe above. After the user chooses the task, the virtual human, if the object is feasible, grasps with the type of grasp calculated as a function of output in a single perceptron. The new concept in this paper is that the virtual human can grasp autonomously without the user once the task is chosen. Support vector machine (SVM) theory, for a perceptron, was applied for this autonomous grasp. After we developed the approach without lost generality, we implemented and showed two examples.

Acknowledgements

This work was partially supported by the projects DPI2007-63665 and the Caterpillar Inc. project: Digital Human Modeling and Simulation for Safety and Serviceability.

References

1. Miyata, N., Kouchi, M., Kurihara, T., Mochimaru, M.: Modeling of human hand link structure from optical motion capture data. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 3, pp. 2129–2135 (2004)
2. Miyata, N., Kouchi, M., Mochimaru, M.: Posture estimation for design alternative screening by dhaibahand - cell phone operation. In: SAE 2006 Digital Human Modeling for Design and Engineering Conference 2006–01–2327 (2006)
3. Miyata, N., Kouchi, M., Mochimaru, M.: Generation and validation of 3d links for representative hand models. In: SAE 2007 Digital Human Modeling for Design and Engineering Conference, 2007–01-2512 (2007)
4. Savescu, A., Cheze, L., Wang, X., Beurier, G., Verriest, J.P.: A 25 degrees of freedom hand geometrical model for better hand attitude simulation. In SAE International, 2004-01-2196 (2004)
5. Cutkosky, M.: On grasp choice, grasp models, and the design of hands for manufacturing tasks. IEEE Transactions on Robotics and Automation 5(3), 269–279 (1989)
6. Albrecht, I., Haber, J., Seidel, H.P.: Construction and animation of anatomically based human hand models. In: Eurographics/SIGGRAPH Symposium on Computer Animation, pp. 1–12 (2003)
7. Aydin, Y., Nakajima, M.: Database guided computer animation of human grasping using forward and inverse kinematics. Computers and Graphics 23(1), 145–154 (1999)
8. Barbagli, F., Frisoli, A., Salisbury, K., Bergamasco, M.: Simulating human fingers: a soft finger proxy model and algorithm. In: HAPTICS 2004. 12th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, pp. 9–17 (2004)
9. Braido, P., Zhang, X.: Quantitative analysis of finger motion coordination in hand manipulative and gestic acts. Human Movement Science 22, 661–678 (2004)
10. Huang, C., Xiao, S.: Geometric modeling of human hand for muscle volume conductor computation. In: 19th International Conference - IEEE/EMBS, vol. 3, pp. 207–208 (1997)
11. Kragic, D., Miller, A., Allen, P.: Real-time tracking meets online grasp planning. In: 2001 ICRA. IEEE International Conference on Robotics and Automation, vol. 3, pp. 2460–2465 (2001)
12. Miller, A., Allen, P., Santos, V., Valero-Cuevas, F.: From robotic hands: a visualization and simulation engine for grasping research. Industrial Robot: An International Journal 32(1), 55–63 (2005)

13. Nolker, C., Ritter, H.: Visual recognition of continuous hand postures. IEEE Transactions on Neural Networks 13(4), 983–994 (2002)
14. Rijpkema, H., Girard, M.: Computer animation of knowledge-based human grasping. Computer Graphics 25(4), 339–348 (1991)
15. Saito, F., Nagata, K.: Interpretation of grasp and manipulation based on grasping surfaces. In: IEEE International Conference on Robotics and Automation, vol. 2, pp. 1247–1254 (1999)
16. Chalfoun, J., Younes, R., Renault, M., Ouezdou, F.: Forces, activation and displacement prediction during free movement in the hand and forearm. Journal of Robotics Systems 22(11), 653–660 (2005)
17. Hagan, M.T., Demuth, H.B., Beale, M.H.: Neural network design. PWS Pub., Boston (1996)
18. Haykin, S.: Neuronal Network. A Comprehensive Foundation., 2nd edn. Prentice Hall international, Inc., Englewood Cliffs (1999)
19. Anderson, J.A.: An Introduction to Neural Networks, 3rd edn. Bradford Book (1997)
20. Peña-Pitarch, E., Yang, J., Abdel-Malek, K.: SANTOSTM Hand: Workspace Analysis. In: 16th IASTED International Conference on Modelling and Simulation, Cancun, Mexico, May 18-20 (2005)
21. Peña-Pitarch, E., Yang, J., Abdel-Malek, K.: SANTOSTM Hand: A 25 Degree-Of-Freedom Model. In: SAE 2005 Digital Human Modeling for Design and Engineering Conference, 2005-01-2727 (2005)