6D Hands: Markerless Hand Tracking for Computer Aided Design

Robert Y. Wang[1,3], *Sylvain Paris*[2], *Jovan Popović*[2,4]

[1]3Gear Systems
Foster City, CA
rywang@threegear.com

[2]Adobe Systems, Inc.
Cambridge, MA
{sparis,jovan}@adobe.com

[3]MIT CSAIL
Cambridge, MA

[4]Univ. of Washington
Seattle, WA

ABSTRACT
Computer Aided Design (CAD) typically involves tasks such as adjusting the camera perspective and assembling pieces in free space that require specifying 6 degrees of freedom (DOF). The standard approach is to factor these DOFs into 2D subspaces that are mapped to the x and y axes of a mouse. This metaphor is inherently modal because one needs to switch between subspaces, and disconnects the input space from the modeling space. In this paper, we propose a bimanual hand tracking system that provides physically-motivated 6-DOF control for 3D assembly. First, we discuss a set of principles that guide the design of our precise, easy-to-use, and comfortable-to-use system. Based on these guidelines, we describe a 3D input metaphor that supports constraint specification classically used in CAD software, is based on only a few simple gestures, lets users rest their elbows on their desk, and works alongside the keyboard and mouse. Our approach uses two consumer-grade webcams to observe the user's hands. We solve the pose estimation problem with efficient queries of a precomputed database that relates hand silhouettes to their 3D configuration. We demonstrate efficient 3D mechanical assembly of several CAD models using our hand-tracking system.

ACM Classification: H5.2 [Information Interfaces and Presentation]: User Interfaces–*Input devices and strategies*

General terms: Design, Human Factors, Algorithms

Keywords: computer aided design, hand-tracking, 3D object manipulation

INTRODUCTION
Computer Aided Design (CAD) is an essential component of modern mechanical engineering, architectural design and visual effects design. 3D CAD software enables a user to digitally specify shapes, positions and orientations of objects in a virtual 3D scene. Formally, most of these tasks require users to specify 3 or more variables, for instance, the xyz components of a translation or the three angles of a rotation. However, a 2D mouse is still the predominate method by which users interact with CAD software. It is typically controlled unimanually by the dominant (right) hand and used for both adjusting the camera perspective and moving objects in a 3D scene. In this paper, we propose a new input method that is complimentary to the mouse and better matches the dimensionality of 3D assembly tasks.

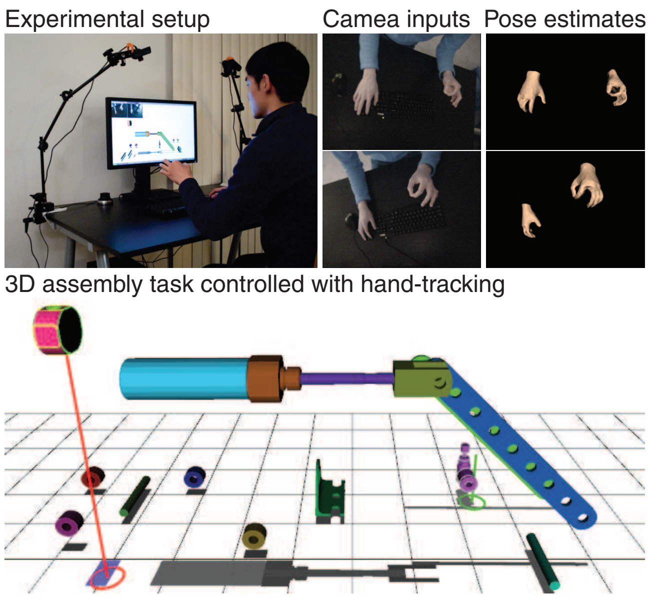

Figure 1: We propose a markerless hand-tracking system using two webcams for 3D assembly in the context of Computer Aided Design.

We facilitate more efficient interactions for CAD by enabling the user to manipulate the camera perspective and objects in the scene with both hands in 3D. We can track 6 degrees of freedom for each hand and a pinching gesture for selection. To avoid wrist-strain, we primarily use the three translational degrees of freedom of each hand tracked at the tip of the thumb. We propose comfortable, memorable and efficient gestures that map unimanual translation to 3D translation in the virtual world and bimanual translation to 3D rotation.

A complete CAD software system typically includes functionality for modeling 3D parts, assembling the parts into a whole, and, in the case of engineering CAD, laying out views of the assembly on a paper blueprint. In this work, we focus on the assembly component of CAD, which primarily involves the 3D positioning of parts, a task well suited for direct hand manipulation.

A major feature of our system is that it tracks the hands without gloves or markers, leaving them unencumbered to use the keyboard and mouse. This enables users to transition seamlessly to existing functionality in the CAD software such as navigating a menu with the mouse and typing coordinates on the keyboard. By facilitating these mixed-mode operations, our system serves as a practical complement to the mouse and keyboard for 3D assembly interactions.

RELATED WORK

Many methods have been proposed for markerless or gloveless hand tracking, but they are either too slow for interactive applications, e.g. [24, 7], or the range of poses that they can detect do not permit the precise selection required in CAD applications, e.g. [14, 13, 20]. In comparison, our system achieves bimanual 6-DOF pose estimation at interactive rates and reliably detects poses suited for discrete selection such as pinching and pointing.

Glove tracking has been proposed to ease and speed up the problem of hand tracking, e.g. [27, 25]. However, gloves are a significant drawback if one wants to also use the keyboard and mouse. Users may be reluctant to put on a glove when switching from a 2D task such as menu navigation to a 3D task such as object assembly. Wearing a glove may also become uncomfortable during long work sessions.

Dedicated 6-DOF rate-control devices such as the 3DConnexion SpaceNavigator are used for smooth camera control, but are generally not suited for or used for object selection or manipulation in CAD. We propose a set of hand gestures that works well for both camera adjustment and object manipulation.

Recently Microsoft introduced and deployed the Kinect motion capture system. While successful results have been shown for whole body tracking [21], it is unclear if the Kinect system can be used to track hands. In particular, occlusions of the fingers are difficult to resolve using a single viewpoint.

Pinching has been shown to be an effective gesture for "clicking" in 3D space. Hilliges and colleagues use a single depth camera to detect pinches above a table top [9]. Benko and Wilson [4] track pinches using an infrared camera above a projector. Wilson uses a webcam to detect pinches above the keyboard [26]. However, all three approaches rely on a single-view pinch detection technique that suffers from occlusions, restricting the hand orientations that can be tracked. A unique feature of our approach is the use of two wide-baseline viewpoints. Our two-view approach resolves occlusions from one view using information from the other, enabling robust gesture (e.g. pinch) detection. Our contribution is independent of the particular type of camera used (depth or RGB).

Recent work on 3D assembly by Kin and colleagues [11] addressed the construction of organic sets on a multi-touch surface. In organic set dressing, the artistic look and feel of a scene is more important than precise placement of parts. In comparison, we focus primarily on 3D assembly for mechanical engineering where exact relationships between parts is crucial.

A large body of HCI research has used various forms of 3D input for virtual reality applications [1, 10, 3] but studies have shown that 3D input usability is often inferior to the mouse [5, 23]. Based on this observation, we designed our system to limit the duration of 3D input sessions and to be complementary to existing input devices so that users can fall back to standard mouse+keyboard interaction at any time. We envision that our hand tracking will be used only for 3D assembly tasks where it offers a natural and intuitive three-dimensional interaction, while other tasks such as menu navigation will continue to be done with the mouse.

TRADITIONAL 3D ASSEMBLY WORKFLOW

3D assembly refers to positioning a set of 3D parts to create a larger whole. It is used to construct a set from a collection of props or to assemble mechanical pieces into a machine. It is performed in two main ways.

Coarse 3D placement is used primarily in the context of computer animation or special effects, where the location of an object is only important so much as it generates a convincing image or effect. For such placement, users mostly rely on the translation manipulator to specify position in 3D space with a 2D mouse. A translation manipulator multiplexes 3D translation onto three 1D components projected as axes on the screen (Figure 2). The user selects an arrow and drags along each axis, one at a time, to move an object to its desired location. CAD software also lets users drag objects freely in the image plane. These manipulators are good at specifying axis-aligned and in-plane translations, but require more effort for other directions. Equivalent metaphors with similar pros and cons exist for rotations.

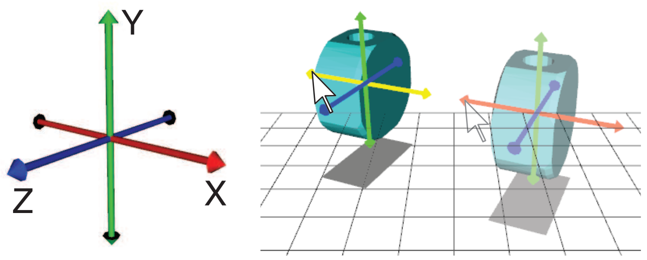

Figure 2: A traditional translation manipulator multiplexes 3D translation onto three 1-D translation modes. The user enters the x translation mode by dragging along the projected x-axis.

These manipulators provide a continuous and thus imprecise translation control. For precise placement of mechanical parts, a process of specifying constraints between faces and boundaries (or "mates") is used to define the position of a part exactly. Such positioning is crucial in manufacturing, and mating is the primary mode for 3D assembly in all mechanical engineering CAD software (e.g. SolidWorks, Autodesk Inventor, CATIA). A mate is specified with the mouse by selecting two of the features to align, and defining the relationship between them (Figure 3). For instance, if two circular boundaries are required to lie along the same axis, a user would click each boundary successively and select the concentric mate option. Specifying mates requires significant adjustments to the camera perspective because mates often involve occluded features. For instance, when specifying that two faces should be coincident, one of the faces is often facing away from the user, and thus not directly selectable.

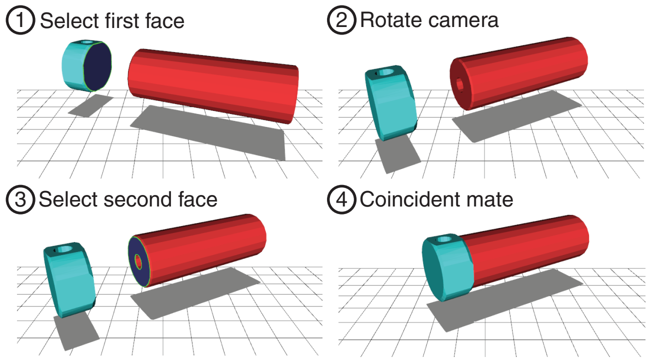

Figure 3: To specify a constraint or "mate" using a traditional 2D mouse, the user clicks on one face, rotates the camera, clicks on the other face, and indicates that they should be coincident.

In addition to manipulating objects, users also adjust the camera perspective to visualize the assembly from different view points or to select occluded features. This is classically done with the mouse using an arcball rotation control. This tool maps x and y mouse translations to the rotations around the y and x screen axes respectively. z-axis rotation, when available, is modal and involves either a different mouse button or a modifier key.

We observed several mechanical engineers using the Solidworks 2009 CAD software. Other functions often used in 3D assembly include importing parts from the file system into the scene, hiding or isolating parts in complex assemblies, and performing specialized shape modifications to a part (e.g. adding threads to a screw). These actions are accessed through mouse-driven menus, and would be more difficult to map to hand tracking.

We contribute a system that uses hand gestures for the positioning of parts for 3D assembly while other tasks such as entry of numerical values, annotations, or menu navigation are still performed with the keyboard and the mouse. Our gestures allow users to efficiently reposition pieces in 3D by selecting and moving them with their hands. Users can also adjust the camera perspective to access pieces and explore the 3D scene. Furthermore, our system enables modeless specification of exact relationships, making it suitable for the assembly of mechanical parts for engineering.

HAND GESTURES FOR 3D ASSEMBLY

Based on the observations made in the previous section, we first describe the design principles that guided how we built our system. Then, we propose a set of gestures that follow these guidelines.

Design Principles

Before describing the gestures that we implemented in our system, we first discuss the challenges that we faced while we designed our system and the high-level principles that we followed to address them.

A small number of simple hand poses Although a human hand has 27 DOFs, only a few poses are comfortable and can be reproduced without training. Guided by this idea, we built our system mostly on the pinch pose inspired by Andrew Wilson's [26] Thumb and Fore-Finger Interface. We use pinching as an analog of the mouse click to indicate the selection of an object. We also explore pointing for specifying remote locations, and touching the desktop with fingers to turn the desk surface into a multi-touch interface.

Use precise and memorable gestures To create precise and memorable gestures, we use metaphors that correspond closely to physical actions. We directly map 3D physical positions to virtual positions. Once a user understands a virtual scene, reaching for the object leverages his physical intuition to reach for a point in 3D space. We also adopt a physically-based mental model to design hand gestures for rotation. Motivated by work showing that users tend to perform rotation and translation separately [15], we decouple camera rotation and camera translation as two distinct gestures to provide precise and physically-based control for both.

Limited hand motion Unrestricted 3D interactions and large movements are tiring and only useful for short periods of time. We exploit the desktop environment to address the fatigue issue and design our system such that users can rest their elbows or forearms on the desk most of the time. Inspired by the Eden system, [11], we also allow the user to pass objects between the hands (throw-and-catch) to minimize dragging. We also amplify the user's 3D motion so that only small gestures are needed, e.g. we map a 10° hand rotation to 20° in the modeler.

Specification of exact constraints Exact constraints, or *mates*, are crucial in mechanical engineering applications to align pieces or put objects in exact contact. Specifying these mates explicitly involves selecting small features such as boundaries and faces. This is already challenging in 2D with a mouse, and even more difficult with 3D selection. Instead, we "snap" a part into place when it is sufficiently close to satisfying a concentric or a coincident mate. This feature enables our prototype CAD system to facilitate the precise alignment and contact specification required in mechanical engineering.

Concentric and coincident mates account for a large proportion of mates between mechanical parts. For instance, any mechanical assembly held together by screws and nuts use concentric and coincident mates. However, we hope to address more general mates (e.g. distance mates, curved surfaces) in future work. Presently, we fall back to traditional mouse and keyboard input for more complex mates.

Cues for 3D positioning We need to provide visual cues that relate the physical space where the user's hands are with the virtual space where modeling occurs. Since we do not want to assume that a 3D display is available, a simple "3D pointer", e.g. a small 3D arrow, is not enough because of the depth ambiguity inherent in 2D displays that makes it difficult to know if the pointer is at the same depth as another object. We address this challenge with a shadow metaphor that provides unambiguous depth cues in addition to the main 3D view. We render a virtual ground plane on which each piece projects a shadow, and we augment our pointer with a stem and base that shows its projection on the same plane [8], thereby resolving the depth ambiguity. Further, we also highlight the closest object to each hand to show which object can be selected in the current configuration.

Seamless transition between tracking and other modalities 3D assembly in CAD software is performed in conjunction with other activities that require the use of the keyboard and mouse such as structural analysis, reading documentation, or even just checking e-mail. Hand tracking should not interfere with other input modalities and should be an addition to them, not a replacement. We designed our system so that it can track bare hands since gloves and markers would impede the users' ability to type and use a mouse comfortably. We also automatically stop tracking when the hands approach the keyboard or the mouse, which allows the user to seamlessly go back to a classical keyboard-and-mouse interaction at any time.

Gestures

We support several gestures for object and camera manipulation (Figure 4). These are designed according the guidelines previously discussed.

Object translation When the user pinches with one hand, we select the closest object, and translate according to the pinching hand. If the user drags the object sufficiently close to another object, we examine the shapes of both objects and consider snapping the dragged object to generate a mate.

In our prototype CAD system, we focus on two common mates, concentric mates between circular boundaries and coincident mates between two faces. We detect approximate alignment of circular boundaries [16, 12] to snap the object into an exact alignment. Similarly, we snap together parallel faces that are almost touching to generate a coincident mate. Our 3D snapping generalizes snapping with 2D input devices [17] by generating the same mates used in CAD software.

Long-distance object translation Dragging objects is suitable for small translations but becomes tedious for large displacements, for example to go from one side of the workspace to the other. We address this with a throw-and-catch gesture akin to the Eden system [11]. The user pinches with one hand next to an object, which selects it, then pinches with the other hand where the object should go, and finally opens again the first hand, i.e. stops pinching, which transports the object where the second hand is. This gesture may be better seen in the companion video.

Camera translation When the user double-pinches with the two hands, we "attach" the camera to the two hands and translate it according to the average hand motion.

Camera and object rotation The user can control rotation by pinching with both hands, using a new *sheet-of-paper* metaphor. The motion is inspired by the grab-and-twirl gesture developed by Cutler and colleagues [6], and reproduces what users would experience if they were handling a sheet of paper as shown in Figure 5. Compared to grab-and-twirl, we remove the dependence on the hand orientation completely. Because we can track the 3D pinch points more accurately than hand orientation, we can amplify the rotated angles to minimize motion. We detail the formulas to derive the rotation angles from these gestures in the appendix.

When nothing is selected, a two-handed pinch rotates the viewpoint direction. When an object is selected with one of the hands, the rotation affects that object. Because CAD models are almost always axis-aligned, we snap object rotations to 90 degree increments.

Discussion We also experimented with a direct mapping of hand orientation to camera and object rotation. Direct orientation mapping worked well for small rotations, but large rotations led to significant strain on the wrists. This was problematic because camera viewpoint changes and object reorientations in CAD typically involve rotations of 90 degrees or more. In comparison, our two-handed pinching gestures do not require uncomfortable extreme wrist rotations and distributes effort across the larger muscles of the arm. Overall, we found our sheet-of-paper metaphor as easy-to-learn as a direct mapping while being more accurate and less straining on the wrists.

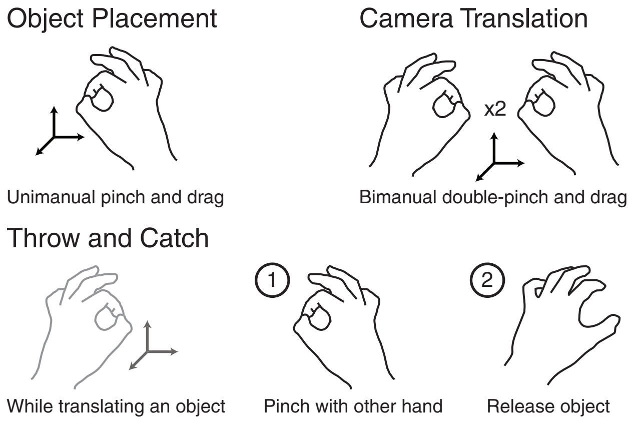

Figure 4: We support a few simple gestures for 3D manipulation and camera adjustment

MARKERLESS HAND-TRACKING

The main technical contribution of this work is a markerless hand tracking system that accurately and robustly recognizes the gestures previously described. As discussed in the previous work section, real-time markerless hand tracking is still an unsolved problem, and our approach leverages a wide-baseline camera setup, our constrained set of gestures, and the calibrated desktop environment to make this problem tractable.

Physical Setup

We designed our setup with several factors in mind. We ensure that both cameras can see the hands over a large 3D

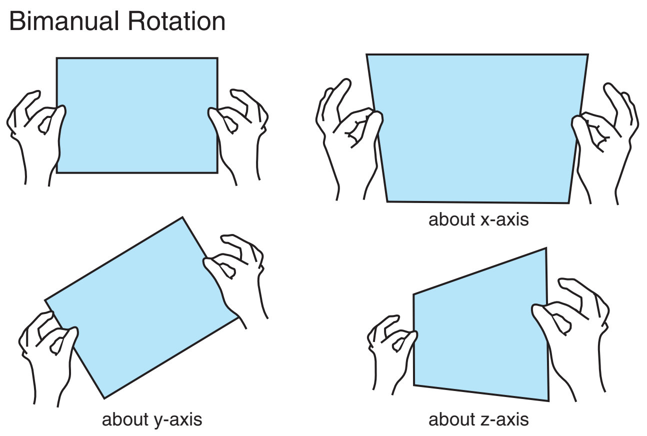

Figure 5: We mimic the physical actions of rotating an imaginary piece of paper with two hands to define our gestures for bimanual rotation. Rotating the sheet about the y or z-axis involves moving the hands in opposite directions along the xz or xy-plane respectively. To rotate the paper about the x-axis, one lifts or lowers the hands while bending the wrists (resulting in a translation along the y-axis about the elbow pivot)

capture volume. Our pose estimation algorithm benefits from two significantly different viewpoints to maximize information and minimize self-occlusion of the hand. The cameras are placed so that one hand does not occlude the other from either view. In particular, this could be a problem when one hand is raised above the other (Figure 6).

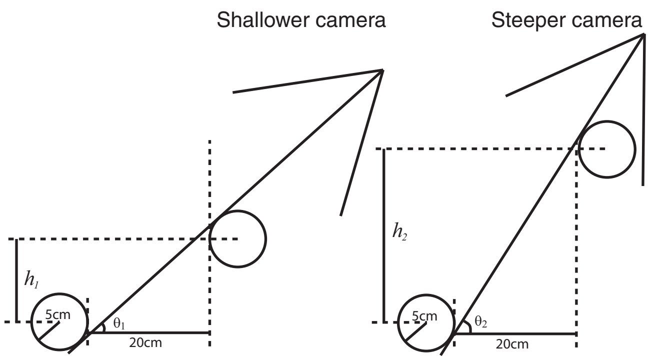

Figure 6: The most common occlusion case is when one hand (approximated as the circle on the right) is raised above another (the left circle). A steeper camera angle $\theta_2 > \theta_1$ allows a larger vertical separation $h_2 > h_1$ between the hands without occlusion.

Given these constraints, we propose a configuration such that a 34cm × 46cm × 24cm rectangular operating region is completely visible from both cameras. The cameras are placed so that their principal axes differ by an angle of $45°$, which yields significantly different viewpoints. Each camera also forms a steep $67°$ angle with the ground plane. Given a spacing of 20 cm between the two hands, this allows one hand to be 10 cm above the other without inter-hand occlusion.

Markerless Pose Estimation

Our technique applies a data-driven pose estimation algorithm inspired by the marker-based technique of Wang et al. [25]. However, using Wang's color glove would greatly impede users' interaction with other input devices such as a mouse or a keyboard, and we opted for a markerless approach, which makes the tracking problem much more challenging. We extend Wang's technique in several ways to address our needs and leverage our specific configuration. First, we sample only the limited set of hand poses relevant to our gestures rather than arbitrary configurations of the hand. We also consider only comfortable hand orientations and positions within our rectangular area. Finally, we modify the pose estimation algorithm to use two cameras rather than one.

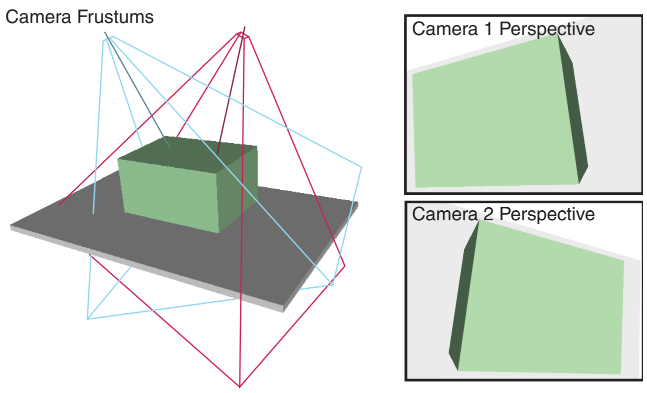

Figure 7: Our cameras are arranged so that both cameras observe the hand in a rectangular operating region (green).

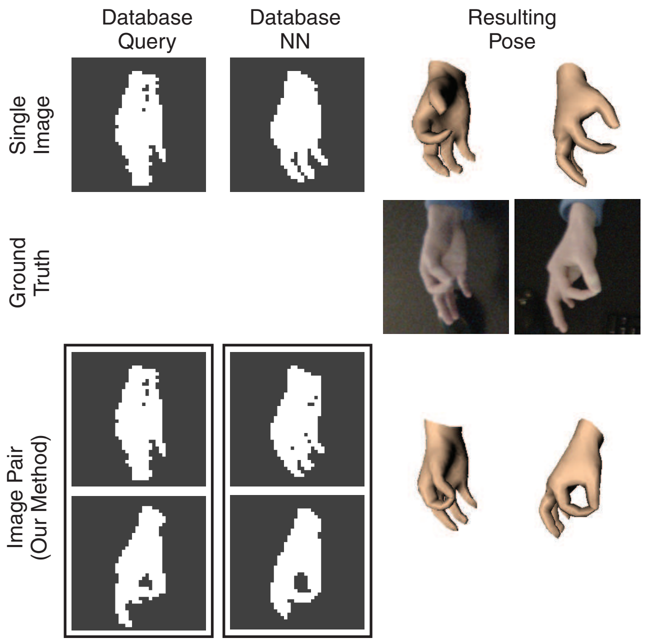

Figure 8: Given only silhouette data, a single camera view does not provide information to resolve ambiguities in orientation and finger configuration. Two cameras from different view points provides much less ambiguous data.

Two Camera Pose Estimation Our system builds upon recent work in data-driven pose estimation that uses a precomputed database to map image features to poses, e.g. [2, 19]. Specifically we adapt a technique designed to track color gloves using a database that maps tiny 40×40 pixel images of the hand to their associated 3D poses [25].

For background subtraction, we build a Gaussian mixture model for the background [22] and segment the two largest connected skin-toned regions (from the left and right hand). We encode the hand regions from each camera as a *pair* of tiny images, and query for the nearest neighbors in a precomputed database mapping hand image pairs to their associated 3D poses. Finally, we solve a 6-DOF inverse kinematics (IK) problem to place the resulting 3D hand pose to match the hand region locations (Figure 9).

A pair of hand images from different viewpoints provides much more information than a single image, but storing a database of image pairs requires twice as much memory for the same density of hand poses. To reduce memory usage, we exploit the redundancy of hand images of the image pair set. Given a set of $N = 2.1 \times 10^6$ relevant hand poses, we render these poses from two camera viewpoints to obtain tiny image pairs $\{(\mathbf{r}_i^0, \mathbf{r}_i^1)\}_{1\ldots N}$. We then sample K = 30,000 of the most different (i.e. least redundant) images from these image pairs $\{\tilde{\mathbf{r}_k}\}_{1\ldots K}$ using low dispersion sampling. These K images approximate the original set of image pairs, mapping them to a much smaller set of approximations $\{(\tilde{\mathbf{r}}_i^0, \tilde{\mathbf{r}}_i^1)\}_i$, thus making image-pair-based pose estimation efficient.

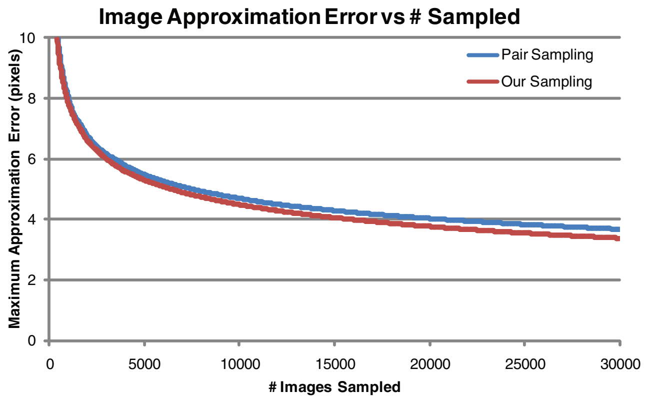

Figure 10: We compared our proposed image sampling approach that exploits redundancy between cameras and decouples sampling with a naive approach that samples pairs of images. Our approach yields a savings of 27% for equal quality at 30,000 samples.

We compared our sampling approach to a naive image-pair sampling approach that does not exploit redundancy of images between cameras (See Figure 10). Both curves converge slowly, but to achieve equal quality with 30,000 samples from the naive approach, our approach only requires 22,000 samples. To achieve equality quality with 30,000 samples taken with our approach, the naive approach would require 50% more samples (45,500).

Gesture and Desktop-Specific Sampling Because our system relies on only a few gestures, we only need to track a small set of hand poses. Hand poses outside of this relevant set are ignored during tracking because they do not correspond to actions in our CAD system. While the work of Wang and colleagues sampled a large variety of hand poses for general purpose tracking, we restrict our sampling to only the set of relevant hand poses for our gestures and for our particular desktop environment.

Our technique concentrates on the finger configurations required by our gestures. We use a relaxed hand pose, a pinching hand pose, and a pointing hand pose as a finger configuration basis $\{\mathbf{q}_i\}$. We then take pairwise blends between each of these poses $\{\mathbf{q}|\mathbf{q} = \alpha\mathbf{q}_i + (1-\alpha)\mathbf{q}_j\}$ for $\alpha \in [0, 1]$ to generate a dense set of finger configuration transitions between basis poses.

Our model assumes that the hand is in a $34\text{cm}\times 46\text{cm}\times 24\text{cm}$ box above the keyboard, and we only sample hand positions in this region. The principal axis of the hand can move in a cone subtending an angle of $\theta = 60°$ and the hand can twist about that axis $\phi = 130°$. While most people can twist their hands more, these ranges are sufficient to cover the set of comfortable poses. With this restrictive sampling, we are able to make markerless hand-tracking feasible.

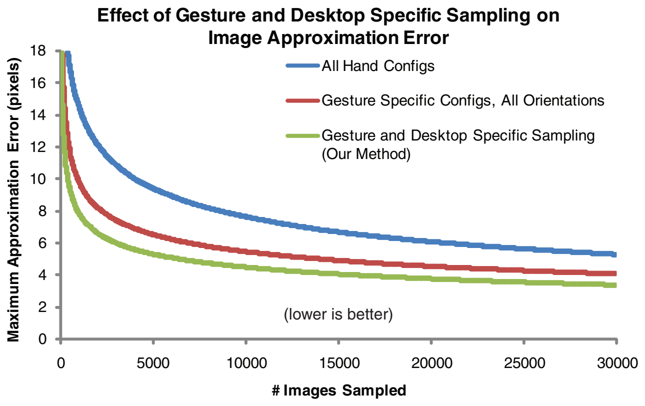

Figure 11: Our gesture-specific and desktop-specific sampling allows us to use as much as 80% fewer samples to achieve equal sampling quality with a gesture and desktop agnostic approach.

In Figure 11, we quantify the benefits of our gesture-specific, desktop-specific sampling, compared to sampling all hand configurations and all orientations, many of which are impossible given the user and camera placement. A gesture-agnostic sampling requires 30,000 samples to achieve equal quality with 5,000 samples taken with our approach. A desktop-agnostic sampling requires 30,000 samples to achieve equal quality with 15,000 samples from our approach.

Pinch / Click Detection A robust pinch detector is the basis of our gestures, and we address it separately from 3D tracking. Our pinch detection is based on detecting separation of the tips of the index finger and thumb in at least one of the two camera images. First, we check for extrema [18] of the silhouette close to the predicted locations of the index finger and thumb from our 3D pose estimate. Thumb-index separation is detected if the geodesic distance between the extrema is longer than the Euclidean distance. If no separation is detected in either view, we register a pinch (Figure 12).

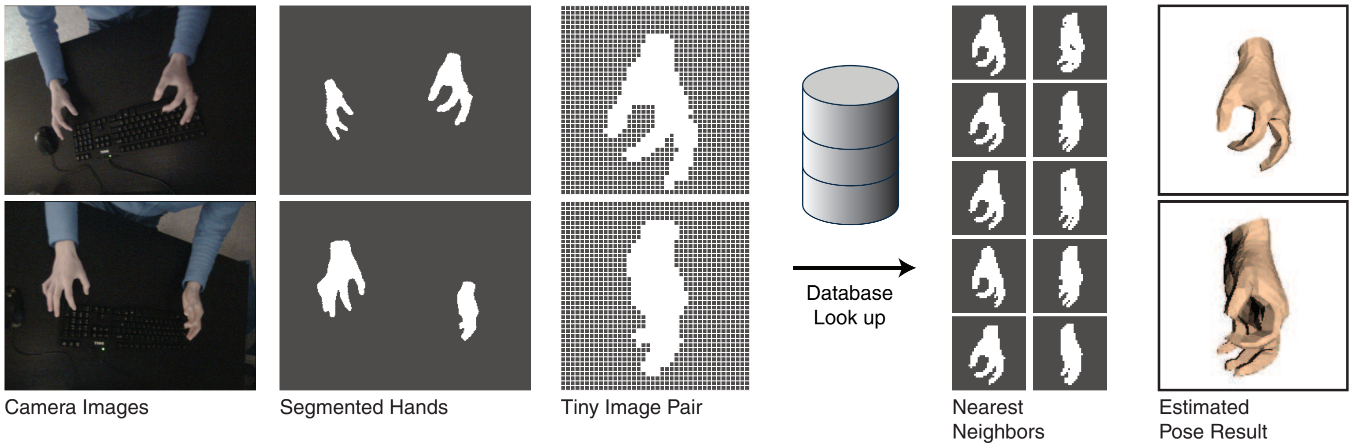

Figure 9: We segment the hands using background subtraction and skin-tone detection. We take the resulting hand regions and encode them as tiny images. We use a pair of tiny images to query the database, blend the nearest neighbors, and solve a 6-DOF IK problem to obtain the 3D hand pose.

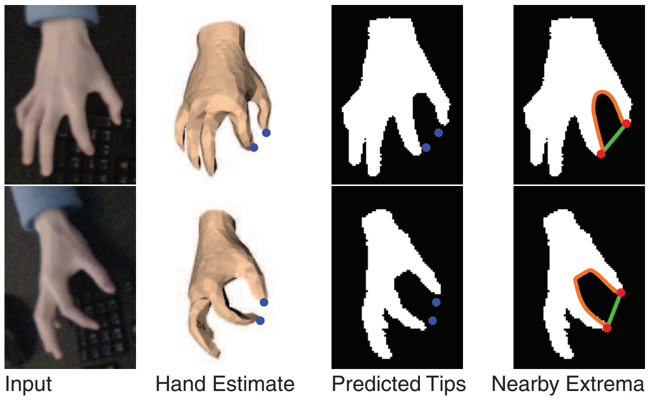

Figure 12: We look for extrema (red points) in the silhouette that match the predicted locations of the thumb and index finger tips (blue points). If the geodesic distance (orange line) between the extrema is larger than the Euclidean distance (green line), we detect a separation.

To evaluate our approach, we recorded 8 sequences from 4 people consisting of pinching gestures performed at various locations in the capture volume. We found that we could track pinching with about 99% accuracy. Further analysis, showed that the occasional missed detections and false pinch detections mostly occur at extreme poses or at extreme positions of our capture volume. As demonstrated in our companion video, our robust detection mechanism enables complex interaction.

Limitations and User Calibration Our computer-vision-based approach is currently limited to environments where the hands can be segmented from the background with skin-tone detection and background subtraction. Presently, we also ask the user to wear a long-sleeved top because we do not model the skin-toned arms. Both of these issues can be resolved with the use of depth cameras, and we hope to explore two-depth-camera setups in future work.

Unlike the technique of Wang and Popović, our database sampling is specific to the camera setup and needs to be regenerated when the cameras are moved. This requires approximately 30 minutes.

For best results, we also manually calibrate three hand poses of the user. Differently shaped hands tend to pinch and point differently. We kinematically adjust the joint angles of our basis poses to reflect the user-specific pinching, pointing and resting gestures. This takes approximately ten minutes of manual calibration, and we hope to incorporate a more accurate and automatic technique in the future [7].

EVALUATION

We tested our system on an Intel Core i7 desktop computer with two PlayStation 3 Eye cameras. Our hand-tracking system runs at interactive rates (20 Hz) on two threads of the processor. Our complete 3D assembly system runs at a slower rate of 17 Hz on the models we tested.

We measured precision (jitter) of the tracked thumb tips to be approximately 3mm. Using known rotation sequences, we also estimated the rotation precision to be between 1 and 3 degrees around the x and y axes, and about 6 degrees around the z axis.

We tested our assembly system on several models used to teach 3D assembly included as part of the SolidWorks CAD software. We show that these models can be efficiently assembled using hand gestures alone. In the hands of an expert user of both our system and a traditional CAD mating interface, we have observed time savings of up to 40%. This comes from significantly fewer modal transitions from selecting parts to adjusting the camera. We also evaluated our bimanual rotation gestures on a rotation-matching task. While not quite as efficient as an expert arcball user, our bimanual gestures provides a comfortable and easy-to-use rotational mapping. In another sequence, we show that our system works well alongside a mouse, keyboard and 3DConnexion SpaceNavigator.

In an early pilot study with three novice users, we found that due to the lack of tactile feedback, visual feedback is crucial for keeping the user in sync with the state of the assembly system. Users would often mis-select objects or confuse being in object translation mode versus camera rotation mode. In response, we gave each assembly action, including camera rotation, camera translation, object selection, and object release, a distinct visual cue. For instance, we draw a bounding box around the scene and a 3D manipulator for camera rotation and translation. We higlight the active object with a golden halo when it has been selected for manipulation.

We also found that shadows were an insufficient depth cue for 3D object selection on a 2D display. Users would often select a different object than the one they had intended. We addressed this by brightening objects near each hand and always outlining the closest object that would be selected if the user were to pinch. These cues inform the user about the relative position of his hands in the scene without having to look down at shadows before selecting an object, reducing the incidence of mis-selections.

Extensions

In addition to 3D CAD assembly, our hand-tracking system can be used to turn the desk plane into a touch-sensitive device. We use our two-camera setup to triangulate the 3D location of the index finger and detect intersection with the calibrated desk plane. We demonstrated this system on a 2D multi-touch photo rotation task. While our touch detection is only accurate within five millimeters, this is sufficient for simple multi-touch applications at the desktop.

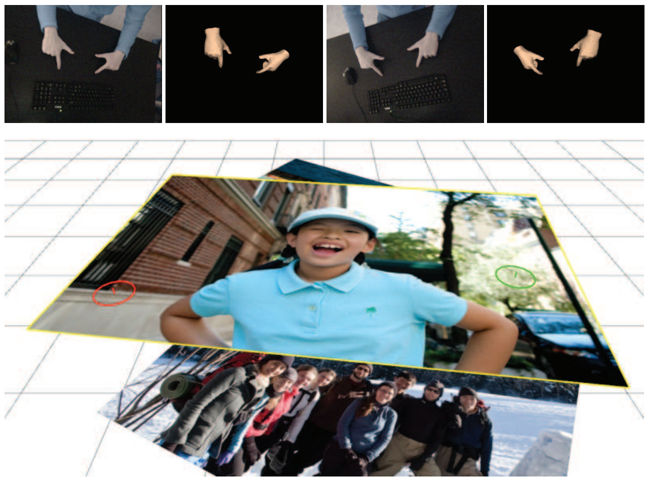
Figure 13: We detect touch with a calibrated desk to enable a virtual multi-touch surface.

CONCLUSION

We have developed a hand-tracking system tailored for a fundamental task in computer aided design, 3D assembly. Our system relies on a small set of gestures that are comfortable to use, precise, and easy to remember. To recognize these gestures, we built a data-driven pose-estimation system that uses a wide-baseline camera setup and an efficiently sampled database to track the hands without markers, alongside the keyboard and mouse. Furthermore, our prototype CAD system generates the exact positioning constraints used in traditional CAD for mechanical engineering. In summary, we have developed a complementary user input device for efficient 3D assembly of mechanical parts.

REFERENCES

1. Maneesh Agrawala, Andrew C. Beers, Bernd Fröhlich, Patrick M. Hanrahan, Ian McDowall, and Mark Bolas. The two-user responsive workbench: Support for collaboration through independent views of a shared space. In *Proceedings of SIGGRAPH 97*, pages 327–332, 1997. 2

2. Vassilis Athitsos and Stan Sclaroff. Estimating 3D hand pose from a cluttered image. In *Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 432–439, 2003. 6

3. Hrvoje Benko and Andrew Wilson. Depthtouch: Using depth-sensing camera to enable freehand interactions on and above the interactive surface. Technical Report MSR-TR-2009-23, Microsoft, 2009. 2

4. Hrvoje Benko and Andrew D. Wilson. Multi-point interactions with immersive omnidirectional visualizations in a dome. In *Interactive Tabletops and Surfaces (ITS)*, pages 19–28, 2010. 2

5. Francois Bérard, Jessica Ip, Mitchel Benovoy, Dalia El-Shimy, Jeffrey Blum, and Jeremy Cooperstock. Did "Minority Report" Get it Wrong? Superiority of the Mouse over 3D Input Devices in a 3D Placement Task. *Human-Computer Interaction–INTERACT 2009*, pages 400–414, 2009. 2

6. Lawrence D. Cutler, Bernd Fröhlich, and Pat Hanrahan. Two-handed direct manipulation on the responsive workbench. In *Proc. of Interactive 3D Graphics (I3D)*, 1997. 4

7. Martin de La Gorce, Nikos Paragios, and David J. Fleet. Model-Based Hand Tracking with Texture, Shading and Self-occlusions. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008. 2, 7

8. Michael Glueck, Keenan Crane, Sean Anderson, Andres Rutnik, and Azam Khan. Multiscale 3D reference visualization. In *Proc. of Interactive 3D Graphics and Games (I3D)*, pages 225–232, 2009. 4

9. Otmar Hilliges, Shahram Izadi, Andrew D. Wilson, Steve Hodges, Armando Garcia-Mendoza, and Andreas Butz. Interactions in the air: adding further depth to interactive tabletops. In *User Interface Software and Technology (UIST)*, pages 139–148, 2009. 2

10. Daniel F. Keefe, David B. Karelitz, Eileen L. Vote, and David H. Laidlaw. Artistic collaboration in designing vr visualizations. *IEEE Computer Graphics and Applications*, 25(2):18–23, 2005. 2

11. Kenrick Kin, Tom Miller, Björn Bollensdorff, Tony DeRose, Björn Hartmann, and Maneesh Agrawala. Eden: A professional multitouch tool for constructing virtual organic environments. In *Human Factors in Computing Systems (CHI)*, 2011. 2, 3, 4

12. Wilmot Li, Maneesh Agrawala, Brian Curless, and David Salesin. Automated generation of interactive 3D exploded view diagrams. *ACM Trans. on Graphics*, 27(3), 2008. 4

13. S. Malik and J. Laszlo. Visual touchpad: a two-handed gestural input device. In *Int. Conf. on Multimodal interfaces*, pages 13–15, 2004. 2

14. Cristina Manresa, Javier Varona, Ramon Mas, and Francisco J. Perale. Real-time hand tracking and gesture recognition for human-computer interaction. *Electronic Letters on Computer Vision and Image Analysis*, 2000. 2

15. Maurice Masliah and Paul Milgram. Measuring the allocation of control in a 6 degree-of-freedom docking experiment. In *Human Factors in Computing Systems (CHI)*, pages 25–32, 2000. 3

16. Niloy J. Mitra, Yong-Liang Yang, Dong-Ming Yan, Wilmot Li, and Maneesh Agrawala. Illustrating how mechanical assemblies work. *ACM Trans. on Graphics*, 29(3), 2010. 4

17. Ji-Young Oh and Wolfgang Stuerzlinger. Moving objects with 2D input devices in cad systems and desktop virtual environments. *Proc. of Graphics Interface*, pages 195–202, 2005. 4

18. Christian Plagemann, Varun Ganapathi, Daphne Koller, and Sebastian Thrun. Real time identification and localization of body parts from depth images. In *Int. Conf. on Robotics and Automation (ICRA)*, 2010. 6

19. Javier Romero, Hedvig Kjellstrm, and Danica Kragic. Hands in action: real-time 3D reconstruction of hands in interaction with objects. In *Int. Conf. on Robotics and Automation (ICRA)*, pages 458–463, 2010. 6

20. Markus Schlattmann and Reinhard Klein. Efficient bimanual symmetric 3D manipulation for markerless hand-tracking. In *Virtual Reality International Conference (VRIC)*, 2009. 2

21. Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. Real-Time Human Pose Recognition in Parts from Single Depth Images. In *Computer Vision and Pattern Recognition (CVPR)*, 2011. 2

22. Chris Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 246–252, 1999. 6

23. Wolfgang Stuerzlinger and Chadwick Wingrave. The Value of Constraints for 3D User Interfaces. *Dagstuhl Seminar on VR*, 2010. 2

24. Erik B. Sudderth, Michael I. Mandel, T. Freeman, and S. Willsky. Distributed occlusion reasoning for tracking with nonparametric belief propagation. In *Neural Information Processing Systems (NIPS)*, pages 1369–1376, 2004. 2

25. Robert Y. Wang and Jovan Popović. Real-time handtacking with a color glove. *ACM Trans. on Graphics*, 28(3):1–8, 2009. 2, 5, 6

26. Andrew D. Wilson. Robust computer vision-based detection of pinching for one and two-handed gesture input. In *User Interface Software and Technology (UIST)*, pages 255–258, 2006. 2, 3

27. Thomas G. Zimmerman, Jaron Lanier, Chuck Blanchard, Steve Bryson, and Young Harvill. A hand gesture interface device. In *Human Factors in Computing Systems (CHI)*, pages 189–192, 1987. 2

Appendix: Formulas for camera rotation

Given previous click points $(\mathbf{p}_0, \mathbf{p}_1)$ and current click points $(\mathbf{p}'_0, \mathbf{p}'_1)$, the bimanual rotation is specified by $R_1 R_2$. The rotation R_1 corresponds to the elbow rotation about the x-axis of $\arctan((((\mathbf{p}'_0 + \mathbf{p}'_1) - (\mathbf{p}_0 + \mathbf{p}_1))/2)_y, f)$, where f is the forearm length. The rotation R_2 corresponds to rotating the sheet about its center axis. Let the normalized vector between the click points be $\mathbf{n} = (\mathbf{p}_1 - \mathbf{p}_0)/\|\mathbf{p}_1 - \mathbf{p}_0\|$ and $\mathbf{n}' = (\mathbf{p}'_1 - \mathbf{p}'_0)/\|\mathbf{p}'_1 - \mathbf{p}'_0\|$. Then R_2 is a rotation about the sheet's center axis $\mathbf{n} \times \mathbf{n}'$ with an angle of $\arctan(\|\mathbf{n} \times \mathbf{n}'\|, \mathbf{n} \cdot \mathbf{n}')$.