Assisted Teleoperation Strategies for Aggressively Controlling a Robot Arm with 2D Input

Erkang You and Kris Hauser
School of Informatics and Computing, Indiana University at Bloomington
{erkyou,hauserk}@indiana.edu

***Abstract*—This paper studies assisted teleoperation techniques for controlling a 6DOF robot arm using click-and-drag input from a computer mouse. Experiments were conducted to investigate how task performance and user preferences are affected by low-level motion control strategies, which must deal with collision avoidance, dynamics constraints, and erroneous input. Five strategies were implemented and compared. As baseline strategies we study direct joint control and Cartesian positioning via inverse kinematics. We also implemented three obstacle avoidance strategies, including a predictive safety filter, a reactive potential field, and a real-time sample-based motion planner. Blind experimental trials assigned 22 novice subjects to five subgroups corresponding to each strategy and asked them to control the arm in simulation on a variety of reaching tasks in cluttered environments. Unsurprisingly, the obstacle avoidance strategies achieve major safety improvements, although subjects felt noticeably less in control of the robot than those using the baseline methods. The motion planning strategy shows the most promise; it completed tasks twice as fast as any other method and received high ratings for perceived safety, cooperativeness, and overall satisfaction.**

I. Introduction

Assisted teleoperation allows a user to operate a robot while offloading some of the work of motion control, which allows the user to concentrate on more important high-level tasks. Such systems may be more convenient, safe, and easier to use than direct teleoperation, and may enable novice users to control robot arms successfully in applications such as material handling, industrial assembly, household robots, remotely operated vehicles, and security applications (e.g., bomb disposal). Assisted teleoperation is also needed to enable robot control through nontraditional input devices, like brain-machine interfaces and voice control, which can be used by impaired users. But they inherently cause some loss of control for the user, and there are major knowledge gaps about whether users are willing to accept losses of control for increased convenience or safety — and if so, which control strategies provide greatest gain with least frustration.

This paper is an attempt to address some of these gaps in the context of systems that offload low-level task achievement and collision avoidance onto the robot. We consider a user operating a 6DOF robot arm at high speed in a cluttered known environment, using point-and-click input on a 2D display via a computer mouse — essentially "click-dragging" points on the robot to their desired positions. Assisted teleoperation strategies are used to process these raw commands in real-time in order to avoid unsafe motion and to deal with the significant dynamics of the robot. Although the user and robot have a clear master-slave relationship, the user has an active cognitive role in reacting, learning, and adapting his/her input to the robot. We wish to identify analytical properties of teleoperation strategies (e.g., collision avoidance, response times, and controllability properties) that correspond to empirical performance characteristics (e.g., target reaching speed, safety, learning curves, and user perceptions) of the human-robot system.

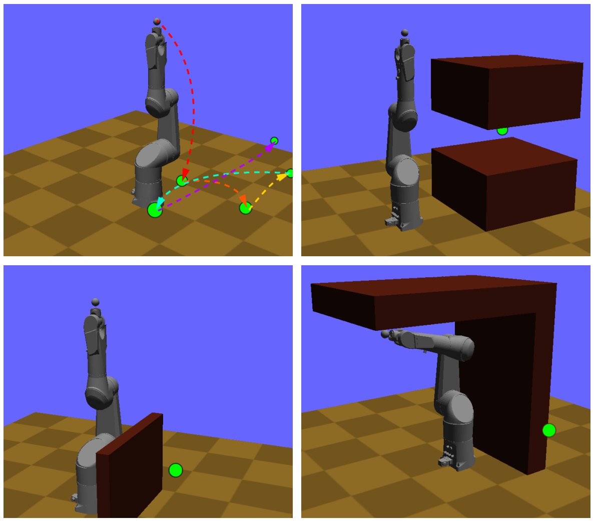

Fig. 1. The four scenarios in our user trials. Users are asked to touch the targets (green) while avoiding obstacles. (This figure is best viewed in color)

Our experiments compare three types of collision avoidance techniques: inverse kinematics with a predictive safety filter (IK+S); reactive potential fields (PF); and a real-time sample-based motion planner (MP) based on the work of Frazzoli et al. [9]. We also tested direct joint control (JC) and basic inverse kinematics (IK), which do not perform collision avoidance. Each technique allows the robot to maneuver aggressively across the workspace by handling dynamic constraints. We were particularly interested in the MP technique; it uses a randomized motion planner that is probabilistically complete and hence, theoretically allows the robot to access all feasible configurations in its configuration space. But, it is nondeterministic, less responsive, and leads to the least predictable motions of all techniques.

We performed user trials with 22 novice operators, divided into five groups each assigned an assisted teleoperation technique. Each user attempted four different task scenarios in a simulation environment, in which the user is asked to reach targets in 3D space (Figure 1). Our results demonstrate that automating collision avoidance achieves statistically significant improvements in robot safety, but causes users to feel less in control of the robot. Surprisingly, the sample-based motion planner performed well overall; it is safe, solved tasks over 50% faster than any other technique, and received either the highest or second-highest user ratings in the categories of safety, cooperativeness, and overall satisfaction. These results suggest that users are largely receptive to losses of control as long as the system demonstrates large enough gains in safety, completeness, and convenience. We suspect the motion planner "abstracts out" the complexity of the configuration space, allowing users to focus more on the positioning task rather than constructing feasible paths.

II. Related Work

Teleoperated robot arms are widely used in space robotics, bomb disposal, remotely operated vehicles, and robotic surgery, but are usually direct-controlled by the human operator. Direct joint control is widely used but slow, tedious, and unintuitive. Haptic input devices, such as those used in robot surgery, are more intuitive, but are expensive, highly specialized, and have limited workspaces. In both cases, the user is completely responsible for guiding the robot's motion, which requires extensive training and constant attention. in order to achieve low-level motions and to avoid collisions.

Nontraditional input devices such as voice [12] and gesture [19] are of interest for novice robot control as well as in assistive robotics for those who have lost function or fine motor skills. Another technology under consideration is brain-computer interfaces (BCIs). Invasive direct neural input has been used for real-time robot control, starting from work that enabled rats to control a 1D robot lever [4] and leading to monkeys self-feeding with a 5DOF robot arm [23]. Noninvasive BCIs, e.g., electroencephelography (EEG), have also been used for robot control [2], but exhibit extremely noisy signals that provide no more than a few bits of reliable information per second. The major challenge in using any nontraditional input device is that input signals are low-bandwidth, noisy, and often systematically erroneous, and are therefore poor for direct robot control. Hence, assisted teleoperation is needed.

Assisted teleoperation can provide multiple types of assistance to a human user, such as collision avoidance, tremor filtering, targeting precision, path tracking, orientation control, dynamics compensation, and trajectory smoothing, and can provide this assistance in multiple ways, such as autonomous low-level control, information displays, and haptic feedback. Several models of assistance have been proposed:

- *Supervisory control* assigns the user a managerial role over the robot's largely autonomous behavior [7].
- *Collaborative control* treats the user and robot as peers that must resolve conflicts using negotiation and dialogue [8].
- *Shared control* gives the robot control of some known dimensions of a task to be handled at a fast update rate [3, 10]. Virtual fixtures are a common shared control technique for manipulator arms that uses haptic feedback to help users control end effectors along specified paths, surfaces, or orientations [18].
- *Adjustable autonomy* addresses the issue of choosing autonomy levels appropriate to preference, trust, skill level, or the demands of a given situation [6, 15].

Most strategies considered in the current work fall within the shared control model and respond to erroneous commands by halting or resisting motion. The exception is the MP technique, which can be viewed as a non-verbal collaborative control scheme in which the robot has clear operational constraints and independent reasoning processes that may cause it to disobey the user's direct commands in a nontrivial way.

As proposed by adjustable autonomy, it would be wise to allow experienced users to choose automation strategies as appropriate for the task at hand. For clarity of interpretation, the experiments in this paper do not permit users to switch between strategies; doing so requires more user training and requires more complex safeguards against experimental bias due to ordering effects (users' choices may be affected by subtle presentation factors such as order and naming).

III. Problem Definition

This paper considers user interfaces for commanding a robot arm to rapidly reach targets in a cluttered three-dimensional environment using a "point-and-click" interface from a 2D input device (e.g., a computer mouse). The user views a 2D display of the robot, and can translate, rotate, and scale the view freely. To help judge depth we include a widget that indicates the distance from the end effector to the goal position (Figure 2). We provide visual collision cues in the form of brightly-colored contact force vectors.

The user's basic mode of input is a *click-and-drag* operation in which the user clicks a point on the robot and then drags the cursor to indicate a desired translation. For simplicity we do not consider orientation control. Some processing of these commands is necessary because direct translation could cause the robot to perform unsafe movements; the cursor can travel and accelerate faster than the robot, or could guide the robot into collision.

The robot platform is a Staubli TX90L 6DOF industrial robot. Its state includes both configuration q and velocity $\dot{q}$. The control algorithms in this paper are given a CAD model of the robot, its joint limits $q_{min} \leq q \leq q_{max}$, velocity bounds $|\dot{q}| \leq \dot{q}_{max}$, and conservative acceleration bounds $|\ddot{q}| \leq \ddot{q}_{max}$. The robot can reach fairly high speeds; the velocity limits are $360°$/s and higher. The acceleration constraints are expressed as box-bounds primarily for computational convenience, rather than because of an inherent limitation of our approach. This formulation allows us to quickly compute dynamically-feasible interpolating curves between states, using the method of [11]. The values of $\ddot{q}_{max}$ are chosen to be a conservative approximation of the torque and power bounds that apply to

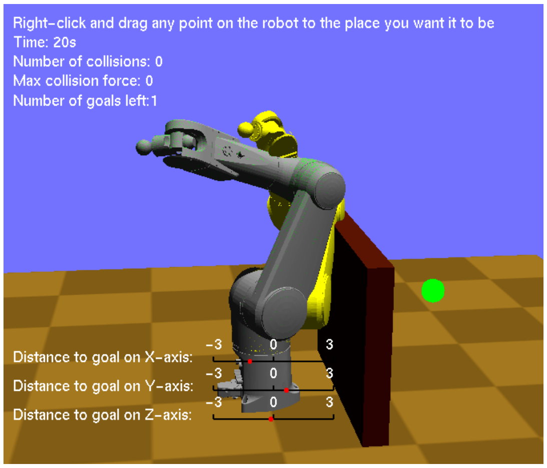
Right-click and drag any point on the robot to the place you want it to be
Time: 20s
Number of collisions: 0
Max collision force: 0
Number of goals left:1
Distance to goal on X-axis:
Distance to goal on Y-axis:
Distance to goal on Z-axis:

Fig. 2. Snapshot of the GUI. The spherical target (green) lies behind the obstacle. Both the current (gray) and desired (yellow) configuration are displayed to the user. (This figure is best viewed in color)

the actual robot arms. Obstacles in the environment are given as a rigid CAD model.

All user studies are performed on a dynamic simulation based on Open Dynamics Engine (ODE); experiments on the physical robot in our lab are planned in the near future. To approximate real-world conditions our experiments contain several sources of uncertainty. Obstacle measurement uncertainty is introduced by growing the simulation geometry by 2 mm from the planner's model. Also, the robot's controller produces tracking errors that are usually less than 0.5 mm but can approach 4 mm at high speeds. Finally, numerical errors in ODE's integration scheme can approach 0.01 mm.

IV. Assisted Teleoperation Strategies

To process click-and-drag input we implemented five assisted teleoperation techniques that represent a range of representative strategies in the literature:

1) Direct joint control (JC)
2) Inverse kinematics (IK)
3) Inverse kinematics with predictive safety filter (IK+S)
4) Reactive potential field (PF)
5) Sample-based motion planner (MP)

To summarize, the JC technique differs from the rest of the in that it does not use Cartesian positioning. The latter three strategies use different collision avoidance techniques: IK+S disallows infeasible commands; PF pushes the arm away from obstacles and joint limits; MP uses a configuration space planning approach that explores the space of collision-free motions. These differences have various implications as discussed in Section IV-F.

A. Direct Joint Control

In the basic direct joint control scheme (JC) the user controls one joint at a time by a click-drag operation. The vertical displacement of the mouse is linearly translated into the joint displacement of the desired configuration q_d. q_d is clamped to joint limits and is drawn transparently on the display for visual feedback. A velocity- and acceleration-bounded interpolation curve is constructed between the current configuration and current velocity and q_d (with zero velocity). This approach is taken because q_d can change faster than the robot can move. Figure 3 illustrates this method.

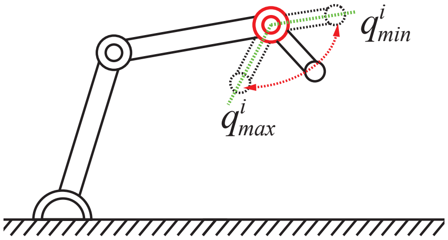

Fig. 3. The joint control strategy (JC) translates vertical mouse motion into the desired angle of the selected joint (highlighted in red). Angles are clamped within joint limits.

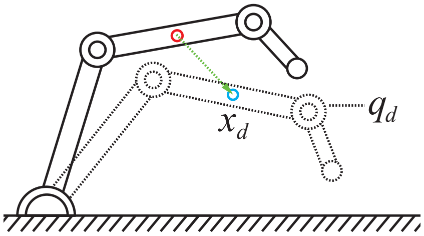

Fig. 4. The inverse kinematics strategy (IK) translates a click-and-drag operation into a Cartesian motion of the clicked point (highlighted in red).

B. Inverse Kinematics

Inverse kinematics (IK) is a classic technique in both robotics and animation for controlling articulated robots and characters in Cartesian space [22]. In the IK scheme, the user drags the point on the robot parallel to the image plane to a desired 3-D world space position (Figure 4). Prior work has used a variety of techniques, e.g., resolved rate control, in order to generate smooth robot trajectories from smooth Cartesian motion [5, 17]. We use a slightly different approach to deal with jerky mouse motions.

Define $x^0_c \in R^3$ be the point originally clicked on the robot and let $x_c(q) \in R^3$ be its new position viewed as being attached to the robot as it moves to configuration q. Let $x^t_d \in R^3$ be the desired position of the clicked point at time t obtained by displacing x^0_c by the mouse motion vector parallel to the image plane. The IK technique uses a numerical Newton-Raphson inverse kinematics solver (based on the Jacobian pseudoinverse) to find a desired configuration q_d that solves $\min_{q_d} ||x_c(q_d) - x^t_d||^2$. Joint limit constraints are applied during the numerical procedure. The technique then generates a trajectory to q_d using the same dynamically-feasible interpolation scheme described in IV-A.

C. Inverse Kinematics with Predictive Safety Filter

IK+S is a simple extension of the IK method that performs a motion only if the generated trajectory is predicted to be safe. This method incorporates a collision-checking step by discretizing the trajectory (at a resolution of $0.5°$ in joint space in our implementation) and checking for feasibility at each state. Collision checks are the bottleneck in this process, but the method is fast enough to compute and verify trajectories in tens to hundreds of milliseconds.

D. *Potential Field Collision Avoidance*

Our fourth technique is a classic reactive potential-field (PF). Potential fields have been widely used for mobile robot obstacle-avoidance in real time [1] and have also been applied to robot arms [13]. In the basic scheme, a scalar field is constructed by summing an attractive potential with a minimum at the goal with repulsive obstacle avoidance potentials (Figure 5). To prevent collisions, repulsive potentials grow as the distance to an obstacle decreases to 0. The robot then follows the negated gradient of the field. We believed the PF method may be less tedious than IK+S to use in cluttered environments because repulsive forces allow the robot to make progress in the null space of a nearby obstacle rather than preventing motion entirely.

The attractive term consists of a velocity $\dot{x}_{att}$ to be applied to the clicked point $x_c(q)$ in the direction of x_d^t: $\dot{x}_{att} = C_{att}(x_d^t - x_c(q))$ where C_{att} is a gain constant. The repulsive force f_{rep} is applied to the point on the robot x_{rep} that minimizes the robot-obstacle distance, and has direction in the direction of maximum distance increase. The magnitude of f_{rep} increases with the inverse of the distance d, and decays to zero at a distance d_{safe}. In other words, $||f_{rep}|| = \max(0, C_{rep}(1/d - 1/d_{safe}))$ where C_{rep} is a gain constant. We also set f_{rep} to 0 when the obstacle approach velocity $\dot{d}$ is negative.

The $\dot{x}_{att}$ and f_{rep} terms are combined to produce a desired joint velocity $\dot{q}_D$ as follows:

$$\dot{q}_D = J_{att}^{+}\dot{x}_{att} + J_{rep}^{T} f_{rep} \qquad (1)$$

where J_{att} is the Jacobian of x_c, J_{rep} is the Jacobian of x_{rep}, and $\cdot^+$ denotes the pseudoinverse. To prevent violations of joint limits we clamp $\dot{q}_D$ to the velocity bounds $[-\sqrt{2(q - q_{min})\ddot{q}_{max}}, \sqrt{2(q_{max} - q)\ddot{q}_{max}}]$ (computed element-wise) such that the rest configuration attained through maximum braking satisfies joint limits. We then cap $|\dot{q}_D|$ to $\dot{q}_{max}$, and define a desired acceleration $\ddot{q}_D$ using a finite difference from the prior velocity. Finally, $|\ddot{q}_D|$ is capped to $\ddot{q}_{max}$ and sent to the robot.

While there are ways to construct PFs to guarantee convergence (e.g., [21]) they assume a fixed target and do not scale well to many-DOF systems and complex obstacles. Thus, significant tuning is needed to avoid pathological behavior when a target is moved in real-time. It can indeed be challenging to attain acceptable PF performance. Dynamic constraints induce oscillations in steep-walled channels of the potential field [14]. Discontinuities in closest points between obstacles and robot links cause discontinuous repulsion forces. One approach to reduce, but not eliminate this problem is to add repulsive forces at all robot links. We decided against this approach because we found tuning to be more difficult. Finally, bounded accelerations or torques can cause collisions when a robot is moving at high speed toward an obstacle because the repulsive field does not have enough time to slow the robot down. So to improve safety at high speeds we vary the safe radius with velocity, $d_{safe} = d_0 + \frac{\dot{d}^2}{2a_{max}}$, where d_0 is a constant and a_{max} is the estimated maximum acceleration of x_{rep} computed from the Jacobian and joint acceleration bounds. Our implementation sets $C_{att} = 10$ $C_{rep} = 5$, and $d_0 = 0.2\,\text{m}$.

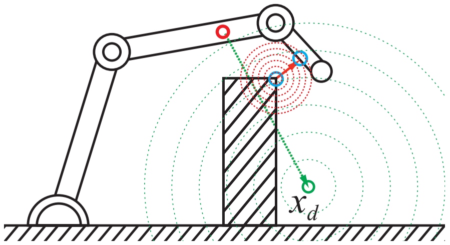

Fig. 5. The potential field strategy (PF) combines an attractive field (green) for the dragged point (highlighted in red) with a repulsive field (red) to avoid obstacles. Repulsive forces are applied at the point (cyan) on the robot arm that is closest to the obstacle.

E. *Real-Time Sample-Based Planner*

Sample-based motion planners such as Rapidly-Exploring Random Trees (RRTs) have been successful at planning collision-free motion for high-dimensional robot systems [16]. Because they broadly explore configuration space we were interested in their use to avoid the local minima problems of IK and PF approaches. Motion planning is more computationally expensive than the other approaches, and has traditionally been considered too expensive for real-time use in dynamic environments outside of relatively low-dimensional problems, e.g. 2D helicopters [9] and ground vehicles [20]. But due to advances in planning techniques and faster processors, sample-based techniques are beginning to show promise for real-time control of larger systems.

The MP control scheme is largely based on the RRT-like methods of [9, 20], with some modifications to account for the time-varying goals defined by the user and for real-time use. Like in the IK and PF approaches, the mouse motion defines a potential field function $V(q,t)$ which measures the distance from the clicked point $x_c(q)$ to the target point x_d^t. At each time step, the planner attempts to plan a dynamically feasible, collision free path that ends in zero velocity and also has a lower value of $V(q,t)$ than the endpoint of the current trajectory. Planning continues for a finite time interval Δ. After this time has elapsed, the new trajectory is taken if planning is successful, and the planner begins anew on the next step.

Our underlying planner is a variant of the RRT motion planner. The planner grows a tree of states, connected by local collision-free trajectories, forward in time. In order for the path to be applicable once planning is complete, the root of the tree is not the current state but rather the predicted state $(q(\Delta), \dot{q}(\Delta))$ along the current trajectory, propagated forward in time by Δ. We made the following performance enhancements to the basic RRT algorithm:

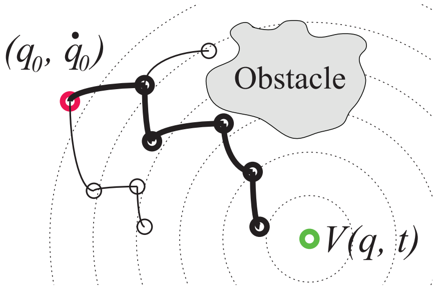

Fig. 6. The motion planning technique (MP) performs a sample-based exploration of state space starting from the current state $(q_0, \dot{q}_0)$ and picks the path that minimizes the user-controlled potential field V.

TABLE I
OBJECTIVE CHARACTERISTICS OF STRATEGIES

Strategy	Cartesian	Obst. avoid.	Controllable	Resp. time (ms)
JC	No	None	STLC	∼0
IK	Yes	None	STLC	0.2 (0.4) <2
IK+S	Yes	Filter	STLC	7 (24) <180
PF	Yes	Repulsion	No	50 (42) <150
MP	Yes	Planning	PC	168 (323) <1,900

- We extend the search tree by sampling extensions to stationary configurations sampled at random. The local planner constructs dynamically feasible interpolants between states and stationary configurations (a similar strategy was used in [9]). For every randomly generated sample, we generate a second configuration using a few steps of an inverse kinematics solver in order to get closer to the target.
- We use a lazy collision checking mechanism that delays expensive edge feasibility checks until the planner finds a path whose endpoint improves $V(q, t)$.
- To improve motion fluidity we devote 20% of each time step to trajectory smoothing. We used the shortcutting heuristic described in [11] that repeatedly picks two random states on the trajectory, constructs a dynamically feasible interpolating segment between them, and replaces the intermediate portion of the trajectory if the segment is collision free.
- We adapt the time step Δ by increasing it if the planner failed on the prior time step (which indicates a hard plan), or reducing it if the planner succeeded on the prior time step (which indicates an easy plan). We also reduce it in proportion to the magnitude of the cursor movement in order to respond faster when the user changes goals. More precisely, we scale the time step by e^{-cD} where D is the cursor travel distance and c is a scaling coefficient.

F. Comparison and Discussion

Some intrinsic characteristics of these strategies are summarized in Table I. The Controllable column lists whether the strategy gives the user the ability to achieve any robot configuration with a sequence of click-and-drag commands. JC and IK are small-time-locally controllable (STLC) in that any coordinated joint space motion can be composed of several user commands (with JC, at most 6; 6 also holds for IK away from singularities). IK+S is STLC in the interior of the free space. PF is not controllable because large repulsive forces prevent passing close to obstacles. MP uses a probabilistically complete (PC) motion planner and can theoretically access all configurations in the configuration space given enough time for planning. In practice the planner is given a time limit, which limits the ability of the robot to pass through narrow passages in the feasible space. The final column lists mean, standard deviation in parentheses, and maximum command response times of each strategy on a typical run of Scenario 2 in Figure 1 tested on a 2.8 Ghz PC. All obstacle avoidance techniques incur a loss of responsiveness, and MP is the slowest.

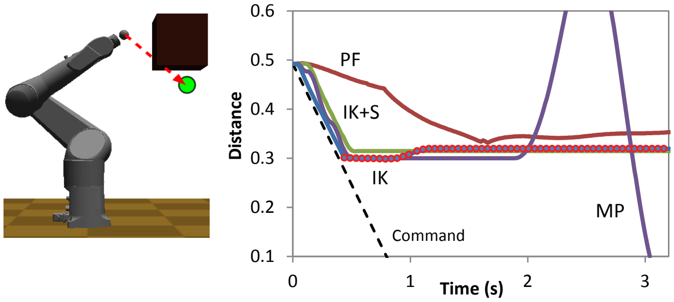

Fig. 7. A simulated command that drives the end effector in a straight line toward a target through an obstacle (top). Target distance over time for the four Cartesian techniques (bottom). Collisions are indicated as open circles. The IK+S and PF techniques avoid collision, but get stuck in local minima on the left side of the obstacle. At 2 s the MP technique begins to escape the minimum after ∼1.5 s planning delay.

Controllability may be a major factor in determining the flexibility of assisted teleoperation techniques. STLC methods allow a user to control the robot along any feasible path with arbitrary precision. PF and MP, on the other hand, do not give users control at the path level. This makes it more difficult for the robot to perform tasks that are not built-in to the implementation, for example, orientation control. This does not mean, however, that all STLC techniques are equally convenient; for example, orientation control is difficult using only Cartesian positioning.

Users may also be affected by other ways in which a technique conflicts with intuition. The robot may appear vaguely anthropomorphic to users because it consists of two shoulder joints, one elbow, and three wrist joints, but its joint axes and ranges of motion are actuall quite different from a human arm. This makes it somewhat unintuitive to control. For example, during Cartesian positioning, IK, IK+S, and PF can get trapped in local minima caused either by the robot kinematics or its joint limits. Obstacles also produce minima for the IK+S and PF methods (Figure 7). This may have significant effects for the performance of the user-robot system because user must in some sense learn the potential landscape and plan commands to escape minima.

Because the MP strategy can escape minima automatically it may offload the need for users to understand the complexities of the configuration space. On the other hand, the MP method has a peculiar characteristic in that it leads to unpredictable, jerky motions that may be viewed as unnatural (despite some smoothing performed in the planner). Our experiments investigate whether users can learn and adapt to such behavior.

V. User Trial Experiments

Our user trials were designed to capture dimensions of task performance that would be broadly characteristic of teleoperation scenarios in cluttered environments. In addition to quantitative measures of performance we also studied qualitative responses such as feelings of intuitiveness and user friendliness.

A. Experimental Procedure

22 subjects from the university with different backgrounds were recruited from mailing lists and flyers. Pre-trial surveys indicated 18 males, 4 females; 12 subjects aged 18–25, 7 aged 26–30, 3 aged 31 and above; 11 computer science majors, 11 other majors; all had 5 or more years of computer experience. All but 2 played video games at least once a month; 10 played video games at least once a month but not more than once a week, and 10 played at least once a week.

Subjects were assigned one of the methods at random, with 5 subjects in the JC and IK+S groups and 4 subjects in the IK, PF, and MP groups. Assignment was blind; subjects were not given the technique name or even knowledge that multiple techniques were being compared. Each subject was given a step-by-step tutorial on the GUI and were then given five minutes to test the controls in an obstacle-free environment.

We tested users on four different task scenarios presented in random order, each consisting of a set of obstacles and one or more goal points (Figure 1). They were chosen to be roughly representative of various challenging end-effector positioning tasks expected of teleoperated robot arms:

1) Track a series of six goals in open space, where collision is not a major concern.
2) Reach a goal that is set deep between two blocks. The opening between the two blocks is narrow, giving little room for the robot to maneuver.
3) Reach a goal over a low wall which requires simultaneous, precise positioning of the robot's wrist joints and joints near the base of the robot.
4) Reach a goal near an obstacle in a highly restrictive environment, consisting of a ceiling and a wall.

Although scenarios 2-4 all involve maneuvering around cluttered obstacles, the motions that are needed to avoid obstacles are highly different.

The user is instructed to command the robot to touch all the goals in the environment with the robot's end effector. Goals are presented in sequence; when one goal is reached the next appears. Five minutes were given for each scenario, at which point the user could opt to skip to the next scenario.

B. Task Performance Results

TABLE II
Performance results. Standard deviations in parentheses.

Strategy	Completed	Coll. free	Time comp. (s)	Time coll. (s)
JC	16/20	7/16	52.7 (48.4)	7.88 (19.8)
IK	**15/16**	3/15	36.5 (28.8)	1.71 (1.85)
IK+S	13/20	**11/13**	50.2 (38.3)	**0.04** (0.11)
PF	4/16	3/4	36.6 (9.35)	0.34 (0.69)
MP	**15/16**	10/15	**18.7** (12.4)	0.16 (0.31)

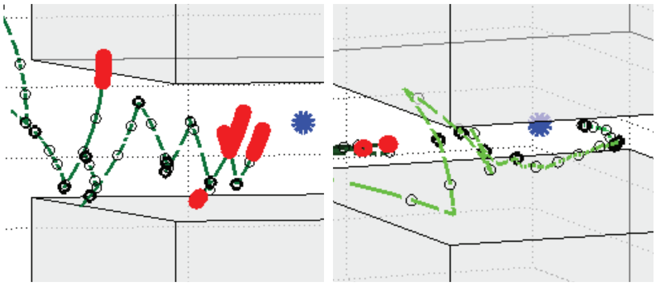

Fig. 8. Close-up of end effector traces for two completed trials in scenario 2 generated by using JC (left) and MP (right). Open dots indicate passage of time. Red dots indicate points in time at which the robot is in collision (collisions may be caused by links other than the end effector).

Performance data for each treatment group is shown in Table II. Results include the fraction of completed tasks, fraction of tasks completed without collision, completion time (only over completed tasks), and duration of collision. A task is defined as completed when the user reaches all goals in the scenario, and a task is incomplete when the user quits the scenario after at least five minutes of trying. Bold entries indicate the best in each column.

The JC method is the slowest among the five, while IK is fast and has a high completion rate. This agrees with our intuition that Cartesian control is more intuitive for novice users. Both methods collided in over 55% of cases. The incidence of collision for the JC method was lower than for IK, although individual collisions were more severe (Figure 8, left). Collision rate and duration were greatly reduced by all collision avoidance techniques, but only MP conferred a major improvement in completion time.

Results of statistical significance testing at the 95% confidence level are summarized as follows:

- Considered as a group, collision avoidance techniques reduce the incidence of collision during a trial from 65% to 25% and collision rate over time by 96%. Collision incidence differences are signficant at $p = 0.0011$ as determined through a Pearson's chi-squared test. Assuming that collision rates over time λ_0 and λ_c respectively of the baseline and collision avoidance techniques are Poisson, an exact conditional test shows that $\lambda_0/\lambda_c \geq 12.62$ with 95% confidence.
- MP leads to a reduction in completion time over all other techniques ($p = 0.008$, one-way ANOVA), averaging 59% faster.
- PF is by far the least successful in cluttered environments, with only 25% of scenarios completed. Only one of the four successes is in a cluttered scenario.
- Performance is not significantly correlated with demo-

graphic characteristics.

C. Post-trial Survey Results

Post-trial surveys asked users to reflect upon their experiences in the trial. Users were asked to rate on a scale from 1 (worst) to 10 (best) the following items:

- Overal quality of the visual display (Visual quality)
- Overall quality of the control interface (Control quality)
- Predictability of the control interface (Predictable)
- Safety of the control interface (Safe)
- Cooperativeness of the control interface in completing the task (Cooperative)
- Overall experience interacting with the arm (Experience)

Subject were also asked for open-ended comments about both the display and the control interface. Results are plotted in Figure 9. Group-wise tests using one-way ANOVA and pairwise tests using the Holm-Bonferroni method yielded the following results above an 85% confidence level:

- Differences in user response for overall experience are somewhat significant ($p = 0.124$), with between-group variation accounting for 67% of total variation. MP and IK both offer significantly better experience than PF ($p = 0.001$).
- Safety ratings for MP are higher than JC by 3.3 points with significance ($p = 0.014$). (Note that under the Bonferroni correction for multiple hypothesis testing this passes an 85% but not a 90% confidence threshold.)
- MP and IK have the highest ratings in cooperativeness; IK is rated more cooperative than PF ($p = 0.002$) by an average of 2.5 points, and MP is rated more cooperative than PF ($p = 0.005$) by an average of 3 points.
- No signficant results are obtained for user ratings in quality of visual display ($p = 0.652$), quality of control interface ($p = 0.576$), and predictability ($p = 0.734$).

D. Interpretation and Discussion

Overall, the MP technique conferred a major reduction in collision rate and completion time over direct teleoperation. Users did not seem to mind the loss of control, planning delays, and suboptimal motions produced by the planner. Cartesian control enables users to solve tasks faster and is preferred overall to joint space control.

The poor performance of the PF technique may be due to being tuned to prioritize obstacle avoidance. One user's frustration was expressed in a comment: "It would be helpful if I can have more control over the robot arm. Time by time, the robot arm will not follow my orders and that bothers a lot." To follow up on this hypothesis, we repeated our experiments using a less conservative set of parameters ($C_{rep} = 1$, $d_0 = 0.1$) on 6 subjects who were previously assigned to a non-PF technique. We found a large increase in success rate (24/24 trials) along with a large decrease in safety. Collisions occurred in 12/24 trials and time in collision (3.09 s, std. dev. 4.58 s) was high, second only to the JC method. Solution times were statistically indistinguishable the original PF. We suspect that further tuning might yield additional points along a control-safety tradeoff curve.

Another user found the IK+S technique frustrating; he interpreted that the robot remained still because it was performing a long computation, whereas it had actually rejected his infeasible command. Others may have had similar interpretations given the relatively high failure rate of 35%. Another user commented "[The MP technique] should give a sign indicating that the program is trying to calculating out a solution." This suggests that conveying internal status would help users form mental models of the robot's behavior more quickly.

Although IK+S and MP could be hypothetically 100% safe, they encountered minor grazing collisions (Figure 8, right). Approximately 3% of these collisions are an implementation artifact in that our collision-checker discretizes trajectories at a 0.5° resolution and may miss collisions between samples. Exact collision checking would help, but usually is more computationally expensive. The remaining 97% are caused by the sensing and actuation errors introduced into the simulation as described in Section III. To further reduce risk in practice the obstacle CAD models might be conservatively expanded. An interesting question for future work would be to study how such an approach would conflict with tasks that require manipulation or tool use.

Finally, we note that a limitation of simulation studies is that the perceived risks of crashing a simulated robot are less than those of crashing a physical robot. This may explain the high collision rate for the JC and IK techniques; in fact, many users found that forcefully sliding the robot across obstacles was a successful strategy. Nevertheless we expect similar effects will occur when users teleoperate physical robots that do not belong to them (due to moral hazard), are perceived as durable (due to a lower perception of risk), or perform imperfect autonomous collision avoidance (due to shifting of blame onto the robot). We expect that users with greater perception of crash risk will be more careful but substantially slower.

VI. Conclusion

This paper implemented and compared five assisted teleoperation strategies for controlling a 6DOF robot arm using click-and-drag input from a computer mouse. Results from a user study ($N = 22$) on reaching tasks in cluttered environments suggests that a real-time motion planning strategy improves safety and reduces task completion time compared to direct control technique, and is rated favorably by users. Interestingly, collision avoidance strategies that filter out or resist infeasible motion were not rated as highly and were generally viewed as uncooperative. This suggests that users are willing to tolerate loss of control, slower reaction times, and less predictable motions only for significant improvements in performance and convenience.

Based on these results, we hypothesize that assisted exploration of complex configuration spaces provides robust performance benefits. Future work should study whether this hypothesis generalizes to other input devices, such as gestures, and other robots, such as in multi-handed manipulation and

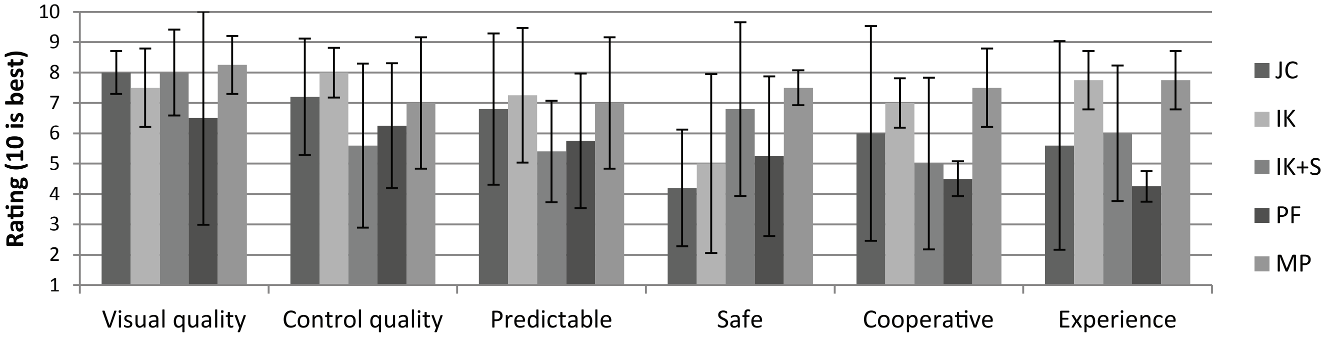

Fig. 9. Mean and standard deviations of user ratings across each treatment group.

multi-robot systems. We have also gathered detailed user input, camera viewpoint, and robot motion data from our experiments, and we hope to use this data to study how users respond, learn, and plan in conjunction with the robot. Finally, we are currently implementing and testing the motion planning assisted teleoperation strategy on the real robot arm in our lab.

References

[1] R. C. Arkin. Motor schema-based mobile robot navigation. *Int. J. Rob. Res.*, pages 92 – 112, 1989.

[2] C. J. Bell, P. Shenoy, R. Chalodhorn, and R. P. N. Rao. Control of a humanoid robot by a noninvasive brain-computer interface in humans. *J. of Neural Engineering*, 5:214 –220, 2008.

[3] D. A. Bell, S. P. Levine, Y. Koren, L. A. Jaros, and J. Borenstein. Design criteria for obstacle avoidance in a shared-control system. In *Proc. of the RESNA*, 1994.

[4] J. K. Chapin, Karen A. Moxon, R. S. Markowitz, and M. A. L. Nicolelis. Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex. *Nature Neuroscience*, 2(7):583 – 586, 1999.

[5] B. Dariush, M. Gienger, A. Arumbakkam, Y. Zhu, B. Jian, K. Fujimura, and C. Goerick. Online transfer of human motion to humanoids. *Int. J. Humanoid Robotics*, 6(2):265 – 289, 2009.

[6] M. Desai and H. A. Yanco. Blending human and robot inputs for sliding scale autonomy. In *IEEE Int. Workshop on Robot and Human Interactive Communication (ROMAN)*, pages 537 – 542, August 2005.

[7] W. Ferrel and T. Sheridan. Supervisory control of remote manipulation. *IEEE Spectrum*, 4(10):81 – 88, 1967.

[8] T. Fong, C. Thorpe, and Charles Baur. Advanced interfaces for vehicle teleoperation: Collaborative control, sensor fusion displays, and remote driving tools. *Autonomous Robots*, 11:77 – 85, 2001.

[9] E. Frazzoli, M. A. Dahleh, and E. Feron. Real-time motion planning for agile autonomous vehicles. In *American Control Conf.*, volume 1, pages 43 – 49, 2001.

[10] W. B. Griffin, W. R. Provancher, and M. R. Cutkosky. Feedback strategies for telemanipulation with shared control of object handling force. *Presence*, 14(6):720 – 731, 2005.

[11] K. Hauser and V. Ng-Thow-Hing. Fast smoothing of manipulator trajectories using optimal bounded-acceleration shortcuts. In *Intl. Conf. Rob. Automation*, 2010.

[12] B. House, J. Malkin, and J. Bilmes. The voicebot: a voice controlled robot arm. In *ACM Conf. on Computer Human Interaction*, 2009.

[13] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *Int. J. Rob. Res.*, 5(90):90 – 98, 1986.

[14] Y. Koren and J. Borenstein. Potential field methods and their inherent limitations for mobile robot navigation. *Intl. Conf. Rob. Automation*, pages 1398 – 1404, 1991.

[15] D. Kortenkamp, D. Keirn-Schreckenghost, and R. P. Bonasso. Adjustable control autonomy for manned space flight systems. In *Proc. IEEE Aerospace Conf.*, 2000.

[16] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[17] J. Luh, M. Walker, and R. Paul. Resolved-acceleration control of mechanical manipulators. *IEEE Trans. on Automatic Control*, 25:468 – 474, 1980.

[18] P. Marayong, M. Li, A. M. Okamura, and G. D. Hager. Spatial motion constraints: Theory and demonstrations for robot guidance using virtual fixtures. In *Intl. Conf. Rob. Automation*, pages 1270–1275, 2003.

[19] D. Perzanowski, A. C. Schultz, W. Adams, E. Marsh, and M. Bugajska. Building a multimodal human-robot interface. *IEEE Intelligent Systems*, 16(1):16 – 21, 2001.

[20] S. Petti and T. Fraichard. Safe motion planning in dynamic environments. *Intl. Conf. Intel. Rob. Sys.*, 2 (6):2210 – 2215, 2005.

[21] E. Rimon and D. Koditscheck. Exact robot navigation using artificial potential fields. *IEEE Trans. Robotics and Automation*, 8(5):501 – 508, 1992.

[22] D. Tolani, A. Goswami, and N. I. Badler. Real-time inverse kinematics techniques for anthropomorphic limbs. *Graphical Models*, 62:353 – 388, 2000.

[23] M. Velliste, S. Perel, M. C. Spalding, A. S. Whitford, and A. B. Schwartz. Cortical control of a prosthetic arm for self-feeding. *Nature*, 453(19):1098 – 1101, 2008.