30

A VISION-BASED APPROACH OF BARE-HAND INTERFACE DESIGN IN VIRTUAL ASSEMBLY

Xiaobu Yuan and Jiangnan Lu
University of Windsor, Windsor, Ontario, Canada N9B 3P4
{xyuan|lu12}@uwindsor.ca

This paper presents the continuous work on a previous project on vision-based hand interaction design. The original work developed a vision-based approach of bare-hand-based posture recognition and motion tracking. This paper further investigates its application in virtual assembly. The accuracy and robustness of the developed approach enable product engineers to perform assembly operations while manipulating virtual objects directly in virtual environments with bare hands. The direct human involvement creates a user-defined assembly sequence, which contains the human knowledge of mechanical assembly. By extracting the precedence relationship of machinery parts, the system becomes capable of generating alternative assembly sequences for robot reprogramming. Operation of the presented approach is illustrated and analyzed with experiments.

1. INTRODUCTION

Virtual assembly replaces physical objects with the virtual representation of machinery parts, and provides advanced user interfaces for human operators to design and generate product prototypes, to analyze and optimize manufacturing processes, and to verify and control work-floor implements directly in computer-synthesized environments. The elimination of physical prototyping and on-site verification makes virtual assembly a powerful tool to reduce the life-cycle of manufacturing and to adapt changes or introduce new products.

With the support of new devices, virtual assembly develops new means for human operators to directly manipulate virtual objects with their hands. Hand-based interface in virtual assembly or other manufacturing systems can be considered as an upgrade to the conventional interface design, and has become the key to improve the naturalness of human-computer interaction in virtual environments. Consequently, the tracking and recognition of hand shapes and movements are the main issues in interaction design for virtual assembly (Rivière, 2004).

In the past, special virtual reality devices are employed to recognize hand shapes and to track hand movements (Fels, 1993). Data gloves, for example, are such a type of device that produces sensory inputs to determine the spatial positions of hands and to monitor the temporal parameters of hand shapes. However, data gloves are

expensive and fragile. Cables connecting to the computer also impose limits to the movement of human operators (Baudel, 1993). The problems of glove-based devices have motivated active research to search for better solutions, and a vision-based approach of bare-hand-based posture recognition and motion tracking was developed (Yuan and Lu, 2005).

This paper further applies the developed approach in virtual assembly. The structure of this paper is as follows. After providing a brief survey of related work in the next section, this paper first highlights the vision-based approach of posture recognition and motion tracking in Section 3, and then discusses interaction design with bare hands in Section 4. Afterwards, Section 5 explores virtual assembly with bare-hand interaction. The discussions of this paper finish with experiment results and conclusions at the end.

2. RELATED WORK

Virtual assembly involves two processes, i.e., robot programming to teach robotic manipulators carrying out assembly tasks and sequence planning to determine the order of assembly tasks for optimal mechanical assembly. The old-fashion online approach suffers from the down-time of robot operations, the danger imposed upon human operators, and the difficulty of making adjustments for new products. In comparison, virtual assembly works inside the computer. It promotes the development of automated manufacturing tools and allows for the integration of different technologies from a wide range of originally separated areas.

However, the lack of flexible and user-friendly human-computer interaction has been a major bottleneck. Text-based programming, low-end graphics interfaces, and advanced graphics systems have been the common approaches to program robots accomplishing assembly tasks. While robot programming languages suffer from high complexity and inaccurate calibration, graphics interfaces with traditional input devices, such as keyboards and mouse, inherently limit the speed and naturalness. In addition, high-level robot programming without motion planning creates problems in implementation as objects run freely in virtual environments but not in the physical world.

In fact, the paradigm of robot programming has not changed until the introduction of hand functioning in human-computer interaction. Data glove devices have been successfully used to track hand movements and to measure hand configuration parameters (Ma, 1998). One typical example is programming-by-demonstration (PbD), which uses data gloves to convert the movement of human operators into assembly operations (Friedrich, 1996). Unfortunately, virtual reality devices are commercially expensive and cumbersome to use. Moreover, glove-based interaction is powerless for remote control when the user at the far side is not equipped with such devices.

Recent advancement in computer vision stimulated studies of vision-based hand interfaces (Ionescu, 2005). Without the need of mechanical devices, vision-based interfaces provide a more convenient platform that visually interprets hand postures and tracks hand positions. While glove-based interfaces quantify human hands with electronic signals, vision-based interfaces represent the model of bare hands with visual features that reside in image frames (Comport, 2003). Uniquely colored

gloves or markers, for example, belong to such vision-based approaches. They present distinguishable color patches in images for hand tracking. Colored gloves and marks are cheaper than data gloves, but are still inconvenient to use. Besides, changes to marker colors or lighting conditions require system modification and re-initialization.

Ideal hand-based interaction neither relies on data-glove devices nor needs any artificial markers. It uses bare hands as human beings do in their daily lives. Different models of bare hands have been developed for the localization and recognition of hand shapes (Rivièrel, 2005). They use a variety of hand features, including skin colors, contours, and fingertip and binary silhouette. Most of them help to segment and identify bare hands for posture classification, but fail to preserve depth information. Nevertheless, depth information is essential for the tracking of hand rotations and movements, and plays an important role in virtual programming as robot operations take place in 3D work cells.

3. BARE-HAND PROCESSING

Hand segmentation and posture recognition are two major parts of bare-hand interaction. While the former deals with the detection of low-level hand features, the latter uses hand features to reconstruct 3D hand trajectories.

3.1 Hand Localization and Segmentation

Color-based approaches of hand processing are robust to dynamic and changing light environments, but work only with monochromic backgrounds. They are also computationally intensive, and therefore unable to implement in real-time. Motion-based mechanisms, in comparison, are fast in computation and work well with complex backgrounds. Nonetheless, motion-based algorithms are highly susceptible to noise and require static background. The drawbacks make techniques in the second category inapplicable in real environments.

A hybrid model is developed by combining skin color detection with motion differencing. This hybrid model maximizes the system's ability to separate skin class from non-skin class while minimizing negative effect of illumination. Experimental results reveal that HSV color space yields 72% correct classification rate, and is therefore the choice for skin color detection. Furthermore, by extending skin color detection from individual pixels to a window of 3x3 pixels, not only the candidate pixel but the neighboring pixels are involved in classification.

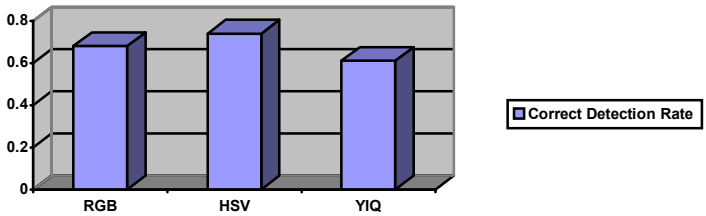

Figure 1 – A comparison of different color spaces

The selection of an HSV color and the use of neighboring pixels help to remove most constraints and to alleviate the computational burden. However, noise still remains due to changes in lighting condition and the presence of other parts of human body. To eliminate the impacts of noise, a skin tone filter is also designed. It is a gradient pattern matching for the hue component of skin color in HSV color space, and the hue pattern of user skin is obtained from the image differencing algorithm given in Equation 1 (Yuan and Lu, 2005). Analysis shows that the hue values vary from 0.8 to 1.0 for the same bare hand, but never reach 0.65 for noisy hands or the background.

$$F_H = F_u / F_d \tag{1}$$

where,

$$\mathrm{F_u} == \sum\nolimits_{(u,v)\in W} H_{user}(u,v) \bullet H_{noise}(x+u, y+v)$$

$$\mathrm{F_d} = \sqrt{\sum\nolimits_{(u,v)\in W} (H_{user}(u,v) - \overline{H}_{user})^2 \bullet \sum\nolimits_{(u,v)\in W} (H_{noise}(x+u, y+v) - \overline{H}_{noise})^2}$$

3.2 Fingertip Detection and Tracking

Feature selection is critical in bare-hand-based interaction, and fingertips are one of the frequently used features in hand-based interaction design. Among other fingertip detection techniques, such as template matching and texture mapping, contour-based detection has demonstrated its potentials for its simplicity, flexibility, and efficiency. Contour-based detection relies on the characteristic properties of fingertips in an image. For instance, curvature of a fingertip outline follows a characteristic pattern (low-high-low), which can be cues for feature detection. Other benefits for contour-based detection come from its robustness to noise and tolerance to image scaling.

The developed approach uses K-curvature measurements for fingertip detection. Once the hand is segmented from the original images, the system extracts boundary pixels in the hand contour image. The pixels that reside in the outline of the hand form a list in which each pixel is denoted by $P_i=(x_i, y_i)$. The K-curvature is the angle C_i between two vectors $[P_{i-k}, P_i]$ and $[P_i, P_{i+k}]$, where k is a constant. When curvatures are greater than a fixed threshold with positive or negative magnitudes respectively, the curvatures of pixels along the boundary reach a local maximum/minimum. They are the "local features" in forms of "peaks" or "valleys".

Fingertip tracking needs to find corresponding fingertips in consecutive frames of an image stream, and to record hand movements. It is more challenging than rigid object tracking as it has many degrees of freedom to deal with but few features to use. To eliminate motion blurring caused by fast fingertip movements, the developed approach identifies fingertips by re-running the fingertip-finding algorithm for each frame. Suppose $p_t=(x_1, y_1\ t)^T$ and $q_t=(x_2, y_2, t)^T$ are two coordinates of a fingertip in a pair of stereo-vision frames at moment t. The 3D position of the target fingertip can be calculated with projective geometry. Fingertip trajectory is then reconstructed and stored in a sequence of matrices.

4. BARE-HAND INTERACTION

A bare hand interface allows operators to specify and evaluate assembly operations directly in the virtual environment with their bare hands. In particular, hand postures trigger control commands for object manipulation, and hand gestures map to 3D trajectories of collision-free paths.

4.1 Identification of Hand Gesture and Postures

Each assembly task involves both the global and fine motions of objects. The bare-hand-based interaction uses four controlling commands to quit an action and to point to, hold on, or free objects. Accordingly, a set of four postures defines the vocabulary for the four commands respectively as shown in Figure 2. This interface uses the number of fingertips to define controlling commands. In particular, the "point" posture uses only one finger, and "hold" uses two fingers. When the posture shows five fingers, it means "free"; and a no-finger hand posture means "quit".

Figure 2 – Hand postures for object manipulation

Assembly operations translate and/or rotate machinery parts in the 3D space. Fingertip tracking in bare-hand-based interaction makes translation relatively easy to achieve, but the measurement of rotations needs three fingertips to determine. Due to the fact that the fingertips of the thumb, the index finger, and the little finger can hardly be collinear, they are the fingertips to define rotations. In addition, the difference in shape between the thumb and the little fingers makes them easy to distinguish. Hand morphology and biomechanics further suggest that the index finger is always in the middle of the two fingers in natural gesticulations. The index fingertip is then identified by examining their relative positions.

4.2 Bare-Hand-Based Object Manipulation

Shown in Figure 3 is a state transition diagram that illustrates hand functioning in the bare-hand-based interface. The posture and gesture start as two concurrent states. They operate in the super state of 'continuous modes', generating control commands and motions. In particular, a closed hand triggers the *hold* command to make a selection or uphold an action; an open hand implies *free*, which releases an item in the possession of the virtual hand or frees the hand from constraints; a point sign invokes items for selection; and an "quit" sign terminates a running process.

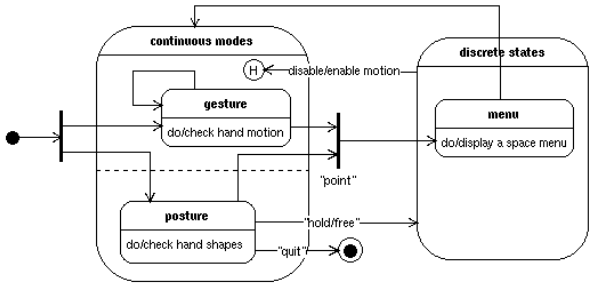

Figure 3 – The state transition diagram of "hand"

Before the hand picks up an object, the *hold* command works primarily with *free* for hand adjustment. It releases the hand from the continuous control mode. The motion of the hand in the physical world no longer contributes to any motion in the virtual environment. In such a way, the hand is able to prepare itself to a desired position or orientation. Object manipulation begins with a *point* command, which prompts up a space menu in the virtual environment and activates menu selection. A casting ray from the index finger then highlights the menu item it hits, which becomes selected when followed immediately by *hold*.

If the selected item is labeled as "reference", the control goes into a remote selection mode, in which a user selects graphics features to define reference coordinate systems. Alternatively, the selection of menu item "motion" starts object manipulation. At this time, a hold posture grabs the object under selection, and makes it to move together with the tip of the index finger until a free posture releases it. In either the "reference"' or "motion" mode, a *quit* command returns the control to the menu state. One of them remains active until the selection of menu item "done".

5. VIRTUAL ASSEMBLY WITH BARE-HAND INTERACTION

In a virtual assembly, the positions of objects are visually presented as part of the virtual environment. The user's manipulation of virtual objects through a human/computer interface transforms into the definition of assembly tasks and a user-defined assembly sequence. An assembly task starts when the hand grabs an object and finishes until the object reaches to its destination. An assembly is a product put together from a set of parts P by a virtual robot manipulator accomplishing a sequence of assembly tasks T in the three-dimensional workspace. In the assembly, $P = P_0+P$, where P_0 denotes a stable workstation and $P = \{P_1, P_2, ..., P_n\}$ is the set of machinery parts, and $T = t_1\ t_2\ ...t_n$ is an ordered assembly tasks.

The goal of virtual assembly is to produce optimized assembly sequences that are ready to direct robot manipulators implementing assembly tasks and putting together machinery parts into products. Given a user-defined assembly sequence, virtual assembly applies geometric and symbolic reasoning to determine the precedence constraints of individual objects, and converts the constraints into an assembly tree for the generation of alternative sequences (Yuan and Gu, 1999).

Optimization comes from the evaluation of alternative sequences against certain selected measures, and calibration takes the characteristics of each individual robot manipulator into consideration.

Precedence constraints are the ordering constraints that guarantee the validity of assembly task sequences. A sequence is feasible if it satisfies common requirements such as geometric, stable, and orientation constraints. Following the order of object manipulation in the user-defined assembly sequence, constraints deduction checks one by one if the assembly of a machinery part must wait for the assembly of any others. This information then leads to the construction of an assembly tree, in which the root stands for the stable workstation P_0 and each path that links from the root to a leaf node represents an assembly sequence.

In addition to the generation of alternative sequences, the virtual assembly system is also responsible to assist human operators with their work of programming assembly tasks and evaluating assembly sequences. The scope of assistance may range from robotics simulation to automatic reprogramming. The former uses computer graphics to present assembly tasks by animating object movements and robot operation with inverse kinematics and dynamics. In comparison, the latter allows the import of human expertise through virtual programming, but still reduces human involvement to the least possible level.

6. EXPERIMENTS AND ANALYSIS

Experiments have been conducted in two groups to verify the operation of virtual assembly with bare-hand interaction, and to evaluate the performance in hand posture recognition and hand movement tracking.

6.1 The Assembly of a Die-Set

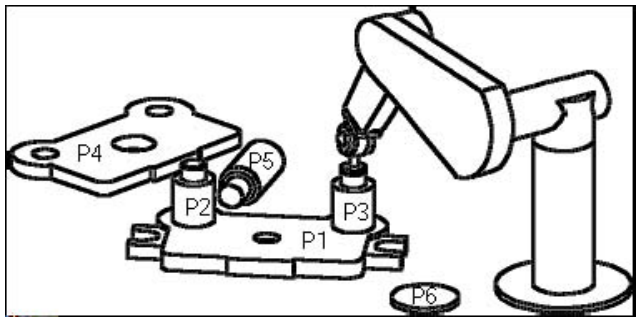

Figure 4 – A Virtual Environment for the assembly of a die-set

Figure 4 illustrates a typical die-set. To put together the die-set with five components and to place the metal plate for stamping, a user-define feasible sequence of assembling the objects can simply be in the order of P_1,P_2, P_3, P_4, P_5, and P_6. Correspondingly, the assembly tasks to place the objects are t_1, t_2, t_3, t_4, t_5, and t_6. Each of the tasks t_i, as listed below, contains the information to instruct a robot manipulator reaching to, moving around, and releasing an object P_i.

t_1: place P_1 with its half-open hole facing up;
t_2: stand up P_2 by inserting it into a guide-hole;
t_3: stand up P_3 at the other guide-hole of P_1;
t_4: sit the two corner-holes of P_4 on the guideposts;
t_5: insert P_5 half-way through the bigger hole of P_4;
t_6: slide P_6 on top of the half-open hole of P_1.

Right after the user defined assembly sequence, the virtual assembly system determines the precedence relationship between objects. Sequence generation then begins to construct the assembly tree. Figure 5 shows ten possible sequences to assemble the die-set in the constructed assembly tree, including the one defined by the user. Taking a sequence out of the tree, a replay moves each object in the sequence one after another by applying the corresponding assembly task recorded during programming. For the first sequence in the tree, as an example, operations t_1, t_2, t_3, t_6, t_4, and t_5 are applied to objects P_1, P_2, P_3, P_6, P_4, and P_5. Experiment shows that there is no problem applying the original tasks in this new sequence.

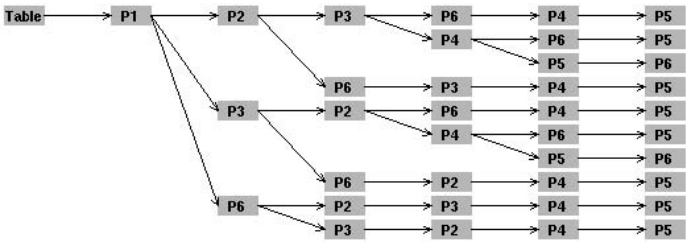

Figure 5 – An automatically constructed assembly tree of the die-set

6.2 Robustness Analysis

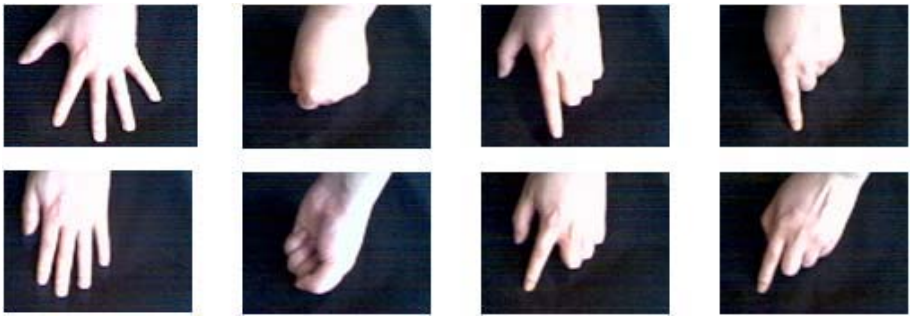

Figure 6 – Hand postures recognition

Experiments in the second group evaluate the performance of hand posture recognition and hand movement tracking with bare-hand interaction. Hand posture recognition plays a major role in bare-hand-based interaction design, and its robustness in virtual assembly was a focus of the evaluation. As shown in Figure 6,

the shape of the four defined hand postures was intentionally changed or rotated in experiment, and results showed that all of the distorted postures were recognized correctly.

Bare-hand-based interface is primarily designed to eliminate the dependency of virtual assembly on specialized devices. Due to the fact that no one can repeat the same hand movements in the 3D space, it is impossible to evaluate a bare-hand-based interaction by comparing the tracking results with the measurements from data glove devices. In comparison, marker-based hand tracking is similar to bare-hand-based hand tracking in the way that it tracks hand movements in image streams. The use of extra color marks on hands is expected to produce better results than bare hands.

Accuracy evaluation is therefore conducted by comparing results obtained from the two algorithms on the same hand movements. Three test cases were built in experiment, each of which contains a pair of hand trajectories reconstructed by the two tracking algorithms. In all three test cases, ten sampling points were checked. A direct comparison of the 3D coordinates with the sampling points indicates the accurate rate was at 87.3%, 91.2%, and 89.1% for the three test cases respectively. The result is very encouraging as 90% is considered good accuracy for hand tracking with data glove devices.

7. CONCLUSIONS

Human-computer interaction based on bare hands has undeniable advantages in virtual manufacturing. This paper investigates the application of bare-hand-based interaction in virtual assembly. By visually tracking hand motions and interpreting controlling commands, the presented approach enables human operators to perform assembly operations in an intuitive and natural way. The user-defined assembly sequence embodies human knowledge of mechanical assembly, which in turn helps to generate alternative assembly sequences for robot reprogramming. Experiments demonstrated the robustness and efficiency of the presented approach. Active research is being conducted to further investigate its application in practice.

8. ACKNOWLEDGEMENTS

This work was funded by Natural Sciences and Engineering Research Council of Canada (NSERC).

9. REFERENCES

1. Lakatta EG, Cohen JD, Fleg JL, Frohlich ED, Gradman AH. Hypertension in the elderly: age- and disease-related complications and therapeutic implications. Card Drugs Ther 1993; 7: 643-54.
2. Smith, Adam. “An Inquiry into the Nature and Causes of the Wealth of Nations”. In Classics of Economics, Charles W. Needy, ed. Oak Park, IL: Moore Publishing, 1989.
3. Ward, Benjamin. What’s Wrong with Economics. New York: Basic Books, 1972.
4. Baudel T, Charade M. Remote control of objects using free-hand gestures. Comm. ACM. 1993; 36: 28-35.

5. Comport A, Marchand É, Chaumette F. "A real-time tracker for markerless augmented reality." In Proc. 2nd IEEE and ACM International Symposium on Mixed and Augmented Reality. 2003 ; 36-45.
6. Rivière J, Guitton, P. Hand postures recognition in large-display VR environments. Lecture Notes in Computer Science. 2004; 2915: 259-268.
7. Rivière J, Guitton, P. Model-based video tracking for gestural interaction. Virtual Reality. 2005; 8: 213-221.
8. Fels S, Hinton G. A neural network interface between a data-glove and a speech synthesizer. IEEE Trans. Neural Networks. 1993; 4: 2-8.
9. Friedrich H, Munch S, Dillmann R, Bocionek S, Sassin M. Robot programming by demonstration (rpd) - supporting the induction by human interaction. Machine Learning. 1996; 23: 163-189.
10. Ionescu B, Coquin D, Lambert P, Buzuloiu V. Dynamic hand gesture recognition using the skeleton of the hand. Journal on Applied Signal Processing. 2005; 13: 2101-2109.
11. Ma L, Lau R, Feng J, Ng C. Shape modifying and controlling with sensor glove in virtual-reality. Progress in Natural Science, 1998; 8: 488-497.
12. Yuan X, Gu Y. "An integration of robot programming and sequence planning". In Proc. the 1999 IEEE International Conference on Robotics and Automation. 1999; 102-107.
13. Yuan X and Lu J. "Virtual programming with bare-hand-based interaction". In Proc. of 2005 IEEE International Conference on Mechatronics and Automation. 2005; 896-900.