Leveraging Depth Data in Remote Robot Teleoperation Interfaces for General Object Manipulation

Journal Title
XX(X):1–14

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

David Kent[1], Carl Saldanha[1], and Sonia Chernova[1]

Abstract

Robust remote teleoperation of high-DOF manipulators is of critical importance across a wide range of robotics applications. Contemporary robot manipulation interfaces primarily utilize a Free Positioning pose specification approach to independently control each degree of freedom in free space. In this work, we present two novel interfaces, Constrained Positioning and Point-and-Click. Both novel approaches incorporate scene information from depth data into the grasp pose specification process, effectively reducing the number of 3D transformations the user must input. The novel interactions are designed for 2D image streams, rather than traditional 3D virtual scenes, further reducing mental transformations by eliminating the controllable camera viewpoint in favor of fixed physical camera viewpoints. We present interface implementations of our novel approaches, as well as Free Positioning, in both 2D and 3D visualization modes. Additionally, we present results of a 90-participant user study evaluation comparing the effectiveness of each approach for a set of general object manipulation tasks, and the effects of implementing each approach in 2D image views versus 3D depth views. The results of our study show that Point-and-Click outperforms both Free Positioning and Constrained Positioning by significantly increasing the number of tasks completed and significantly reducing task failures and grasping errors, while significantly reducing the number of user interactions required to specify poses. Additionally, we found that regardless of interaction approach, the 2D visualization mode resulted in significantly better performance than the 3D visualization mode, with statistically significant reductions in task failures, grasping errors, task completion time, number of interactions, and user workload, all while reducing bandwidth requirements imposed by streaming depth data.

Keywords

robot teleoperation; object manipulation; user interface design; usability study; rgbd interfaces

1. Introduction

Robust remote teleoperation of high-DOF manipulators is of critical importance across a wide range of applications, including assistive robotics, space exploration, search-and-rescue, and web-based robot control. As use cases for remote teleoperation expand beyond pick-and-place to more complex manipulation tasks, there is a growing need to develop effective user interfaces that minimize operator error and improve task efficiency.

Most mouse-and-keyboard robot manipulation interfaces currently utilize a 6-DOF ring-and-arrow marker for independent control of each axis of rotation and translation. The marker can be used to directly move an end-effector or set pose goals to which the robot can autonomously plan and move. Leeper et al. (2012) evaluated multiple interfaces based on the ring-and-arrow marker design, showing that increasing robot autonomy results in better performance in object picking tasks.

In this work, we view interfaces for remote robot control as falling on a spectrum of reliance on scene information, in the form of depth data. The ring-and-arrow marker does not require the use of depth data, giving robot operators complete control over the pose of their robot's end-effector in free space. We refer to this approach as *Free Positioning*. While this degree of flexibility gives users the maximum level of control, the size of the workspace and the number of controllable 3D transformations may become a burden that can increase the likelihood of operator error. Our work compares the Free Positioning approach to two alternate approaches for human-in-the-loop robot manipulation that leverage depth information to isolate, combine, or reduce the number of 3D transformations for which the user is responsible. We focus specifically on mouse-and-keyboard approaches, as these input devices are readily available to most of the population. We introduce two alternate approaches to Free Positioning – *Constrained Positioning*, which uses depth information at a single point-of-interest in the environment, and *Point-and-Click*, which proposes grasp poses based on local 3D surface geometry – to explore the trade-offs inherent to the use of depth data in a general object manipulation context. The presented techniques do not use scene recognition or object recognition; as a result, they can be immediately deployed in new environments without training. We evaluate all three interaction approaches over a

[1]Georgia Institute of Technology, USA

Corresponding author:
David Kent, Georgia Institute of Technology, Atlanta, Georgia 30332, USA
Email: dekent@gatech.edu

variety of manipulation tasks, from pick-and-place to more complex tasks including opening containers and pouring.

The ring-and-arrow marker approach is typically rendered in a 3D visualization of the robot's environment. In contrast, both of our novel techniques algorithmically incorporate depth data into the pose specification process, which allows us to remove the depth data from the interface. As a result, the novel interaction approaches can be used with an overlay over a 2D image view, rather than a fully realized 3D depth view, further reducing the degrees of freedom of the interface by removing a user-controlled virtual camera. Using only 2D image streams also significantly reduces bandwidth requirements for remote interfaces, as the system need only stream rgb images rather than point clouds. This raises the question, however, of whether removing the depth data visualization from a user interface would reduce situational awareness to the point that performance degrades. To answer this question, we evaluate all three approaches implemented with both a 2D and 3D visualization mode.

This paper makes three contributions to teleoperation for manipulation. The first is the Constrained Positioning approach–a novel interaction approach for positioning a 6-DOF end-effector. We designed this method to incorporate a small amount of scene information (a single point) into a polar (rather than Cartesian) coordinate frame approach that provides users with fine-grained control over grasp pose specification. The result is a constrained step-by-step approach that allows users to set a grasp point, approach angle, and grasp depth with a reduced number of mental transformations.

The second contribution is the Point-and-Click approach–a novel interaction approach that extends the AGILE grasp approach (ten Pas and Platt 2015) for general object manipulation. We present a heuristic-based technique that replaces AGILE's classification step, resulting in an autonomous grasp calculator better suited to manipulating man-made objects during human-in-the-loop operation. We leverage the resulting approach to create an interaction method that incorporates a large amount of scene information (a patch of local surface geometry), requiring only supervisory control from a user.

Our final contribution is a usability comparison of the above human-in-the-loop object manipulation approaches, implemented with both 2D and 3D visualization modes. We present the results of a 90-participant user study that compare the efficiency and effectiveness of the Constrained Positioning and Point-and-Click approaches with the Free Positioning ring-and-arrow marker approach. Our results show with statistical significance that the Point-and-Click interaction approach outperforms the other two approaches, allowing users to complete more manipulation tasks. Both novel approaches significantly reduce the number of grasping errors, with Point-and-Click significantly reducing task completion time, the number of task failures, and the number of interactions required by the user. Furthermore, regardless of interaction approach, using 2D visualization modes significantly reduces task failures, grasping errors, completion time, number of interactions, and user workload, while alleviating bandwidth requirements imposed by streaming depth data.

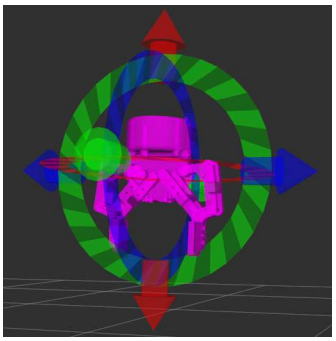


Figure 1. 6-DOF positioning ring-and-arrow marker

2. Related Work

Unconstrained methods allowing individual control of task space translation and rotation are widely used in robotics. Gossow et al. (2011) initially present an interactive marker composed of rings and arrows (Figure 1) for just such a purpose. This marker can be rendered in a 3D scene such as rviz (Quigley et al. 2009) or online via Robot Web Tools' ros3djs (Toris et al. 2015). Leeper et al. (2012) extensively examined the ring-and-arrow marker in multiple robot manipulation use cases in a user study. It's used at varying levels of robot autonomy, including (in order of increasing robot autonomy) a) real-time arm position control, b) setting waypoints executed through linear interpolation, c) setting pose goals followed by autonomous trajectory execution with obstacle avoidance, and d) adjusting autonomously calculated grasp poses for fully autonomous grasp execution. Leeper et al. found the ring-and-arrow marker to be an effective tool for an object picking task, noting that users of the interface made fewer mistakes as the autonomy of the robot increased.

Ciocarlie et al. (2012) showed the ring-and-arrow marker could be used for general mobile manipulation in household environments. Chen et al. (2013) also uses these 6-DOF markers for teleoperating a robot to perform everyday tasks as a part of the Robots for Humanity project. The same markers are used for start and goal pose specification in the MoveIt! rviz plugin (Chitta 2016). Many teams competing in the DARPA Robotics Challenge used this same marker for multiple purposes, including positioning key points of a humanoid robot for balance or manipulation (Zucker et al. 2015; Kohlbrecher et al. 2015; DeDonato et al. 2015), positioning 3D object templates over sensor data (Kohlbrecher et al. 2015), and correcting point cloud alignment (Phillips-Grafflin et al. 2014). The ring-and-arrow marker has also found use in web interfaces. In previous work (Kent et al. 2016), we used the marker in an interface for crowdsourcing object recognition and picking. More recently, Kemp and Grice (2016) showed similar controls superimposed on a camera stream displayed as an overlay on a web interface, rather than rendering them in a 3D environment.

While Leeper et al. (2012) found the ring-and-arrow marker to be most effective when initializing it at a clicked grasp point (as in the initialization of our Constrained Positioning approach) or at a calculated grasp position, we consider this an unconstrained method, as the user can then

move the grasp pose anywhere in free space. The ring-and-arrow marker is also fully functional in the absence of depth data, which is important as depth information around the grasp point-of-interest is not always available due to poor sensor readings arising from difficult materials or cluttered environments. In practice, we find that instead of initializing the ring-and-arrow marker at a user-selected grasp point, the majority of the work described above initializes the marker either at the current pose of the robot (Zucker et al. 2015; Kohlbrecher et al. 2015; DeDonato et al. 2015; Kent et al. 2016; Kemp and Grice 2016; Chitta 2016) or in free space (Phillips-Grafflin et al. 2014; Kohlbrecher et al. 2015). We denote this approach Free Positioning, and use it as our baseline interaction approach, initializing the marker at the current pose of the robot.

Despite its widespread use, the ring-and-arrow approach requires the user to perform many mental 3D rotations, which Zhai and Milgram (1998) show to be mentally taxing. When the marker is rendered in a 3D environment, the user must control both the transformation of the camera within the 3D environment and the transformation of the 6-DOF marker itself. Additionally, use of the marker in a 3D environment requires bandwidth-intensive streaming of depth data, making this 3D visualization unappealing for remote teleoperation interfaces. Conversely, when projected onto a 2D camera feed, manipulating the marker causes it to become unaligned with the user's viewing angle, which Parsons (1995) identifies as a particularly difficult case requiring complex mental transformations. There are two approaches to alleviate this difficulty: Masliah and Milgram (2000) suggest isolating transformations wherever possible (e.g. separating rotation and translation), and DeJong et al. (2004) suggest reducing the number of transformations that the user is required to make. Both of these approaches can be accomplished by incorporating more scene information into grasp pose specification methods, and as such they inspire the Constrained Positioning and Point-and-Click methods contributed in our work. Further, by designing our methods with 2D visualization modes in mind, we remove the additional transformations involving the virtual camera inherent to 3D visualization.

Alternative control methods for 6-DOF object positioning have been studied primarily outside the field of robotics. By using a smartphone as an input device, researchers have shown multiple interfaces for controlling objects in 3D video games (Katzakis et al. 2011) and editing meshes in 3D space (Henrysson and Billinghurst 2007). These approaches typically involve separating translations and rotations by assigning them to either the touch screen or the motion of the phone itself. Within the field of robotics, natural 6-DOF end-effector control methods have been studied recently using motion tracking devices developed for Augmented Reality (AR) and Virtual Reality (VR). These approaches make use of hand tracking (Cancedda et al. 2017), wearable devices (Peppoloni et al. 2015), and VR controllers (Rakita et al. 2017; Lipton et al. 2018), showing feedback to the user in 3D UIs, AR, and VR interfaces (Peppoloni et al. 2015; Cancedda et al. 2017; Lipton et al. 2018). Motion capture techniques have been shown to be effective for pick-and-place and other manipulation actions due to the natural interaction of simply moving one's arms, and the immersive qualities of AR and VR can aid in situational awareness. The disadvantage of these techniques is that they require specialized hardware, although VR controllers are becoming more widely available as consumer products. These interfaces are all designed for direct control, however, which degrade as latency increases, making them less suitable to remote teleoperation.

Alternative mouse-and-keyboard approaches are less common. Hachet et al. (2008) present Navidget, a tool for free-space camera positioning developed specifically for 2D input devices, provides a touchscreen-based method of 6-DOF control that translates nicely to a mouse and computer monitor. It separates translations and rotations into a multi-step process, while additionally reducing the total number of positioning operations by fixing the interaction to a local area and a selected approach angle. As this practically incorporates many of the suggestions from teleoperation research rooted in cognitive science (Masliah and Milgram 2000; DeJong et al. 2004), we base our Constrained Positioning approach on Navidget.

Another approach to grasp pose specification is to completely remove the requirement for specifying transformations by autonomously calculating grasp poses. This is a difficult problem for novel and unrecognized objects, particularly because point cloud data provides only partial object shape information. (Hsiao et al. 2010) present a heuristic-based approach, biasing grasps to orthogonal overhead or side poses. Other work uses classifiers with hand-specified features. Saxena et al. (2008) use contacts, symmetry, and force-closure as features to classify grasp stability. The AGILE grasp approach by ten Pas and Platt (2015) efficiently samples and classifies antipodal grasps. Another algorithm by ten Pas and Platt (2016) detects grasping affordances to identify handle-like geometry. Further examples of hand-engineered features over depth data can be found in Jiang et al. (2011) and Fischinger et al. (2015). Deep learning provides another classification approach which learns robust grasp features by generating large amounts of data through trial-and-error (Levine et al. 2016; Lenz et al. 2015; Pinto and Gupta 2016), or from adversarial synthetic data (Mahler et al. 2017). For our Point-and-Click approach, we use a heuristic-based extension to AGILE's antipodal grasp sampling to create a supervisory control grasp pose specification interface.

3. Interaction Approaches

In this work, instead of focusing on a single interface, we examine a spectrum of grasp pose specification approaches for remote manipulation interfaces. Based on our literature review, we identified three interaction approaches for specifying 6-DOF end-effector poses. The three approaches, which we discuss further in this section, are: *Free Positioning* (*FP*), *Constrained Positioning* (*CP*), and *Point-and-Click* (*P&C*)*. Figure 2 provides an overview of the three approaches. We selected these three techniques to

*We have released our three interaction approaches in ROS under the `remote_manipulation_markers` package. Details of the package and open source code can be found at the ROS wiki page, `http://wiki.ros.org/remote_manipulation_markers`, and a video demonstrating our three approaches can be seen at `https://youtu.be/4NTyDVmpXRY`

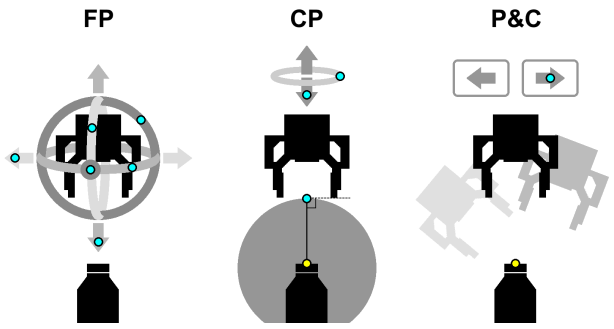


Figure 2. Interaction approaches arranged by number of major interaction points and reliance on depth data. Free Positioning does not require depth data, Constrained Positioning requires a single point, and Point-and-Click requires local surface geometry. Yellow points represent setting initial grasp poses; blue points represent pose refinement

incorporate increasing use of scene information through depth data into the grasp specification process. We hypothesize that incorporating more scene information into the pose specification algorithms will reduce the number of interactions and the time required to specify a grasp pose, as well as lower the cognitive workload of the user by offloading some decision making to the algorithms.

After pose specification, the three approaches handle grasp execution in the same manner. Once the user confirms the pose, the robot uses the same motion planning algorithm and execution control loop to move to the specified location. All three approaches require at most a point cloud as input, and do not rely on object recognition. As such they can be used in novel environments with no need for training time on new objects.

3.1. Free Positioning

Our first approach allows the user to position the robot's end-effector in any 6-DOF pose. We use the standard ring-and-arrow marker and base the approach on the Grasp Execution strategy described by Leeper et al. (2012). As such, the user sets a pose by clicking and dragging on any of the three arrows or rings corresponding to the three Cartesian axes of translation and rotation, respectively. Once the grasp pose is set, the arm autonomously plans and executes an obstacle-avoiding trajectory resulting in a grasp at the specified pose.

We chose this supervisory autonomous planning and execution approach over direct control for two reasons. First, Leeper et al. showed approaches in which the robot performs more autonomous trajectory planning and execution result in better performance (i.e. more successful grasps and fewer collisions). Second, since our system is targeting remote teleoperation, latency is an important concern. Direct control is highly sensitive to latency as it requires real-time visual feedback from the robot's sensors, whereas supervisory methods are virtually invariant to latency, since all of the execution is performed on-board (Sheridan 1993). All ring-and-arrow interactions are performed client-side, so they are not limited by latency.

3.2. Constrained Positioning

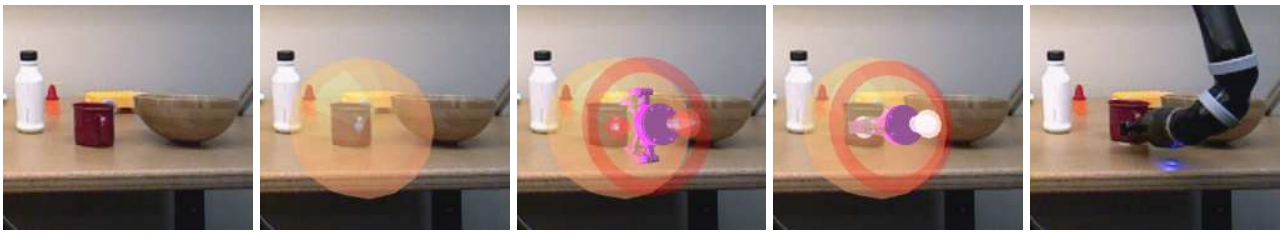
Our second approach seeks to separate the necessary pose transformations in an intuitive manner, as well as to reduce the number of transformations wherever possible. We base our approach on the Navidget interaction technique (Hachet et al. 2008) designed for 3D camera positioning. Camera positioning is a useful analogy for end-effector positioning, as both involve specifying a 3D transformation using 2D mouse inputs. Extending a Navidget-like approach to robot grasping results in a novel interaction method for constrained end-effector positioning, based on a polar coordinate frame defined at a selected point-of-interest, rather than a Cartesian coordinate frame defined at the end-effector goal pose. The Constrained Positioning approach is a 3-step process, visualized in Figure 3 and explained below.

1. *Select a grasp point.* The user first specifies a point-of-interest, in this case a grasp point, by clicking on a stream of camera data. This constrains the final grasp pose to one that must pass through the selected point.
2. *Set an approach angle.* Once the grasp point is selected, Constrained Positioning renders a sphere around the point-of-interest. The user can then set an approach angle with a single click on the sphere's surface. The approach angle is determined by calculating a pitch and yaw angle that create an orthogonal vector to the surface of the sphere. In this way, the user can specify two degrees of rotation in a single action.
3. *Adjust the roll and grasp depth.* Navidget does not include a method for controlling the camera's roll, and while it does include a method for setting the depth of the camera, it does not translate well to robot grasping. As such, we include a single ring and arrow marker so that the user can adjust the wrist roll and the depth of the grasp by clicking and dragging the ring and arrow, respectively.

By using a small amount of scene information (the selected point-of-interest), and constraining the grasp pose to pass through the point-of-interest, we are able to reduce the number of transformations the user is required to input from 6 degrees of freedom (3 rotations and 3 translations) to 4 degrees of freedom (3 rotations and a single translation). Furthermore, the sphere metaphor allows us to group 2 rotations into a single action with an intuitive visual-spatial representation. The two rotations are specified with a single click on a 2D surface, making the approach more suitable to 2D input than the 6-DOF ring-and-arrow marker, which requires clicking and dragging multiple 3D-rendered controls to specify the same approach angle rotations.

3.3. Point-and-Click

Our third approach autonomously calculates grasp poses from a single mouse click. The approach is similar to the Grasp Planning strategy presented in Leeper et al. (2012), although we incorporate new advances in autonomous grasp calculation, which we extend to be more suitable for a human-in-the-loop approach. We base the grasp calculator on the AGILE grasp method (ten Pas and Platt 2015). The original AGILE algorithm has two steps. First, it samples



(a) Original image stream

(b) Clicking the mug handle selects it as the grasp point

(c) Clicking on the sphere sets an approach angle

(d) Ring and arrow markers can be used to adjust wrist roll and grasp depth

(e) The robot executes the final grasp

Figure 3. Constrained Positioning steps

a set of antipodal gripper hypotheses from a point cloud within a given workspace. The sampling algorithm samples the space around the point cloud to find gripper poses that do not intersect the point cloud, but do contain a subset of points between the gripper's fingers. Since these gripper poses are calculated from a point cloud, the algorithm assumes some shape information is missing due to occlusion. To address this, the second step of the algorithm is to classify the gripper pose hypotheses as antipodal or not, using a HOG descriptor to encode the points contained within each gripper pose hypothesis.

Our extension to AGILE replaces the antipodal grasp pose classifier with a heuristic-based approach[†]. We found that the antipodal classifier was too conservative for our general manipulation tasks, frequently resulting in only a single grasp, or no grasps at all, over a small area. Since we are developing a human-in-the-loop system, we require an autonomous grasp calculator that provides more grasp options for the user to choose from. With that in mind, we designed a set of heuristics that result in approximately 5 to 20 potentially effective grasps for the objects used in our general manipulation tasks. We introduce three metrics:

- h_n minimizes the distance between the orientation of the grasp's approach angle and the orientation of the dominant plane's surface normal. Most manipulation features, such as handles and knobs, are situated such that perpendicular grasp poses are more effective.
- h_o represents whether the grasp's orientation is orthogonal to the principal direction (computed with Principal Component Analysis) of the local geometry around the grasp point. This comes from our observation that most man-made objects are designed to be grasped with poses orthogonal to their principal direction, i.e. poses that are "aligned" with the object.
- h_p minimizes the proximity of the grasp point to the algorithm's input point. Since the user specifies a grasp area by inputting an initial grasp point, the third heuristic prioritizes grasps that are closest to the user input.

Each heuristic is multiplied by a normalizing constant (n_n, n_o, and n_p) to place them on a [0,1] scale.

The full algorithm for leveraging the above grasp pose estimates and heuristics is shown in Algorithm 1. The grasp planner takes as input a point-of-interest and a point cloud. All calculations occur within a local workspace (in our case, a 10 cubic centimeter volume) situated around the point-of-interest. The algorithm first uses AGILE grasp to compute a set of hand hypotheses, which we then cluster by position and orientation into a set of unranked grasps (lines 2-3). Note that the hand hypotheses from the AGILE grasp pipeline are not classified by the AGILE classifier. Next, the algorithm performs a few calculations required to compute the heuristics, including cropping the point cloud to the workspace, segmenting the dominant plane, computing its surface normal, and separating the cropped point cloud into local geometry clusters based on Euclidean distance (lines 4-7). For each grasp, the algorithm computes a ranking based on our three heuristics (lines 10-14). The final ranking is determined from a linear combination of the three heuristics, which can be assigned different priority by adjusting the constants α and β (line 14). The ranked grasps are then returned in a priority queue where lower rankings are preferred. For our experiments, we used experimentally determined values of $\alpha = 0.6$ and $\beta = 0.25$, which prioritize perpendicular grasps most, and user input least.

Algorithm 1 Autonomous Grasp Planning and Ranking

Require: Point $pointOfInterest$, PointCloud $cloud$

1: Workspace w = createWorkspace($pointOfInterest$);
2: List<Pose> $hands$ = findGraspsAGILE($cloud$, w);
3: List<Pose> $grasps$ = cluster($hands$);
4: PointCloud pc = crop($cloud$, w);
5: Plane p = planeSegmentation(pc);
6: Orientation n = surfaceNormal(p)
7: List<PointCloud> $clusters$ = cluster(pc);
8: PriorityQueue<Pose> $rankedGrasps$;
9: **for all** g in $grasps$ **do**
10: PointCloud c = nearestCluster($clusters$, g);
11: h_n = n_n∗distance($g.orientation$, n);
12: h_o = n_o∗dstToOrthogonal($g.orientation$, PCA(c));
13: h_p = n_p∗distance($g.position$, $pointOfInterest$);
14: $graspRating = \alpha * h_n + \beta * h_o + (1 - (\alpha + \beta)) * h_p$
15: $rankedGrasps$.push(g, $graspRating$);
16: **end for**
17: **return** $rankedGrasps$;

[†]An open-source ROS package with our heuristic-based AGILE extension can be found here: `https://github.com/GT-RAIL/rail_grasp_calculation`

We leverage the above algorithm to create a point-and-click user interface. The user first inputs the point-of-interest by clicking on a point in a stream of camera data. Algorithm 1 then autonomously calculates a ranked list of grasp points within the local area around the clicked point. The ranked list is presented to the user, who then selects a final grasp to execute.

4. Interface Implementation

In this work, we focus specifically on browser-based interfaces for remote robot teleoperation. A browser-based interface allows naive end users to control the robot with a standard web browser, thus removing the need for them to install and configure additional software. We implemented each of the approaches described in Section 3 in browser-based interfaces for remote teleoperation.

Remote robot interaction interfaces are highly sensitive to bandwidth limitations and network latency. This is true in high-quality network environments, such as the one utilized in our work, and becomes even more critical in applications such as search and rescue, as shown in the DARPA Robotics Challenge (see Yanco et al. (2015) and Norton et al. (2017) for discussions of interaction under degraded communications in the DRC trials and finals, respectively), and space robotics, as characterized by Sheridan (1993). In this work, we compared two categories of interfaces characterized by different visualization modes. One can either stream the depth data and visualize it as a point cloud within a 3D depth view (similar to the main window of rviz[‡]), or stream only the rgb image visualized as a 2D image view. Table 1 summarizes the tradeoffs of the two visualization modes, which we discuss in detail below.

4.1. 3D Visualization

The standard method for 6-DOF teleoperation involves rendering the rgbd camera stream within an rviz-like 3D depth view, where all of the interaction occurs. This raises some difficulties for remote interfaces, the foremost of which is the bandwidth requirements for streaming point clouds. Even when sending a compressed point cloud[§], point cloud streaming with our setup (using ASUS Xtion 3D cameras) required 80 MB/s.

We implement our three approaches with 3D visualization as follows. We render the rgbd data as a point cloud within a 3D depth view. The user can set points-of-interest for CP and P&C by clicking on points in the point cloud. Ring-and-arrow markers for FP and CP, sphere markers for CP, and grasp pose visualizations for all three approaches are implemented as ROS interactive markers rendered within the 3D depth view. Examples of each approach are shown on the left in Figure 4.

The advantages of the 3D depth view are that 3D data is explicitly provided to the user, and the viewpoint, represented by a virtual camera in the rendered scene, is freely adjustable. These properties can increase situational awareness. On the other hand, the free camera results in a more complex interface, as the user is now responsible for controlling an additional 6 degrees of freedom (the camera's pan, tilt, zoom, and (x,y,z) position), potentially increasing workload.

4.2. 2D Visualization

An alternative strategy to the 3D depth view visualization is to use only the rgb camera feed in the interface, provided directly as a 2D image view. In this case the depth data does not need to be streamed, providing a significant reduction in bandwidth (26 MB/s on our setup, or 0.325 times the bandwidth required to stream the equivalent point cloud). While the client-side interface does not have access to depth data, the algorithms can still make use of it by indexing the depth channel with pixel coordinates.

The implementation of our approaches is similar to the 3D visualization mode, except that the user sets points-of-interest by clicking on image pixels rather than point cloud points, and the interactive markers are instead rendered as an overlay on the 2D image view. Examples of this are shown on the right in Figure 4.

Overall, the 2D image view provides a simpler interface for the user. This simplicity may negatively effect situational awareness, however, as the rgb camera feed does not provide any visualization of the depth data, and the viewpoint is fixed to that of the physical camera. The extent to which these factors affect user performance is unclear, and as such we included 2D and 3D visualization mode versions of each interaction approach in the evaluation study presented in Section 5.

5. Evaluation Study Design

We performed an evaluation study in order to compare the effectiveness of the three interaction approaches identified in Section 3, as well as the effect of implementing each approach with either a 2D or 3D visualization mode. The robot used consists of a 6-DOF Kinova JACO2 arm with a Robotiq-85 2-finger gripper and two Asus Xtion PRO LIVE 3D cameras mounted at complimentary angles to a workbench, providing a top and a side view of the workspace. The user interfaces are implemented using Robot Web Tools (Toris et al. 2015) for display in a web browser[¶].

5.1. Procedure

We recruited 90 participants (57 male, 32 female, 1 unreported) from a college campus to take part in a 2x3 factorial between-subjects study. Participants ranged in age from 20-39. Each participant received $10 as compensation. Participants were asked to rate their experience on a 1 (no experience) to 5 (very experienced) scale with robotics (2.5 ± 1.2), video games (3.3 ± 1.2), and 3D software (2.5 ± 1.3). Participants were then assigned to one of six conditions, with 15 participants per condition, balanced by gender and level of robotics experience.

Upon arriving, participants were brought to a computer out of sight and sound of the lab where the robot was operating. After completing the demographics and experience survey,

[‡] http://wiki.ros.org/rviz

[§] We stream point clouds using Robot Web Tools' depthcloud_encoder, which encodes the rgb image and depth image into a single image for streaming, to be reconstructed into a point cloud client-side

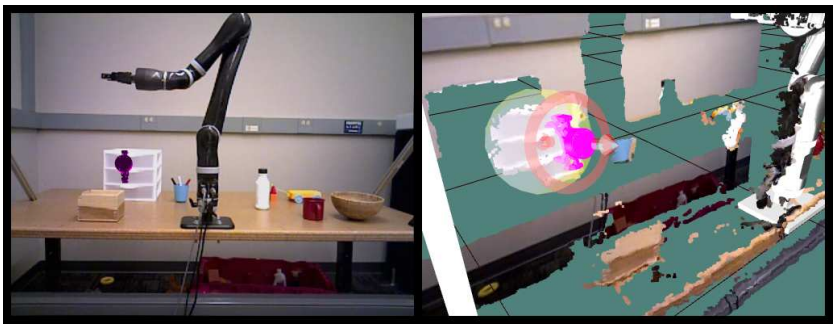
[¶] We have released the full set of browser-based interfaces used in the study here: https://github.com/GT-RAIL/puzzlebot_interface

Table 1. Comparison of visualization modes

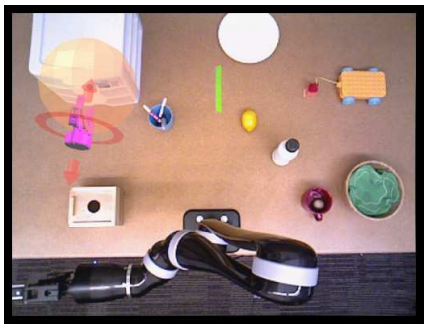
Visualization	**3D depth view**	**2D image view**
Data Properties		
data streamed	rgbd	rgb
latency (relative)	1.0	0.325
depth available to algorithms	✓	✓
User Interaction		
data visualized	rgbd	rgb
point-of-interest selection	click image pixel	click point cloud point
marker interaction	click and drag	click and drag
viewpoint	controllable	fixed

(a) Free Positioning, 3D visualization

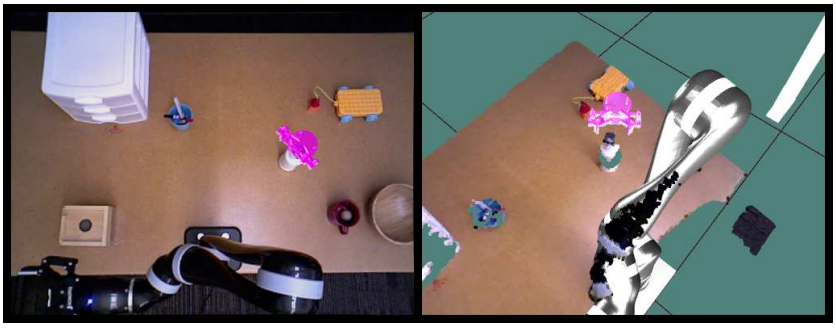
(b) Free Positioning, 2D visualization



(c) Constrained Positioning, 3D visualization



(d) Constrained Positioning, 2D vis.



(e) Point-and-Click, 3D visualization

(f) Point-and-Click, 2D visualization

Figure 4. Side-by-side comparison of the three interaction approaches implemented with both visualization modes. The 3D visualization mode (left column) includes a 2D image view next to a 3D depth view, which present the same data. All interaction happens in the 3D depth view; the 2D image view provides visualization only. In the 2D visualization mode (right column), all interaction happens on the image itself. We include the 2D image view in the 3D visualization mode because the rgb image data is already streamed with the point cloud, requiring no additional bandwidth

participants were given up to 10 minutes to complete two training tasks using the interface to which they were assigned. During training, they could ask the researchers questions about the interface and how to use it. Once training was complete and participants had no further questions, they were instructed to complete as many tasks as possible within a 30 minute time period. Upon completion of the tasks or at the end of the allotted time, participants completed a NASA

TLX (Hart and Staveland 1988) exit survey to evaluate workload while using their respective interfaces.

Each participant was given the same list of tasks to complete during the study. They could complete the tasks in any order, and skip or return to previously attempted tasks at any time. The tasks were as follows:

- *(Training) Move the lemon onto the white plate.* A simple pick-and-place task.
- *(Training) Reset the arm when you're finished.* An introduction to the reset primitive action, which moves the arm to a preset position that does not occlude the objects in either camera view. This also ensured the main part of the experiment would begin with the arm in a consistent starting pose.
- *Open the middle plastic drawer.* A task to grasp the correct handle in a set of stacked drawers, using it to open the drawer.
- *Slide open the wooden box.* Manipulation of an uncommon object that typically required some exploration.
- *Remove the black cap from the white bottle.* A precision task requiring rotation and translation.
- *Pour out the red mug into the wooden bowl.* A task requiring appropriately grasping the mug's handle, then completing the pouring action.
- *Remove a marker from the blue cup.* A cluttered-environment manipulation.
- *Pull the yellow cart onto the green line.* A task involving manipulating an out-of-reach object by pulling an attached string.

5.2. Conditions and Interfaces

The study had two independent variables–interaction approach, and visualization mode–resulting in the 2x3 table of conditions shown in Table 2. Each condition had an associated interface of the form shown in Figure 5. The interfaces differed in the visualization elements (c) and (d) for each condition, which were replaced with the appropriate 2D image view or 3D depth view depending on the visualization mode, and the appropriate set of markers for the interaction approach. The full set of elements for (c) and (d) are shown in Figure 4. The interaction approach-specific instructions and controls (e) also differed depending on the interaction approach.

The remaining elements were the same for every condition. The task list (a) gave users a list of tasks to complete for the evaluation study. Element (b) allows the user to cycle through the two cameras, determining which image is displayed in the 2D image view (c) and which point cloud is displayed in the 3D depth view (d). The robot provides text-based feedback in a feedback bar (f). This includes displaying the state of the current action being executed, the result (success or failure) of the previously executed action, and suggestions for improvement after a failed action execution. The arm controls element (g) provides the user with a set of common primitive actions.

The primitive actions included generally useful actions such as fully opening or closing the gripper and resetting the arm to a position that did not occlude the workspace from either camera view. The remaining actions were selected as the minimum set of primitive actions required to complete any task after performing a grasp. This included fixed-distance 10 centimeter translations in each Cartesian direction (with respect to a coordinate frame defined at the table and thus, aligned with the cameras) and 90 degree clockwise and counter-clockwise gripper rotation with respect to the JACO's wrist. We specifically chose these actions and fixed distances so that the tasks could be completed, thus evaluating the quality of the grasps for each task, while also making it virtually impossible to skip the initial grasp specification by using only the primitive actions to complete a task. In this way we could ensure the users would need to use the interface elements specific to their condition, which were the main focus of our evaluation.

5.3. Measures

Observed Measures. We identify a set of observable measures intended to measure effectiveness (*tasks completed*, *tasks failed*, *errors per task*) and efficiency (*time per completed task*, *interactions per task*). These measures are all calculated from values logged during the main portion of the study, after the training period concluded. Definitions of the measures are listed in Table 3.

We define a *completed task* as any task in which the goal was met at some point within the 30 minute time period, and an *attempted task* as any task the user devoted time towards completing. We define a *miss* as an attempted grasp that does not contact an object or the environment, and a *bad grasp* as any grasp that does make contact but cannot be used to complete the task. We define an *interaction* as any mouse button press that involves specifying or executing a grasp. This includes dragging a ring-and-arrow marker (FP and CP), setting or clearing a grasp point or approach angle (CP), calculating a new set of grasps or viewing a different calculated grasp (P&C), executing a grasp (all conditions), repositioning the virtual camera in the 3D viewer (all 3D conditions), and changing camera streams (all conditions).

Self-Reported Measures. Along with reporting their experience with robotics, video games, and 3D software at the beginning of the study, each participant completed a NASA TLX form at the end of the study. This allows us to measure workload for each of the interfaces and visualization modes, including *mental demand*, *physical demand*, *temporal demand*, *performance*, *effort*, and *frustration*.

6. Results

For each of our performance and efficiency measures described in Section 5.3, we conducted a two-way between subjects Analysis of Variance (ANOVA) to compare the main effects of interaction approach (FP, CP, P&C) and visualization mode (2D, 3D) and the interaction effect of the two independent variables on each measure. If a significant effect was found, we performed Tukey HSD post tests to determine which conditions significantly differed. We present the results for the observed effectiveness measures, followed by the observed efficiency measures, and end with the self-reported measures.

Table 2. 2x3 factorial study conditions

		Interaction Approach		
		Free Positioning	Constrained Positioning	Point-and-Click
Visualization Mode	2D image view	FP-2D	CP-2D	P&C-2D
	3D Depth View	FP-3D	CP-3D	P&C-3D

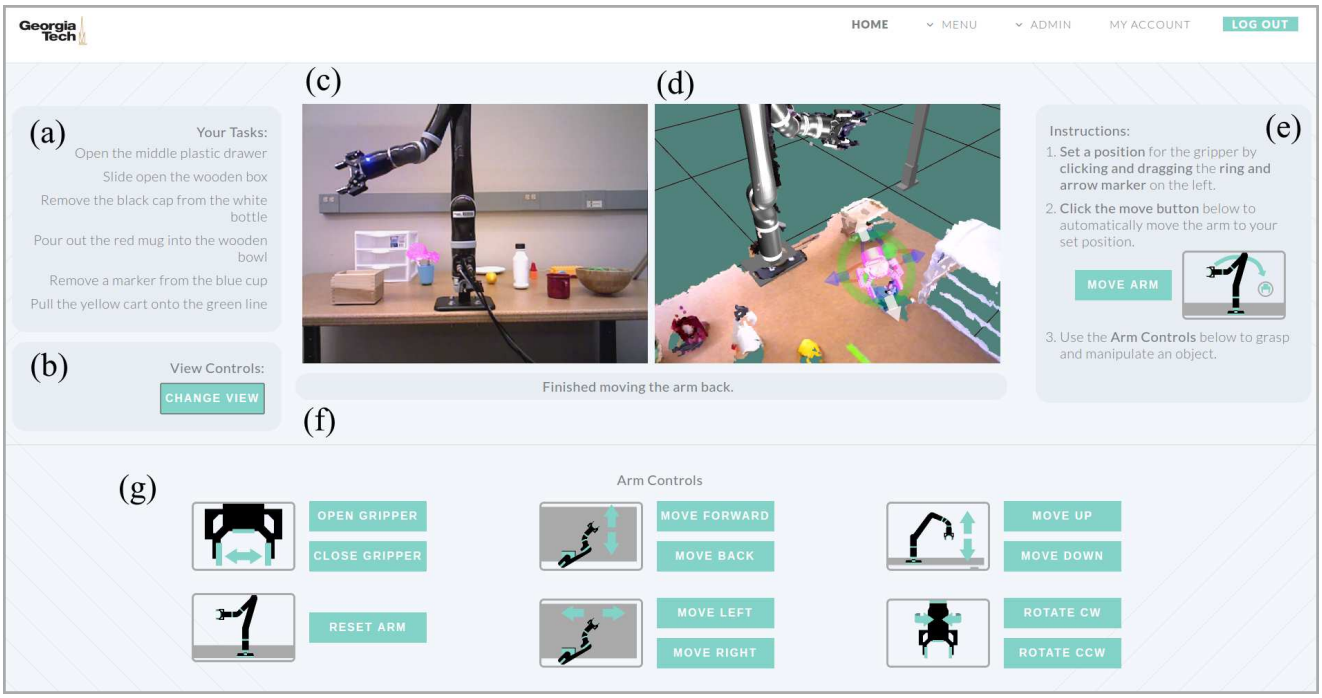


Figure 5. Complete web browser-based interface used to implement the interaction approaches. This version shows the Free Positioning approach with the 3D visualization mode

Table 3. Observed measures and descriptions

tasks completed	$number\ of\ completed\ tasks$
tasks failed	$attempted\ tasks - completed\ tasks$
time per completed task	$\frac{\sum elapsedTime(completed\ tasks)}{completed\ tasks}$
errors per task	$\frac{misses+bad\ grasps}{attempted\ tasks}$
interactions per task	$\frac{interactions}{attempted\ tasks}$

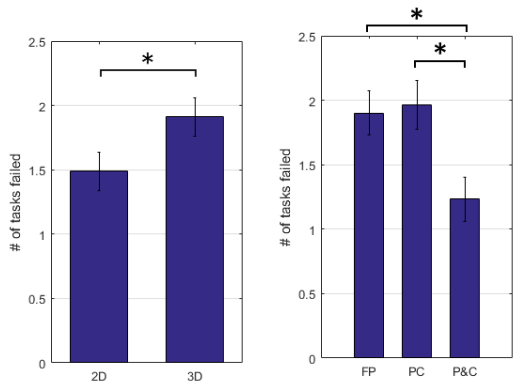
6.1. Observed Effectiveness Measures

Number of tasks completed. ANOVA showed significant main effects of both interaction approach ($F(2, 84) = 10.13, p = 0.0001$) and visualization mode ($F(1, 84) = 12.14, p = 0.0008$). This was qualified by an interaction effect between the two variables ($F(2, 84) = 4.07, p = 0.0206$), suggesting that the best interaction approach for task completion depends on which visualization method is used. We note that this is the only metric that showed an interaction effect. When the visualization mode is 2D, Point-and-Click results in significantly more tasks completed than Free Positioning ($p < 0.01$), with no other significant differences. When the visualization mode is 3D, Point-and-Click results in significantly more tasks completed compared to Constrained Positioning ($p < 0.05$), with no other significant differences. We present the table of means and standard deviations for inspection in Table 4.

Table 4. Means and standard deviations for number of tasks completed over all conditions

	FP	CP	P&C	Total
2D	2.6 ± 1.5	3.2 ± 1.4	4.5 ± 0.8	3.4 ± 1.5
3D	2.7 ± 1.7	1.5 ± 0.9	3.1 ± 1.3	2.5 ± 1.5
Total	2.7 ± 1.6	2.4 ± 1.4	3.8 ± 1.3	

Summary. Overall, we find that users are able to complete either a comparable number of tasks or more tasks when using the Point-and-Click approach as compared to the other interaction approaches. We examine additional measures to better determine the quality of their performance.



(a) Visualization mode **(b)** Interaction approach

Figure 6. Number of tasks failed (± 1 SE)

Number of Tasks Failed. ANOVA showed a significant main effect of interaction approach on task failure rate ($F(2, 84) = 5.36, p = 0.0064$). Point-and-Click had significantly fewer task failures than the other two interaction approaches ($p < 0.05$ for both), with no significant difference between Free Positioning and Constrained Positioning. We also found a significant main effect of visualization mode on task failure rate ($F(1, 84) = 4.36, p = 0.0398$), with 2D resulting in fewer task failures than 3D ($p < 0.05$). Both effects are shown in Figure 6.‖

Summary. After starting a task, users more frequently render the task incompletable (e.g. spilling the contents of the mug before reaching the bowl) or give up and switch to a new task when using Free Positioning or Constrained Positioning as opposed to Point-and-Click, and we see the same relationship when using 3D instead of 2D visualization regardless of interaction approach. Overall, with a 2D visualized Point-and-Click approach, users are more likely to successfully complete tasks they have started.

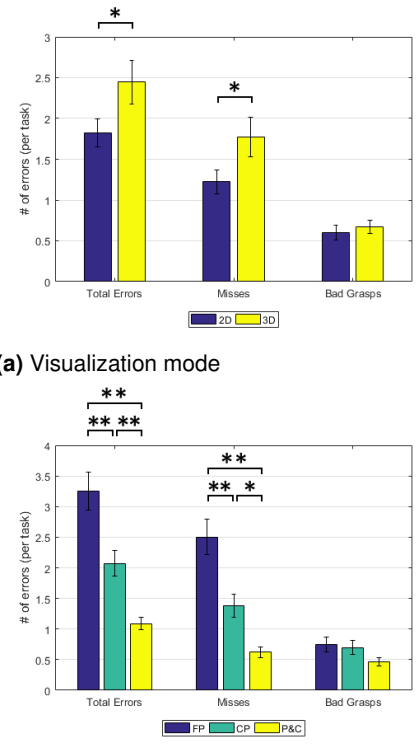
Number of Errors. Figure 7 shows a comparison of errors made, including a breakdown of errors into misses and bad grasps. Our analysis showed a significant main effect of interaction approach on number of errors ($F(2, 84) = 24.79, p < 0.001$). Constrained Positioning reduced the number of operator errors compared to Free Positioning ($p < 0.01$), with Point-and-Click further reducing errors compared to both ($p < 0.01$ in both cases). We also found a significant main effect of visualization mode on number of errors ($F(1, 84) = 6.1, p = 0.016$), with 2D resulting in fewer errors than 3D ($p < 0.05$). The main contribution to mistakes for every condition came from missed grasps, where the same significant relationships exist.

Summary. Interaction approaches that incorporate more scene information reduce the risk of increased operator error, as does using 2D visualization modes.

6.2. Observed Efficiency Measures

Plots for all observed efficiency measures are shown in Figure 8, with significant results reported below.

Time per completed task. As this measure is only for completed tasks, we only include data from participants who completed at least one task. 1 user in the FP-2D



(a) Visualization mode

(b) Interaction approach

Figure 7. Breakdown of mean errors (± 1 SE)

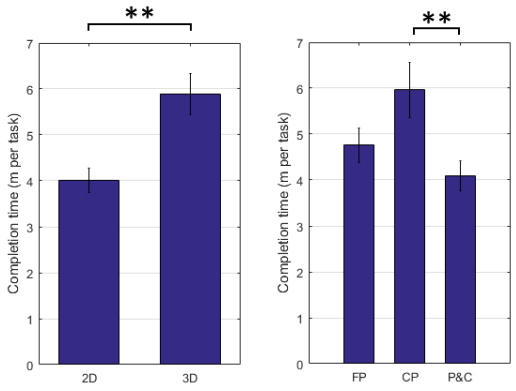
condition, 2 users in the FP-3D condition, and 1 user in the CP-3D condition completed zero tasks, and were thus excluded from this analysis. ANOVA showed a significant main effect of interaction approach on time per completed task ($F(2, 80) = 5.24, p = 0.0073$). Users with Point-and-Click completed tasks significantly faster than users with Constrained Positioning ($p < 0.01$), with no other significant differences. ANOVA also showed a significant main effect of visualization mode on time per completed task ($F(1, 80) = 15.17, p = 0.0002$), with 2D resulting in faster completion times than 3D ($p < 0.01$).

Summary. Regardless of interaction approach, using 2D visualization modes significantly reduces task completion time.

Interactions per task. ANOVA showed a significant main effect of interaction approach on interactions per task ($F(2, 81) = 7.78, p = 0.0008$). Point-and-Click resulted in significantly fewer actions than the other two interaction approaches ($p < 0.01$ for both), with no significant difference between Free Positioning and Constrained Positioning. We also found a significant main effect of visualization mode on interactions per task ($F(1, 81) = 11.51, p = 0.0011$), with 2D visualization resulting in significantly fewer interactions than 3D visualization ($p < 0.01$).

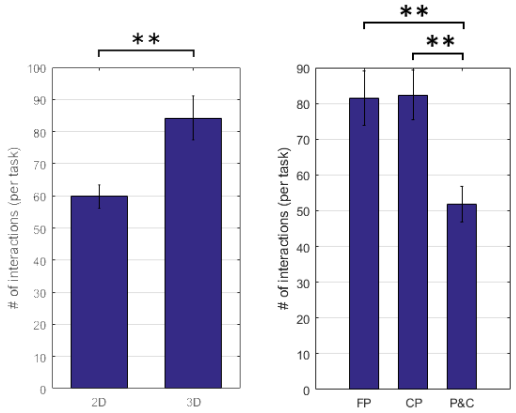
Summary. Incorporating surface geometry into the pose specification process significantly reduces the total number of interactions, as does replacing the 3D depth view with a 2D image view.

‖ For all of our plots, we denote statistically significant results with $* = p < 0.05$, $** = p < 0.01$.



(a) Time per completed task by visualization mode

(b) Time per completed task by interaction approach



(c) Interactions per task by visualization mode

(d) Interactions per task by interaction approach

Figure 8. Efficiency measures (± 1 SE)

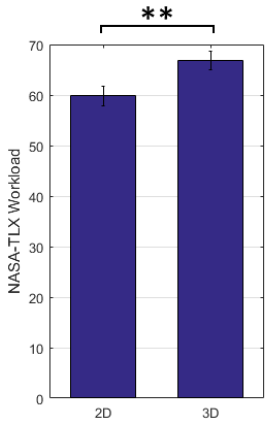


Figure 9. Mean workload (± 1 SE)

6.3. Self-Reported Measures

ANOVA showed a significant main effect of visualization mode on workload ($F(1, 84) = 7.04, p = 0.0095$), as measured by the NASA TLX exit survey. As shown in Figure 9, 2D visualizations significantly reduced workload as compared to 3D visualizations ($p < 0.01$). We found no significant main effect of interaction approach on workload. We present the full breakdown of NASA TLX results in Table 5.

We also examined participants' self-reported experience with robotics, video games, and 3D software to see if experience in any of these areas correlated with any of our performance measures. We found no significant correlation between experience in each area and any of our performance metrics, for any interaction approach or for either visualization mode. While no participants were experienced users of our interfaces, it's interesting to note that past experience in seemingly related areas also did not significantly improve users' performance in 6-DOF manipulator teleoperation.

7. Discussion and Future Work

With regards to interaction approach, our results show that the Point-and-Click approach significantly improves users' performance in remote robot teleoperation object manipulation tasks. While Constrained Positioning had some advantages over Free Positioning (mainly in reducing the rate of grasping errors), Point-and-Click has an even greater reduction in error rate, as well as significant reduction in task failures and similar or better performance in tasks completed and completion time, all with significantly fewer interactions than the other two approaches.

The cause of the larger task failure rates in Free Positioning and Constrained Positioning come from the potentially large space of unsuitable gripper poses reachable with the two approaches. The full 6-DOF control of Free Positioning, and to a lesser extent the 4-DOF control of Constrained Positioning, create such a large workspace of possible grasp poses that movements not carefully planned will frequently result in failed grasps. Less successful users in Free Positioning and Constrained Positioning conditions spent the majority of their time attempting a single task, moving the gripper around the object without successfully grasping it. With the Point-and-Click approach, however, simply clicking on the object to be manipulated often resulted in a successful grasp with no further interaction, thus increasing the baseline performance of non-expert users. Non-expert user performance is an important concern when designing interfaces meant to be deployed to the wider population. We leave identification of other factors affecting non-expert user performance, and how to address them, to future work.

One of the most significant performance improvements from our new interaction approaches was a reduction in errors per task. The most frequent type of error was missing a grasp, in which the gripper failed to make contact with any part of the environment, object or otherwise. Both Constrained Positioning and Point-and-Click require the user to focus their end-effector positioning around a point cloud by clicking on a point to initiate the interaction. Constraining the interaction to a physical surface significantly reduces the number of missed grasps, resulting in more efficient use of the arm.

Dependency on depth does create limitations for the new interaction approaches, however. Point-and-Click requires an accurate point cloud. For difficult to detect objects, such as highly specular objects, it will not be able to calculate a good set of grasps. For objects undetectable by depth cameras, such as transparent objects, or occluded objects in

Table 5. NASA TLX Results (means and standard deviations)

	FP-2D	*FP-3D*	*CP-2D*	*CP-3D*	*P&C-2D*	*P&C-3D*
total workload	57.9 ± 13.6	64.5 ± 10.9	64.1 ± 14.1	70.2 ± 14.1	57.5 ± 10.6	65.8 ± 11.2
mental demand	13.2 ± 9.6	12.9 ± 8.4	11.9 ± 9.1	11.7 ± 6.1	12.1 ± 8.4	14.5 ± 10.2
phys. demand	1.1 ± 3.6	4.5 ± 6.6	2.5 ± 5.1	4.3 ± 6.5	0.9 ± 2.7	0.9 ± 2.8
temp. demand	10.4 ± 9.9	9.1 ± 7.7	9.2 ± 7.4	8.3 ± 6.5	8.2 ± 6.9	9.9 ± 7.8
performance	9.2 ± 6.4	10.5 ± 6.9	11.8 ± 8.5	18.4 ± 8.2	13.1 ± 8.5	12.3 ± 8.0
effort	12.4 ± 6.2	13.3 ± 5.4	12.6 ± 5.3	13.6 ± 6.8	11.9 ± 4.7	10.8 ± 5.7
frustration	11.5 ± 10.0	14.2 ± 10.6	16.0 ± 11.4	13.8 ± 10.2	11.2 ± 10.5	17.4 ± 7.5

cluttered scenes, neither Constrained Positioning nor Point-and-Click will work, since the initial point of interest cannot be selected. An ideal interface would include an option to switch to Free Positioning for redundancy in these cases.

A final interesting point with regards to interaction approaches comes from the NASA TLX data. While there are clear advantages to incorporating more depth data and autonomous calculation in manipulation interfaces (including a significant reduction in number of interactions), users do not appear to feel any reduction in workload. To further explore this, we suggest conducting a within-subjects evaluation study so that more subjective measures about preferred interaction methods can be collected.

The visualization mode of the interfaces consistently had a significant effect on user performance, regardless of the interaction approach. Users given the 2D visualization mode were able to complete more tasks with fewer task failures, making fewer errors per task, and completing tasks more quickly than users given the 3D visualization mode. This was all accomplished with significantly fewer interactions, and users reported a significantly lower workload. This result holds even for approaches such as Free Positioning, which was initially designed for 3D visualization modes. This is consistent with the literature that suggests reducing the number of transformations required by the user can improve teleoperation performance. By eliminating the controllable viewpoint of the 3D depth view, we remove 6 degrees of rotation and translation that the user is responsible for performing.

Another advantage of the 2D visualization mode comes from the initial click required to specify a point-of-interest for Constrained Positioning and Point-and-Click. While this appears to be a simple action, clicking on a point cloud point within the 3D depth view proved to be more difficult. We identify a few reasons in explanation. When clicking in a 3D depth view, much of the window is occupied by free space not contained within the sensor's field of view, creating a smaller target for the user. Points in the 3D scene are rendered with different sizes proportional to their distance from the virtual camera, making specific key points (e.g. the handle of the mug, or the pens in the cup) difficult to click without first setting a camera view that puts those points in the foreground of the scene. Further, self-occlusion of the point cloud is common depending on the viewpoint, resulting in multiple potential points-of-interest occupying the same screen pixels. Conversely, in the 2D image view, every pixel is a valid location to click when setting a point-of-interest and every image pixel is shown without occlusion, preventing miss clicks that cause increased pose specification time and potential grasping errors from inaccurate initialization of CP and P&C.

We found no support for our concern that removing the depth data visualization would reduce situational awareness, as we found significantly better performance with the 2D visualization mode for every interaction approach. When considering the advantages of reduced bandwidth requirements, along with the improvements in the measures we have considered, we recommend using 2D image views over 3D depth views wherever possible for remote robot teleoperation manipulation interfaces. Our one caveat is that the user should be able to switch between multiple camera views to get a better sense of the depth of objects in the scene. In our work, we accomplished this by outfitting the environment with two cameras fixed at complimentary angles. For a single robot with an onboard camera, one could achieve a similar situation by moving the robot to provide additional viewpoints before grasp execution. We leave the realization of the single-robot/single-camera case in a supervisory remote manipulation interface with 2D visualization to future work.

8. Conclusion

This work introduces alternative 6-DOF end-effector positioning approaches to the widely used ring-and-arrow marker approach for remote object manipulation, designed specifically for interactions over 2D camera image streams. The approaches are realized by incorporating depth data into the grasp specification algorithms to reduce the number of 3D transformations the user is responsible for performing. The results are a Constrained Positioning approach which reduces pose specification to a 4-DOF process performed in 3 steps, and a Point-and-Click approach which forgoes control of 3D transformations entirely by leveraging autonomous grasp planning, simplifying pose specification to a single click. Both approaches can be implemented with either a 3D visualization mode (the standard for the 6-DOF ring-and-arrow marker) or a 2D visualization mode (the inspiration for the new approaches).

The results of our evaluation study show that the Point-and-Click approach, which uses surface geometry from depth data to autonomously calculate grasp suggestions, significantly outperforms the Constrained Positioning and Free Positioning approaches in either visualization mode. The Point-and-Click approach translates to both a more effective user interface, resulting in a significantly greater number of tasks completed with significantly fewer failed tasks, and a more efficient user interface, resulting in significantly fewer

grasping errors and requiring significantly fewer interactions than the other two approaches. Additionally, providing users with only a 2D image view rather than a 3D depth view significantly improved performance across all of our measures, while reducing bandwidth requirements, regardless of the interaction approach used. By removing the user-facing presentation of depth data and the need for 3D interfaces that can overcomplicate interactions, while incorporating depth data directly into grasp pose specification algorithms, we can significantly improve remote robot teleoperation interfaces for complex tasks.

Acknowledgment

This work was supported in part by NSF award number IIS 13-17775, ONR award number N000141410795, and NASA award number NNX16AR59G.

References

Cancedda L, Cannavò A, Garofalo G, Lamberti F, Montuschi P and Paravati G (2017) Mixed reality-based user interaction feedback for a hand-controlled interface targeted to robot teleoperation. In: *International Conference on Augmented Reality, Virtual Reality and Computer Graphics*. Springer, pp. 447–463.

Chen TL, Ciocarlie M, Cousins S, Grice PM, Hawkins K, Hsiao K, Kemp CC, King CH, Lazewatsky DA, Nguyen H, Paepcke A, Pantofaru C, Smart WD and Takayama L (2013) Robots for humanity: A case study in assistive mobile manipulation .

Chitta S (2016) Moveit!: An introduction. In: *Robot Operating System (ROS)*. Springer, pp. 3–27.

Ciocarlie M, Hsiao K, Leeper A and Gossow D (2012) Mobile manipulation through an assistive home robot. In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 5313–5320. DOI:10.1109/IROS.2012.6385907.

DeDonato M, Dimitrov V, Du R, Giovacchini R, Knoedler K, Long X, Polido F, Gennert MA, Padır T, Feng S, Moriguchi H, Whitman E, Xinjilefu X and Atkeson CG (2015) Human-in-the-loop control of a humanoid robot for disaster response: A report from the darpa robotics challenge trials. *Journal of Field Robotics* 32(2): 275–292.

DeJong BP, Colgate JE and Peshkin MA (2004) Improving teleoperation: reducing mental rotations and translations. In: *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 4. IEEE, pp. 3708–3714.

Fischinger D, Weiss A and Vincze M (2015) Learning grasps with topographic features. *The International Journal of Robotics Research* 34(9): 1167–1194.

Gossow D, Leeper A, Hershberger D and Ciocarlie M (2011) Interactive markers: 3-d user interfaces for ros applications [ros topics]. *IEEE Robotics & Automation Magazine* 18(4): 14–15.

Hachet M, Decle F, Knodel S and Guitton P (2008) Navidget for easy 3d camera positioning from 2d inputs. In: *Proceedings of the 2008 IEEE Symposium on 3D User Interfaces*, 3DUI '08. Washington, DC, USA: IEEE Computer Society. ISBN 978-1-4244-2047-6, pp. 83–89. DOI:10.1109/3DUI.2008.4476596. URL http://dx.doi.org/10.1109/3DUI.2008.4476596.

Hart SG and Staveland LE (1988) Development of nasa-tlx (task load index): Results of empirical and theoretical research. *Advances in psychology* 52: 139–183.

Henrysson A and Billinghurst M (2007) Using a mobile phone for 6 dof mesh editing. In: *Proceedings of the 8th ACM SIGCHI New Zealand chapter's international conference on Computer-human interaction: design centered HCI*. ACM, pp. 9–16.

Hsiao K, Chitta S, Ciocarlie M and Jones EG (2010) Contact-reactive grasping of objects with partial shape information. In: *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, pp. 1228–1235.

Jiang Y, Moseson S and Saxena A (2011) Efficient grasping from rgbd images: Learning using a new rectangle representation. In: *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. pp. 3304–3311.

Katzakis N, Hori M, Kiyokawa K and Takemura H (2011) Smartphone game controller. In: *Proceedings of the 74th HIS SigVR Workshop*. Citeseer.

Kemp CC and Grice PM (2016) Assistive mobile manipulation: Designing for operators with motor impairments. In: *RSS 2016 Workshop on Socially and Physically Assistive Robotics for Humanity*.

Kent D, Behrooz M and Chernova S (2016) Construction of a 3d object recognition and manipulation database from grasp demonstrations. *Autonomous Robots* 40(1): 175–192. DOI: 10.1007/s10514-015-9451-2. URL http://dx.doi.org/10.1007/s10514-015-9451-2.

Kohlbrecher S, Romay A, Stumpf A, Gupta A, Von Stryk O, Bacim F, Bowman DA, Goins A, Balasubramanian R and Conner DC (2015) Human-robot teaming for rescue missions: Team vigir's approach to the 2013 darpa robotics challenge trials. *Journal of Field Robotics* 32(3): 352–377.

Leeper A, Hsiao K, Ciocarlie M, Takayama L and Gossow D (2012) Strategies for human-in-the-loop robotic grasping. In: *Human-Robot Interaction (HRI), 2012 7th ACM/IEEE International Conference on*. pp. 1–8. DOI:10.1145/2157689.2157691.

Lenz I, Lee H and Saxena A (2015) Deep learning for detecting robotic grasps. *The International Journal of Robotics Research* 34(4-5): 705–724.

Levine S, Pastor P, Krizhevsky A and Quillen D (2016) Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *arXiv preprint arXiv:1603.02199* .

Lipton JI, Fay AJ and Rus D (2018) Baxter's homunculus: Virtual reality spaces for teleoperation in manufacturing. *IEEE Robotics and Automation Letters* 3(1): 179–186.

Mahler J, Liang J, Niyaz S, Laskey M, Doan R, Liu X, Ojea JA and Goldberg K (2017) Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. In: *Robotics: Science and Systems (RSS)*.

Masliah MR and Milgram P (2000) Measuring the allocation of control in a 6 degree-of-freedom docking experiment. In: *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. ACM, pp. 25–32.

Norton A, Ober W, Baraniecki L, McCann E, Scholtz J, Shane D, Skinner A, Watson R and Yanco H (2017) Analysis of human–robot interaction at the darpa robotics challenge finals. *The International Journal of Robotics Research* : 0278364916688254.

Parsons LM (1995) Inability to reason about an object's orientation using an axis and angle of rotation. *Journal of experimental psychology: Human perception and performance* 21(6): 1259.

Peppoloni L, Brizzi F, Ruffaldi E and Avizzano CA (2015) Augmented reality-aided tele-presence system for robot manipulation in industrial manufacturing. In: *Proceedings of the 21st ACM Symposium on Virtual Reality Software and Technology*. ACM, pp. 237–240.

Phillips-Grafflin C, Alunni N, Suay HB, Mainprice J, Lofaro D, Berenson D, Chernova S, Lindeman RW and Oh P (2014) Toward a user-guided manipulation framework for high-dof robots with limited communication. *Intelligent Service Robotics* 7(3): 121–131.

Pinto L and Gupta A (2016) Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In: *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. pp. 3406–3413.

Quigley M, Conley K, Gerkey B, Faust J, Foote T, Leibs J, Wheeler R and Ng AY (2009) Ros: an open-source robot operating system. In: *ICRA workshop on open source software*, volume 3. Kobe, Japan, p. 5.

Rakita D, Mutlu B and Gleicher M (2017) A motion retargeting method for effective mimicry-based teleoperation of robot arms. In: *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*. ACM, pp. 361–370.

Saxena A, Wong LL and Ng AY (2008) Learning grasp strategies with partial shape information. In: *AAAI*, volume 3. pp. 1491–1494.

Sheridan TB (1993) Space teleoperation through time delay: review and prognosis. *IEEE Transactions on robotics and Automation* 9(5): 592–606.

ten Pas A and Platt R (2015) Using geometry to detect grasp poses in 3d point clouds. In: *Intl Symp. on Robotics Research*.

ten Pas A and Platt R (2016) Localizing handle-like grasp affordances in 3d point clouds. In: *Experimental Robotics*. Springer, pp. 623–638.

Toris R, Kammerl J, Lu DV, Lee J, Jenkins OC, Osentoski S, Wills M and Chernova S (2015) Robot web tools: Efficient messaging for cloud robotics. In: *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, pp. 4530–4537.

Yanco HA, Norton A, Ober W, Shane D, Skinner A and Vice J (2015) Analysis of human-robot interaction at the darpa robotics challenge trials. *Journal of Field Robotics* 32(3): 420–444.

Zhai S and Milgram P (1998) Quantifying coordination in multiple dof movement and its application to evaluating 6 dof input devices. In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM Press/Addison-Wesley Publishing Co., pp. 320–327.

Zucker M, Joo S, Grey MX, Rasmussen C, Huang E, Stilman M and Bobick A (2015) A general-purpose system for teleoperation of the drc-hubo humanoid robot. *Journal of Field Robotics* 32(3): 336–351.