Augmented Reality Instruction for Object Assembly based on Markerless Tracking

Li-Chen Wu* I-Chen Lin † Ming-Han Tsai ‡
National Chiao Tung University

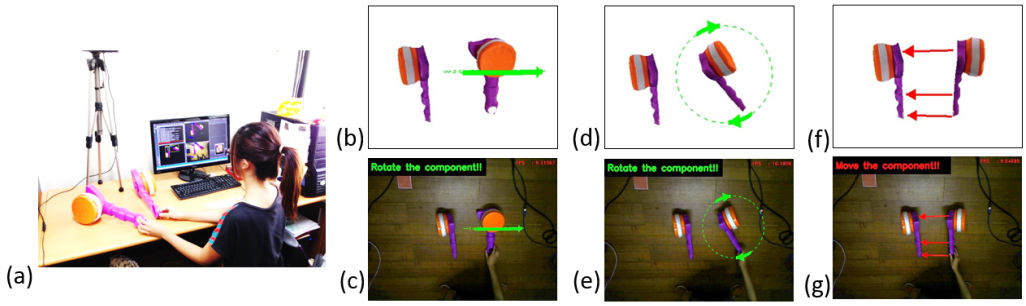

Figure 1: *(a) The working environment of the proposed assembly instruction system. (b)(d)(f) The synthesized models and instructions according to the estimated 3D poses of components. (c)(e)(g) Assembly instructions superposed on live views.*

Abstract

Conventional object assembly instructions are usually written or illustrated in a paper manual. Users have to associate these static instructions with real objects in 3D space. In this paper, a novel augmented reality system is presented for a user to interact with objects and instructions. While most related methods pasted obvious markers onto objects for tracking and constrained their orientations or shapes, we adopt a markerless strategy for more intuitive interaction. Based on live information from an off-the-shelf RGB-D camera, the proposed tracking procedure identifies components in a scene, tracks their 3D positions and orientations, and evaluates whether there are combinations of components. According to the detected events and poses, our indication procedure then dynamically displays indication lines, circular arrows and other hints to guide a user to manipulate the components into correct poses. The experiment shows that the proposed system can robustly track the components and respond intuitive instructions at an interactive rate. Most of users in evaluation are interested and willing to use this novel technique for object assembly.

Keywords: assembly instruction, augmented reality, object tracking

Concepts: •**Human-centered computing** → **Interaction techniques;** •**Computing methodologies** → **Tracking; Mixed / augmented reality;**

*e-mail: lichenwu.cs02g@g2.nctu.edu.tw
†e-mail: ichenlin@cs.nctu.edu.tw
‡email: ParkerTsai@caig.cs.nctu.edu.tw

I3D '16, February 27-28, 2016, Redmond, WA
ISBN: 978-1-4503-4043-4/16/03
DOI: http://dx.doi.org/10.1145/2856400.2856416

1 Introduction

As the popularity of do-it-yourself (DIY) products and online shopping, of which products are usually decomposed into parts for compact packing size, users have more chances to assemble objects by themselves. Assembly instructions are usually drawn or written in manuals. Users have to map the indication on paper onto actions for real objects, and they cannot get any feedback or help from this kind of static instructions. Several researches were proposed to interactively guiding the assembly process of users. They usually attached particular markers on the surfaces of components [Reiners et al. 1998; Zauner et al. 2003; Henderson and Feiner 2011b]. A user has to keep these markers visible during the assembly process. Besides, not all of the objects or components are suitable for marker sticking.

Our goal is to provide instant and dynamic instructions during the object assembly process of a user. Instead of using markers, we adopt recognizing the identifications and poses of components according to depth and color images captured by a camera. For this real-time task, our detection and tracking methods aim at balancing the computation cost and detection accuracy. A template matching method is used to efficiently compare an unknown foreground with multiple views of different components stored in the database. The template matching can provide an initial pose of a recognized component. An extended iterative closest point (ICP) method is further applied to refining and tracking the 3D pose. Our detection and tracking procedure can handle situations of partial occlusion among components and hands.

Based on the relative poses among components, the proposed system infers the current state from an assembly structure tree. It then generates corresponding graphical indication, such as alignment lines, circular arrows and so forth, to guide a user to manipulate the components on hand. Moreover, these instructions are superposed onto the live captured video for intuitive display and interaction. Figure 1 shows the setting and several snapshots of the proposed system. With the instant augmented reality (AR) instructions, users only have to follow the indication arrows and accomplish the object assembly at ease.

2 Related Work

Due to the recent development of mobile displays, augmented reality (AR) become an attractive topic again. Several researches applied this technique for interactive narratives [Kapadia et al. 2015]. In 2003, Tang et al. [2003] conducted experiments about the effectiveness of AR. They specified that the AR system improved the performance of the object assembling processing. Henderson and Feiner [2011a; 2011b] discussed AR in maintenance tasks. Their experiments showed that AR interfaces can reduce the time to locate targets and reduce head movements. AR instruction is more effective than static 3D graphics instruction in the psychomotor phase. Reiners et al. [1998] guided a user to assemble the doorlock onto the car door for industry usage. Zauner et al. [2003] designed an marker-based AR system for furniture assembly. They mentioned and alleviated the occlusion problem by sticking more than one markers on each component. Khuong et al. [2014] utilized a voxel matching method to recognize statuses of LEGO block assembly . Their constrained their pose estimation problem to 2D translations on a table and one in-plane rotation. Alvarez et al. [2011] presented an impressive markerless AR-based system providing disassembly instructions. The statuses and poses of an object were estimated based on edge and junction point features. Therefore, their objects were with salient edge junctions and less surface texture. Their system then superposed predefined instructions according to estimated main object information, and did not consider the relative poses between components.

Several tangible interfaces applied different sensors for user interaction instead of markers. Liang et al. [2013] attached a magnetic sensor grid on the back of a display to track non-ferrous components in which the magnets are embedded. An optical multi-touch tabletop was used to track touch points of users in [Ren et al. 2012]. Other researchers tracked objects based on computer vision techniques. The Portico system proposed by Avrahami et al. [2011] appended two color cameras to a tablet for surrounding objects detection. Gupta et al. [2012] proposed a model assembly system that was exclusively for Dulop blocks. Held et al. [2012] acquired scenes by a RGB-D camera and generated 3D animation according to the poses of physical puppets. Since they utilized the SIFT features [Lowe 2004], this system was applicable to objects with obvious textures or intensity edges.

Template matching is a practical solution for real-time object detection and tracking, when the targets are known. This subsection focuses on the features from depth and color images and how to match templates in 3D space. Lowe [2004] detected rotation and scale invariant key points from images, and the local gradient histograms around a key point were recorded as its descriptor. This SIFT feature is robust for matching objects with rich textures, but it is not suitable for textureless objects. Hinterstoisser et al. [2012a; 2012b] introduced a template matching method, LINEMOD, which combined the color and depth features. This method expresses an image in a binary form and operations and can efficiently detect objects with our without obvious texture.

Tracking the object pose in 3D space can be considered a registration problem between point clouds. Besl et al. [1992] proposed the classic Iterative Closest Point (ICP) method for registering two point clouds. ICP iteratively searches the closest corresponding points between two point sets and estimated their transformations. ICP method has a well-known problem that it tends to be trapped in the local minimum. Yang et al. [2013] obtained the global optimum through searching the whole space of rotation and translation by a nested branch and bound algorithm. However, it is not feasible for real-time tracking. Kyriazis et al. [2013] presented a novel concept to estimate the pose of a handheld object in occlusion situations. They represented this problem by a hand model with 27 degrees of freedom (DOFs). Such a high DOF problem was solved by the particle swarm optimization (PSO).

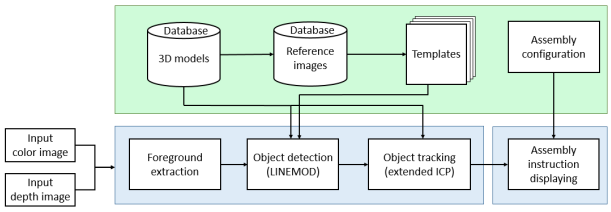

Figure 2: *The flow chart of the proposed system.*

3 System Overview and Dataset Collection

The proposed system is devoted to facilitating object assembly. It can be divided into online and offline processes as shown in the Figure 2. The offline process is shown in green (top), and the online processes are shown in blue (bottom). During the offline stage, we used 123D Catch [Autodesk Inc.] to reconstruct the 3D models of components and objects from multi-view images. These 3D models are then projected onto designated views to generate the reference images (view templates) and their color and depth features. We categorized the models into two types: general and symmetric. The general models are of asymmetric shapes, and the symmetric models are rotational symmetric about one of three coordinate axes. As shown in Figure 3, the viewpoints of a general-type model are sampled at vertices of a sphere mesh derived from an icosahedron. For the symmetric model, the viewpoints are sampled by using a semicircle. The sampled viewpoints represent the out-plane rotation of a model. The included angles between two adjacent viewpoints are around 15 degrees. For each viewpoint, we also sampled 24 reference images regarding in-plane rotation. In addition to template preparation, the relations among components are also defined in the offline stage.

During the online processes, a background subtraction method is used to extract foreground regions in advance. Our system then checks whether a foreground region can be tracked from known components. Otherwise, an extended LINEMOD method is utilized to match an unknown foreground with view templates in the database, and we can acquire the component identification and its rough 3D orientation. The extended ICP is further proposed to refining the orientation and tracking the following movements of a component. In the last step, the proposed system analyzes the relative poses among physical components in the working environment, and infers the indication arrows, sounds and messages to guide a user assembling the components on hand.

4 Object Detection and Tracking

4.1 Detection of components and their rough poses

We adopted the LINEMOD method for detection because the it is capable of recognizing both the textureless components and assembled objects, on which more edges appear. The original method compares an input with database templates according to their color gradients and surface normals from depth images. In our observation, we found that the correct template can get a high similarity score from this method but it may not be the one with the highest score. If we use the template with their highest similarity score, the detection results occasionally become unstable. Therefore, instead of choosing the best template reported by LINEMOD, we get the top K templates with the highest scores and present an second-pass

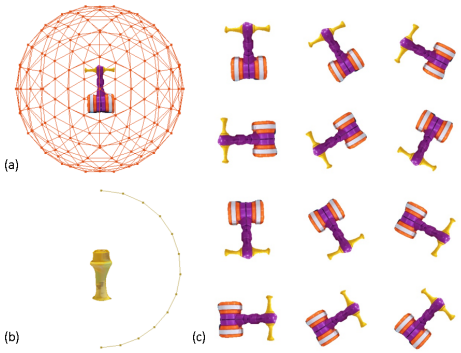

Figure 3: *Two types of models and their reference images. (a) A general-type model. The viewpoints are 162 vertices on a sphere mesh. (b) A symmetric-type model. The viewpoints are sampled at 13 vertices of a semi-circle. (c) Examples of the reference images generated in (a) and there are 24 images in total for the in-plane-rotations.*

evaluation criterion to amend their results.

When inspecting the failure cases, we found that that the silhouette shape and the hue of color can be complement features for matching. We define our measurement function to retrieve the best match $\mathcal{T}$ as shown in Equation (1) and (2).

$$T = \underset{i}{\operatorname{argmin}}\, E_{det.}(I_s^{'}, I_h^{'}, R_s^i, R_h^i) \tag{1}$$

$$E_{det.}(I_s^{'}, I_h^{'}, R_s^i, R_h^i) = \lambda_{det} D_s + (1 - \lambda_{det}) D_h \ , \tag{2}$$

where $I_s^{'}$ and $I_h^{'}$ are the silhouette and hue map of the input region, of which the sizes are normalized to a fixed scale, and R_s^i, R_h^i are the silhouette and hue map of the reference image i among the top K template. Equation 2 is composed of silhouette distance D_s and hue distance D_h.

$$D_s = 1 - \frac{\sum(I_s^{'} \cap R_s^i)}{\sum(I_s^i \cup R_s^i) + \epsilon} \ , \tag{3}$$

, where ϵ is a small constant to avoid division by zero. Equation 3 evaluates the ratio of the intersection to union of two silhouette areas. If D_s is close to zero, it implicates that the $I_s^{'}$ and R_s^i are similar to each other. In order to distinguish the components with the similar silhouettes but different color appearances, the second term D_h is defined as

$$D_h = 1 - \frac{\sum[(I_s^{'} \cap R_s^i) dist(I_h^{'}, R_h^i)]}{\sum(I_s) + \epsilon} \tag{4}$$

$$dist(I_h^{'}, R_h^i) \begin{cases} 1 & if \left| I_h^{'} - R_h^i \right| < T_h \\ 0 & otherwise \end{cases} \tag{5}$$

The hue map is extracted from the hue channel of images in the HSV space. It can reduce the influence of illumination changes. The fraction in Equation 4 represents the percentage of the overlapping pixels of which the hue value are similar (i.e. less than T_h).

Figure 4 shows the effectiveness of our measurement function. We can see that although the silhouettes of the reference image Figure 4 (e) and (i) are similar, the system can still select the correct one by the hue distance. With the above process, we can retrieve the appropriate template $\mathcal{T}$ with the lowest cost $E_{det.}$, and it substantially improves the detection accuracy.

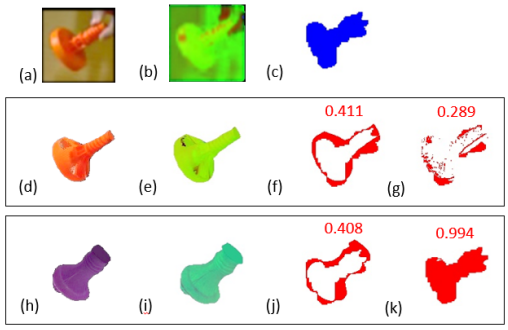

Figure 4: *The detection result and the visualization of D_s and D_h. (a) The normalized input color image. (b) The normalized hue map $I_h^{'}$ of input. (c) The normalized input silhouette $I_s^{'}$. (d)(h) Two reference images (templates) selected by the LINEMOD. (e)(i) The hue map R_h^i. (f)(j) The visualized D_s. (g)(k) The visualized D_h.*

4.2 Extended ICP and the measurement in the projective view

4.2.1 The Extended Iterative Closest Point Method

We also extended the ICP method [Besl and McKay 1992] to refine the coarse pose of a component and update its pose in the following frame. ICP is known for its easiness to be trapped into a local minimum, and thus, we present three modifications to decrease the chances to be trapped.

Hidden surface removal
The goal is to find the optimal rotation and translation to align the input point cloud $\mathcal{Q}$ from the depth map I_d with the point cloud $\mathcal{P}$, generated from the a 3D model. In most of the related methods, the $\mathcal{P}$ is the whole surface points of a model. Since we already have a correct but coarse initial pose (viewpoint), we can exclude the points that should not be visible from the initial viewpoint. Using the partial point cloud decreases the ambiguity during alignment and reduces the tremble of component poses between frames.

Color constraint
The original ICP utilizes the geometric information only. In our system, both the input data point set $\mathcal{Q}$ and the model data point set $\mathcal{P}$ are with color information. Several related work [Douadi et al. 2006; Men et al. 2011] mentioned the benefits of color in ICP. Hence, we add the constraint of color similarity during searching the corresponding points.

Bidirectional correspondence check
In the orignial ICP, for every point $p_i \in \mathcal{P}$, the ICP algorithm finds its corresponding point $q_j^* \in \mathcal{Q}$ in a single direction. Our extended ICP searches and checks the corresponding points in two directions. It not only finds the closest point $q_j^* \in \mathcal{Q}$ for p_i but also the closest point $p_j^* \in \mathcal{P}$ for q_i. When p_i and q_j are the closest points to each other, they can be regarded a bidirectional correspondence and are used to estimate the transformation.

4.2.2 Validation of the extended ICP results

In order to make sure whether the pose estimation result is adequate, we design a validation function. If the cost value $E_{tra.}$ is

smaller than the threshold $T_{tra.}$, it means that the pose is acceptable; otherwise, we mark the component or object is missing in this frame. The validation function is listed as follows.

$$E_{tra.}(I_s, I_d, S_s, S_d) = \lambda_{tra.} D_f + (1 - \lambda_{tra.}) D_d \quad (6)$$

I_s and I_d are the silhouette and the depth map of the input region of the current scene. S_s and S_d are the synthesis silhouette and depth map by projecting the component model onto the refined viewpoint reported by extended ICP. D_f and D_d compute the differences of the silhouettes and depth maps between the input and the synthesis data. The former term D_f measures the difference of the shapes in the projective view:

$$D_f = 1 - Bin(1 - \frac{\sum(I_s \cap S_s)}{\sum(I_s \cup S_s) + \epsilon}) \frac{\sum(I_s \cap S_s)}{\sum(I_s) + \epsilon} \quad (7)$$

$$Bin(D) = \begin{cases} 0 & if D > 0.5 \\ 1 & otherwise \end{cases} \quad (8)$$

The term $\frac{\sum(I_s \cap S_s)}{\sum(I_s)+\epsilon}$ evaluates the ratio of the number of overlapped pixels to that of the input silhouette. We adopt $\sum I_s$ as the denominator instead of $\sum(I_s \cup S_s)$. That is because when there is occlusion by the users' hands, the area of I_s is small and the S_s becomes too dominant. We also designed the $Bin()$ function to decide whether the term $\frac{\sum(I_s \cap S_s)}{\sum(I_s)+\epsilon}$ is valid or not. If the $Bin()$ returns the value 0, implies that the two silhouettes differ from each other substantially, and therefore, the term D_f should be assigned to 1 directly.

We also use the term D_d to evaluate the distance of depth maps between the input and the synthesis.

$$D_d = 1 - Bin(D_f) \frac{\sum[(I_s \cap S_s) dist(I_d, S_d)]}{\sum(I_s \cap S_s) + \epsilon} \quad (9)$$

$$dist(I_d, S_d) = \begin{cases} 1 & if |I_d - S_d| < T_d \\ 0 & otherwise \end{cases} \quad (10)$$

Similarly, the rightmost fraction in Equation (9) evaluates the ratio of the number pixels with similar depth values to that of the intersection region. The depth values are similar if their difference is less than the threshold T_d. The term D_f and D_d are complementary because the D_f measures the contours between the input and synthesis result and the D_d measures the internal undulation. Hence, it can avoid the ambiguity in the cases with the smaller D_f but different poses. An example is shown in Figure 5.

4.3 Runtime States of foreground regions

As mentioned above, when an input color image and depth map are acquired from the RGB-D camera, we extract the foreground pixels by background subtraction. These pixels are then grouped into regions by a flood fill method (connected component labeling). During runtime process, these regions are marked one of the four states: detecting, tracking, closing and combining states, and each state is associated with corresponding operations.

We defined a foreground region set $\mathcal{R}$, where $\mathcal{R} = \{\mathcal{R}_i\}$ and $i = 1, 2, \ldots, N_{\mathcal{R}}$, and this set is updated in each frame. For each $\mathcal{R}_i \in \mathcal{R}$, it has properties $\{I_{ci}, I_{di}, I_{si}, \mathcal{O}_i\}$. I_{ci} is the foreground RGB image of the region such as Figure 6 (a)(e)(i), and I_{di} is the

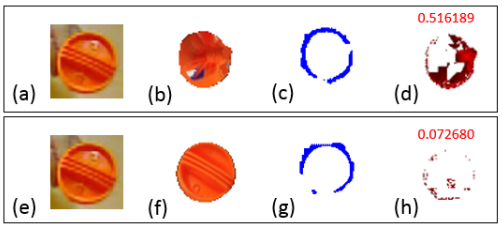

Figure 5: *The visualization of the validation. (a),(e) The color image of an input component. (b),(f) The synthesis result according to estimated poses. (c),(g) The difference of silhouettes D_f. (d),(h) The difference of depth maps by D_d. We can see that (b) and (f) are of similar silhouettes. However, the pose in (b) is incorrect, and it also has a higher cost D_d.*

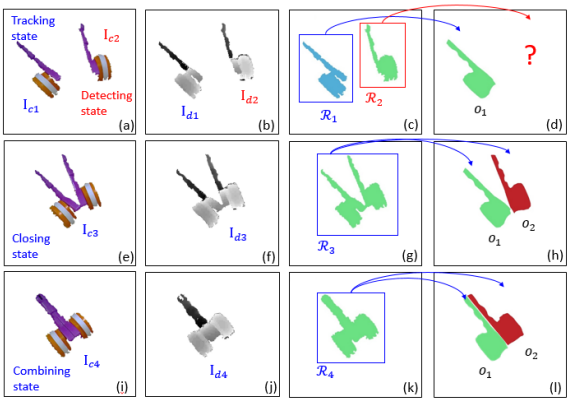

Figure 6: *The illustration of the regions and their states. (a)(e)(i) The foreground of input color image I_{ci}. (b)(f)(j) The foreground of depth image I_{di}. (c)(g)(k) The silhouette I_{si} of the Region R_i and the Region R_i presented in the different color. Each color presents one region. (d)(h)(l) The objects which have occurred and been recognized in previous frames. We can see that there is no object belonging to the R_2, and hence the R_2 is in the detecting state. The R_1 is marked as the tracking state because of the one corresponding object o_1. The R_3 and R_4 are labeled as the closing state and combining state according to the poses of their corresponding objects.*

foreground depth map as shown in Figure 6 (b)(f)(j), and I_{si} is the silhouette of the region as in Figure 6 (c)(g)(k). For each region, we find whether there are close components or objects in previous frames can partially fit this region. The set $\mathcal{O}_i$, where $\mathcal{O}_i = \{o_k\}$ and $k = 1, 2, \ldots, N_{\mathcal{O}_i}$, records the corresponding objects associated with the region $\mathcal{R}_i$. For example, in Figure 6 (c), the corresponding object of R_1 is o_1, so the $\mathcal{O}_1 = \{o_1\}$. In the the Figure 6 (g), the object set of $\mathcal{R}_3$ is $\mathcal{O}_3 = \{o_1, o_2\}$.

Figure 6 exhibits the situations of the four states: the detecting, tracking, closing and combining state. Their definitions are briefly described as follows. Please refer to the supplementary image for the flow of state transitions.

Detecting state: if the set $\mathcal{O}_i \in \mathcal{R}_i$ is empty, which means that no existing component belongs to a region $\mathcal{R}_i$, the region $\mathcal{R}_i$ is labeled as a detecting state.

Tracking state: if the size of the set $\mathcal{O}_i \in \mathcal{R}_i$ is one, meaning that there is only one object belonging to the region $\mathcal{R}_i$, then the region $\mathcal{R}_i$ is in the tracking state.

Closing state: if the size of the set $\mathcal{O}_i \in \mathcal{R}_i$ is two or more than two, and the pose of each object $o_k \in \mathcal{O}_i$ has not reached the com-

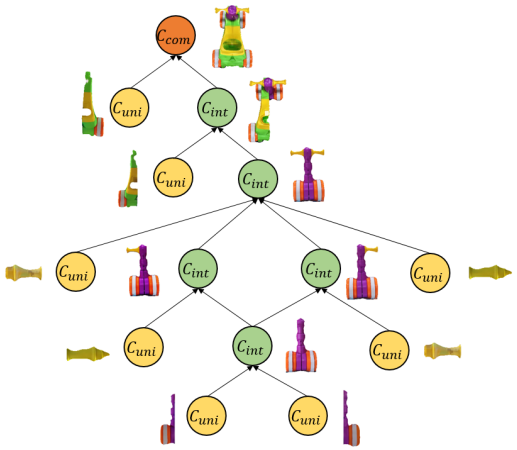

Figure 7: *An example of a tree structure for the assembly process.*

bination requirement yet, we say the region $\mathcal{R}_i$ is in the closing state. We have to perform multiple times of extended ICP to separate the objects from a region.

Combining state: if the size of the set $\mathcal{O}_i \in \mathcal{R}_i$ is two or more than two, and the pose of each object $o_k \in \mathcal{O}_i$ achieves the combination requirement. Then the region $\mathcal{R}_i$ is set to the combining state. The multiple components within the region have chances to become a combined component.

5 Assembly Guidance

Our system provides live instruction for component assembly, which can guide a user to combine the physical components intuitively and correctly. After the object recognition and pose tracking, the system analyzes the pose relations between the components and displays a hint to the user. How to turn the analyzed poses into a visual instruction is also a challenging work because it depends on the technique of 3D object tracking and needs to consider the spatial relationship on the current scene. In this section, we discuss the requirement of combination first, and then present how we show the instruction on the interface.

5.1 Requirement of combination events

We organize the whole assembly process as a bottom-up tree structure shown in the Figure 7. Here, the components are classified into three types: unit component (C_{uni}), internal component (C_{int}) and complete component (C_{com}). If a component is labeled as a unit, it means that the component is at the leaf node in the tree. The assembly process starts from two unit components in the level 0 of the tree. If a component is an internal component, it means that the component is composed of two unit components, one unit and one internal component, or two internal components. The complete component is the root of the tree and it is the final step of whole assembly process. Figure 7 illustrates one example of the assembly structure tree.

We define $\mathcal{A}$, where $\mathcal{A} = \{A_i\}, i = 1, 2, \ldots, N_{\mathcal{A}}$, is the set which records the combination information for each internal node, and $N_{\mathcal{A}}$ equals to the total number of the internal nodes. For $A_i \in \mathcal{A}$, where $A_i = \{R_i, P_i\}$, it records the requirement of the relative pose between two components for invoking a combination event. If the relative pose of the children nodes achieves the requirement A_i, the combination event occurs. Then, the components are combined

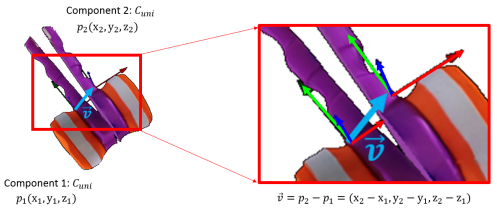

Figure 8: *Illustration of two components which conform with the combination requirement.*

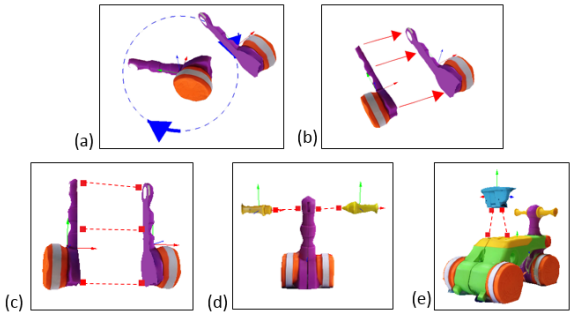

Figure 9: *The illustration of the indication lines. (a) The blue circle is the indication circle (b)The red lines are the alignment lines. (c)(d)(e) shows each set of the components and their representative matched points.*

to form the internal component. Hence, before the system starts, the combination requirement A_i should be predefined for all of the internal nodes. The rotation requirement R_i is a threshold value which controls the tolerance of rotation error during alignment. The right part of Figure 8 shows two components in the expected pose and we can see their local coordinates are in a similar orientation. The position requirement P_i records the relative position between the component origins and an example vector $\overrightarrow{v}$ is shown in Figure 8.

5.2 Display of indication lines and circles

The indications are determined in two steps. At the first step, the system checks the relative rotation between two components on the scene. If the relative rotation does not reach the requirement R_i, the system shows the rotation-circle to instruct the user to rotate the component into the correct orientation. Afterwards, the second step is to correct the component position by showing alignment lines. The indications of a rotation circle and an alignment line are illustrated in Figure 9 (a)(b).

The alignment line links the representative matched points between two components. As shown in Figure 9 (c), three matched points are set on the left and right components respectively. For the indication circle, we estimate the relative rotation matrix R_{ij} between two local coordinates of components i and j. In our early design, we decomposed the matrix R_{ij} by the arbitrary axis rotation, and found the axis $axis_{ij}$ with the smallest angle θ_{ij}. The axis $axis_{ij}$ and circular arrows with angle θ_{ij} were shown on the screen for instruction. However, the axis $axis_{ij}$ may not be aligned with the view axes or the ground axes. In a few cases, some of the pilot users did not well recognized the axis orientation, and users prefer rotating a component on the table to lifting and rotating the component. Therefore, we further separated the rotation process into two sub-steps: rotation along the table normal z direction, and rotation

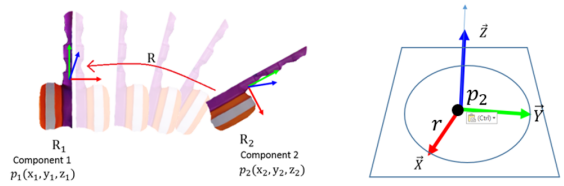

Figure 10: *Illustration of the method drawing the rotation-circle based on the global coordinate. Left: The orientations of two components are not correctly aligned. Right: The circle C of the rotation-circle.*

along the axis orthgonal to z. Figure 10 show that table normal as the z axis and the x and y are the projection of camera view axes of the camera.

5.3 Design of User Interface

In order to design the user interface, we conducted a pilot experiments and design our interface after discussing with the subjects. We invited two users who had not used our instruction system before to be our subjects. In the first experiment, we only displayed the indication lines include rotation circles and alignment lines on the interface and did not show any other hint. After the experiment, the subjects said that the biggest problem they faced during the assembly process was that they felt confused about what the next component should be taken. Therefore, we design a Next Component window to list all the components that can be taken in the next step. In the second experiment, we wonder whether indication lines can effectively guide a user, and thus we closed the indication line and only display the Next Component window. By our observation, during the assembly process, the subjects can easily take the correct component for the next component by following our Next Component window. However, the subjects were not sure about the way to combine two components without indications, which implicated that the indication lines help the user during the assembly process. After these two experiments, the subjects also recommended that adding the sound effects when the special events occur may improve the users concentration. Through our pilot experiment, we design our user interface which includes four parts as shown in Figure 11. Figure 11 (a) is the VR window which displays the synthesis result of detection and tracking. Figure 11 (b) is the AR window, where all of the assembly information are shown . Figure 11 (c) is the window of exhibiting all the components that users should take next. Figure 11 (d) is the Stage window showing the current model that a user have assembled. Besides the four parts on the interface, we also added sound effects when the detecting and combining events occur. The system plays a ding sound when the detecting event occurs, and it plays a triplet chord sound when the combining event is invoked. In few cases, a user have combined two components but the system has not detected the combining event due to missing tracking. We place a red region (button) in the top left of the view. When users touch the red region, the system goes to the next step.

6 Experiment

The proposed system was built on a PC with a quad-core, 3.4 GHz CPU and 12 GB RAM. Currently, only two threads are invoked. We adopted the ASUS Xtion Pro Live [ASUSTek Computer Inc.] as our RGB-D camera device. Due the the limitation of the device, the camera have to be placed 80 cm higher than the table, and the field of view must cover the working area such that any two components

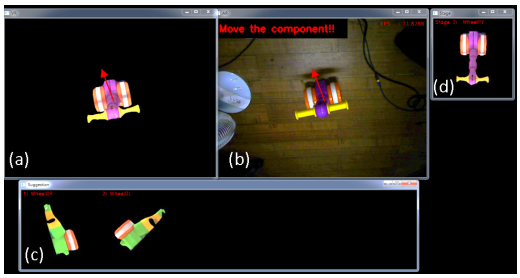

Figure 11: *The design of our user interface.*

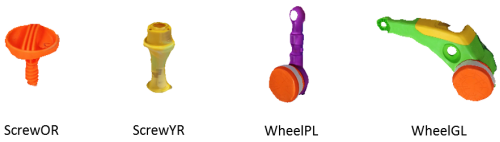

Figure 12: *Examples of components used in the object assembly experiments.*

can be manipulated by a user. The resolution of the input color and depth images are 640×480 pixels. Figure 12 shows several components and their names used in our object assembly.

In our current system, we set the threshold $T_{tra.} = 0.4$ in the pose measurement $E_{tra.}$ and the threshold $T_{det.} = 0.57$ for the detection measurement $E_{det.}$. Both the weights λ_{det} and λ_{tra} are 0.5. The threshold T_h about the hue tolerance in equation (5) is 15 and the threshold T_d in equation (10) is 10. The unit of the depth value is millimeter (mm).

6.1 Efficiency and detected accuracy

For evaluating our system, we recorded a video sequence of 3113 frames with multiple components as shown in Figure 12. The average detecting FPS is 9.55 with 10128 templates for matching and the tracking FPS is 17.20. To evaluate the effectiveness of our detecting measurement function in Equation 2, we conduct two experiments about the true positive rate of our selected template $\mathcal{T}$ from the the top K results compared to the ground-truth component identification and its orientation.

In the first experiment, we detected the components by all the categories of templates. The total categories is 11 and the total template number is 10128. The LINEMOD detector returns the top K matched templates and we select the most appropriate result $\mathcal{T}$ through our measurement function. We define the detected result is true if both of the category and the orientation of a component are correct. Figure 13 shows the detecting rate from $K = 1$ to $K = 10$. Since matching the whole view templates increases the ambiguity during the LINEMOD detection, the accuracy is only acceptable. However, because a large part of errors come from the orientation error which can be fixed in the following extended ICP, our detection performs well in the run-time process. Figure 13 shows the detection rate changes according to the number K. Accordingly, we set $K = 7$ in our system when we need to detect all possible components.

Furthermore, in our assembly application, the detector usually matches a region with templates from only a few components, such as the existing components near the region and the next components. It is more like a conditional detection problem. In the second experiment, we detected the components only using the tem-

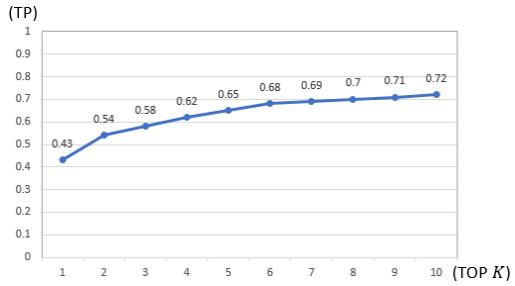

Figure 13: *The true positive rate for the best template selection from top K candidates from all view templates. A test is considered true-positive only when the component identification and orientation are both correct.*

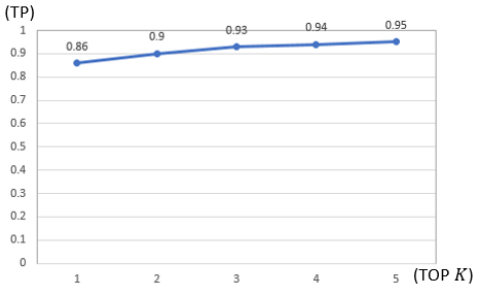

Figure 14: *The true positive rate for the best template selection from top K candidates from view templates of a given category.*

plates of a given component and measured the detection accuracy for $K = 1$ to $K = 5$. Figure 14 shows that the true positive rate under a conditional detection is significantly improved.

6.2 User experience

For the user evaluation, we built two datasets: "Toy Bicycle" and "Toy Cart" are shown in Figure 16. They are from a "trasformable" toy, and most of the components of these two toys are common. The detailed information about the number of the unit component, internal component, complete component and the total number of the templates of each dataset are shown in Table 1. We invited six users as our subjects including four females and two males. They did not used our system before. We separated the users into two groups. In the first stage of our experiment, the subjects in the Group 1 were given the paper manual (with illustrations in clear viewpoints) and assembled the dataset Toy Bicycle. In the second stage, the subjects assembled the other dataset, Toy Cart, through our instruction system. The subjects in Group 2 assembled the Toy Cart first by paper manual(with illustrations in clear viewpoints) and assembled the Toy Bicycle next.

After the experiments, the subjects were asked to fill a questionnaire. In our questionnaire, we designed three major questions to compare our system with paper manual. In the question 1, users have to score the difficulty for assembling the objects by paper manual and our system in overall. In the question 2, users have to score the comprehension for understanding the guidance and applying it on the assembly. In the question 3, users have to score the helpfulness between the paper manual and our system. The result of the comparison is shown in Figure 15. For the question 1 and question 2, the scores 1 to 5 represent the difficulty to simplicity. For the question 3, the scores 1 to 5 represent the helplessness to helpfulness. The reported scores support our system.

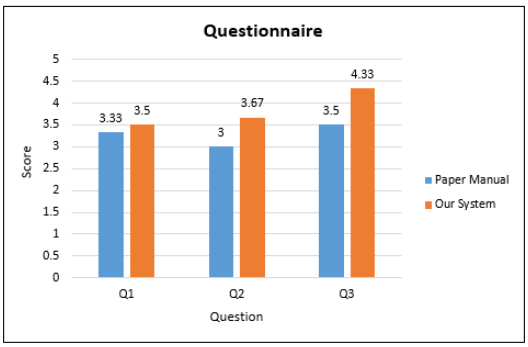

Figure 15: *The reuslt of our questionnaire.*

We also asked the subjects that whether they are willing to use our system to help them during the assembly. Five subjects said that they would select our system because our system makes the whole assembly process easier. It is helpful for them because our system can immediately notice users the current stage and whether the assembly is correct. The subjects also said it is clearer for the whole assembly process through our system, because they know what component they should take is in the next step, and there are matching points between the two components. Only one subject expressed that he/she is not willing to use our system because he/she feel stressful in our limited working space and in front of a camera. She/he preferred assembling the components on his/her way. On the other hand, users reported that they actually enjoyed using this new assembly technology. They did not have to think and just followed the instant instructions. However, due to the response time of our current system (about 10 to 18 fps), they preferred slowing their motions and keeping the indications following their actions. Please refer to the supplementary video to see the user interaction with the proposed system.

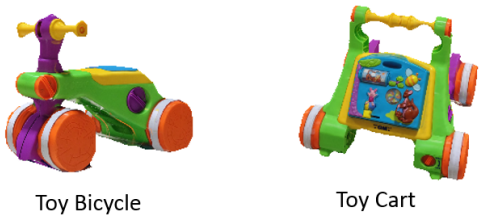

Figure 16: *The two datasets for our experiments. (Left) The complete object "Toy Bicycle". (Right) The complete object "Toy Cart".*

Table 1: *The information of the two datasets.*

Dataset	$\#C_{uni}$	$\#C_{int}$	$\#C_{com}$	#templates
Toy Bicycle	11	16	1	30264
Toy Cart	11	9	1	21432

7 Conclusion and Future Work

In this paper, we propose a novel tangible interface to guide a user assembling the components in an intuitively way. Interacting with real objects is a challenging work. While several related

work adopted using markers, we extended state-of-the-art detection methods and presented an framework to estimate the 3D poses and their interaction among markerless components manipulated by users. An assembly tree structure is also described to handle the intricate assembly process, where multiple components and steps are involved. We also presented two types of indications, rotation circles and alignment lines to guide a user to combine components. In the user evaluation, most of the users give positive responses to our prototype system, where the interaction is interesting and also intuitive.

There are several future works. The proposed system can be developed with parallel computation, and the response time will be substantially improved. It is worthwhile to further analyze the pros and cons of such an interface from various aspects through user evaluation. It is also possible to import graph construction methods, e.g. [Li et al. 2008], to automatically construct our assembly tree. We think this technique is suitable for applications with a head-mounted display (HMD), such as Oculus [Oculus VR]. However, our current camera requires long range for sensing. We plan to port our system to new sensors and HMDs in the future.

Acknowledgements

The authors appreciate helpful comments from reviewers. This paper was partially supported by the Ministry of Science and Technology, Taiwan under grant no. MOST 104-2221-E-009-129-MY2 and 104-2218-E-009-008.

References

ALVAREZ, H., AGUINAGA, I., AND BORRO, D. 2011. Providing guidance for maintenance operations using automatic markerless augmented reality system. In *Proc. IEEE Intl. Symp. Mixed and Augmented Reality*, 181–190.

ASUSTEK COMPUTER INC. Xtion pro live. https://www.asus.com/3D-Sensor/Xtion_PRO/.

AUTODESK INC. 123d catch. http://www.123dapp.com/catch.

AVRAHAMI, D., WOBBROCK, J. O., AND IZADI, S. 2011. Portico: tangible interaction on and around a tablet. In *Proc. ACM Symp. User Interface Software and Technology*, 347–356.

BESL, P. J., AND MCKAY, N. D. 1992. A method for registration of 3-D shapes. *IEEE Trans. Pattern Analysis and Machine Intelligence 14*, 2, 239–256.

DOUADI, L., ALDON, M.-J., AND CROSNIER, A. 2006. Pair-wise registration of 3d/color data sets with icp. In *Proc. IEEE/RSJ Intl. Conf. Intelligent Robots and Systems*, 663–668.

GUPTA, A., FOX, D., CURLESS, B., AND COHEN, M. 2012. DuploTrack: a real-time system for authoring and guiding duplo block assembly. In *Proc. ACM Symp. User Interface Software and Technology*, 389–402.

HELD, R. T., GUPTA, A., CURLESS, B., AND AGRAWALA, M. 2012. 3D puppetry: a kinect-based interface for 3D animation. In *Proc. ACM Symp. User Interface Software and Technology*, 423–433.

HENDERSON, S., AND FEINER, S. 2011. Exploring the benefits of augmented reality documentation for maintenance and repair. *IEEE Trans. Visualization and Computer Graphics 17*, 10, 1355–1368.

HENDERSON, S., AND FEINER, S. K. 2011. Augmented reality in the psychomotor phase of a procedural task. In *Proc. IEEE Intl. Symp. Mixed and Augmented Reality*, 191–200.

HINTERSTOISSER, S., CAGNIART, C., ILIC, S., STURM, P., NAVAB, N., FUA, P., AND LEPETIT, V. 2012. Gradient response maps for real-time detection of textureless objects. *IEEE Trans. Pattern Analysis and Machine Intelligence 34*, 5, 876–888.

HINTERSTOISSER, S., LEPETIT, V., ILIC, S., HOLZER, S., BRADSKI, G., KONOLIGE, K., AND NAVAB, N. 2012. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Proc. Asian Conf. Computer Vision*, vol. 7724, 548–562.

KAPADIA, M., FALK, J., ZÜND, F., MARTI, M., AND GROSS, M. 2015. Computer-assisted authoring of interactive narratives. In *Proc. ACM SIGGRAPH Symp. Interactive 3D Graphics and Games*, 85–92.

KHUONG, B. M., KIYOKAWA, K., MILLER, A., LAVIOLA JR., J. J., MASHITA, T., AND TAKEMURA, H. 2014. The effectiveness of an ar-based context-aware assembly support system in object assembly. In *Proc. IEEE Virtual Reality*, 57–62.

KYRIAZIS, N., AND ARGYROS, A. 2013. Physically plausible 3d scene tracking: The single actor hypothesis. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 9–16.

LI, W., AGRAWALA, M., CURLESS, B., AND SALESIN, D. 2008. Automated generation of interactive 3d exploded view diagrams. *ACM Trans. Graphics 27*, 3, 101:1–101:7.

LIANG, R. H., CHENG, K. Y., CHAN, L., PENG, C. X., CHEN, M. Y., LIANG, R. H., YANG, D. N., AND CHEN, B. Y. 2013. Gaussbits: magnetic tangible bits for portable and occlusion-free near-surface interactions. In *Proc. SIGCHI Conf. Human Factors in Computing Systems*, 1391–1400.

LOWE, D. G. 2004. Distinctive image features from scale-invariant keypoints. *Intl J. Computer Vision 60*, 91–110.

MEN, H., GEBRE, B., AND POCHIRAJU, K. 2011. Color point cloud registration with 4d icp algorithm. In *Proc. IEEE Intl. Conf. Robotics and Automation*, 1511–1516.

OCULUS VR. Oculus rift. https://www.oculus.com/.

REINERS, D., STRICKER, D., KLINKER, G., AND MÜLLER, S. 1998. Augmented reality for construction tasks: doorlock assembly. In *Proc. Intl. Workshop on Augmented reality*, 31–46.

REN, Z., MEHRA, R., COPOSKY, J., AND LIN, M. C. 2012. Tabletop ensemble: touch-enabled virtual percussion instruments. In *Proc. ACM SIGGRAPH Symp. Interactive 3D Graphics and Games*, 7–14.

TANG, A., OWEN, C., BIOCCA, F., AND MOU, W. 2003. Comparative effectiveness of augmented reality in object assembly. In *Proc. SIGCHI Conf. Human Factors in Computing Systems*, 73–80.

YANG, J., LI, H., AND JIA, Y. 2013. Go-icp: Solving 3d registration efficiently and globally optimally. In *Proc. IEEE Intl. Conf. Computer Vision*, 1457–1464.

ZAUNER, J., HALLER, M., BRANDL, A., AND HARTMAN, W. 2003. Authoring of a mixed reality assembly instructor for hierarchical structures. In *Proc. IEEE/ACM Intl. Symp. Mixed and Augmented Reality*, 237–246.