Human Augmentation in Teleoperation of Arm Manipulators in an Environment with Obstacles [*]

I. Ivanisevic and V. Lumelsky
Robotics Lab, University of Wisconsin-Madison
Madison, Wisconsin 53706, USA
iigor@cs.wisc.edu

Abstract

This paper discusses a novel approach to combining human and machine intelligence in teleoperation of 6 degree of freedom (DOF) arm manipulators. Two algorithms are presented that can operate in near real time and can deal with an unknown environment. We capitalize on the fact that the operator needs help primarily in the areas where the arm is near the obstacles; this lowers the computational costs. The main, major linkage algorithm operates on the principle of a greedy search. Perception of the hints provided by the operator is used in a few basic rules that limit the directions of the search. Overall, the intention is to allow the operator to concentrate on “global” motion planning tasks and leave collision detection and local path finding to the machine intelligence. An experimental study in progress is presented that is expected to demonstrate the algorithms' performance and to suggest the types of situations where it will be the most effective.

1 Introduction

In this paper, a novel means for combining human and machine intelligence in teleoperation tasks is discussed. The goal is to improve the efficiency (that is quite low today) of some motion planning tasks for arm manipulators currently handled by the human operator, and reduce the risk of human error and resulting equipment damage due to collisions with obstacles in the work space. The idea is to "splice" in near real time human intelligence with machine intelligence. The word “near” here implies the operator's desire for a continuous motion, rather than any physics-imposed time constraints. Previous studies in our lab [1, 2] confirmed numerous observations that humans perform poorly in non-trivial teleoperation tasks (i.e. tasks which involve complex obstacles of varying shapes). Human performance gets worse as more degrees of freedom (DOF) are added to the arm manipulator. On the other hand, machine intelligence seems to have certain advantages in operating complex geometrical entities. this could be of help if a good way of combining the two types of intelligence is found.

Our previous work [2, 3] focused on assigning machine intelligence with the task of simplifying the motion planning problem presented to the operator through the use of the arm configuration space. The inherent disadvantages of that approach are the need to precompute the arm configuration space and difficulty of handling arms with more than 3 DOF. The latter is due to the problem of presenting four-dimensional or higher configuration spaces to the operator. In contrast, the method in this paper focuses on operation in work space - configuration space is never computed. This allows us to handle more complex arms (the case considered is that of a 6 DOF arm) in real time.

The test bed for our study has been a 6 DOF arm manipulator described in Section 2. We first attempted to identify the kind of tasks in which human performance is particularly unsatisfactory, and tried to delegate the corresponding subtasks to the machine intelligence. To this end, it has been observed that, when attempting to guide an arm manipulator in an environment with obstacles, humans have difficulty visualizing the interaction of the entire arm with the environment - they tend to

[*]This work was supported by the Office of Naval Research Grant N000149710871.

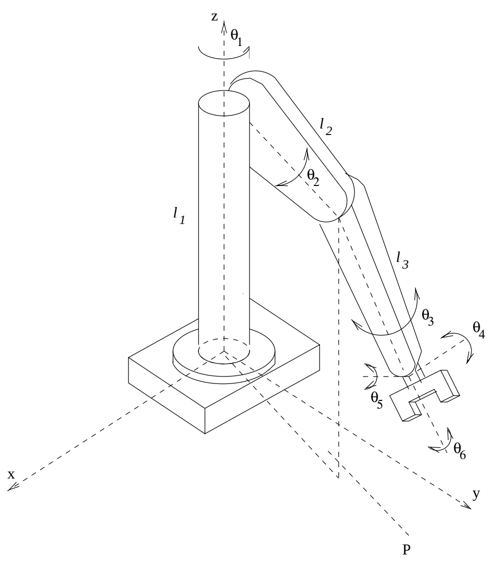

Figure 1: The 6DOF arm manipulator.

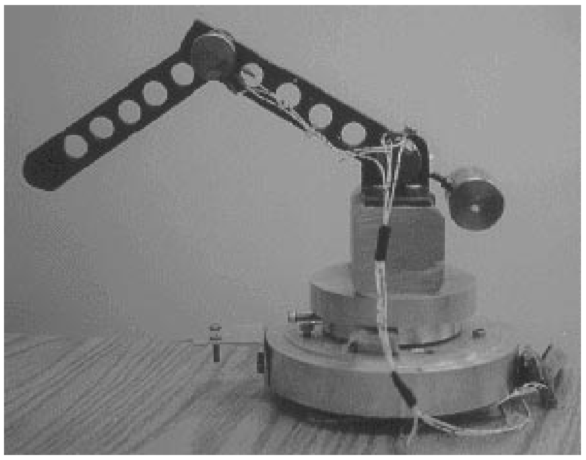

Figure 2: The master arm used as an input device.

focus on one part of it (usually the tip of the end effector, such as a wrist) and disregard the rest. This may result in particularly inferior performance in cases where obstacles endanger different parts of the arm body, especially if this happens simultaneously. In such cases, machine intelligence promises much help.

Another observation made was that, being unable to handle all 6 DOF of the arm, human operators tend to concentrate on a few DOF - typically on controlling the first 3 DOF (major linkage) of the arm. This suggests a scheme in which the arm control is split into two interacting parts, with the operator responsible for controlling the arm's major linkage, and the machine intelligence taking care of the wrist (with a means for the operator to take over the control of the wrist as well if necessary).

The expectation, therefore, is that the said division of responsibilities between the operator and the machine intelligence will allow the operator to focus on those aspects of the problem where humans are known to be good (such as global navigation decision making, or modifying the task's objective), while letting the machine intelligence take care of its areas of specialty – local navigation and collision avoidance. And, most importantly, this would allow real time operation that is otherwise not achievable today in a complex environment. To analyze this scheme, a study involving human subjects (similar to those in [2, 3]) will be carried out.

Below, the arm model and its accompanying interface are discussed in Section 2, the proposed algorithms appear in Section 3, and the results of this study and their discussion are given in Section 4.

2 The arm

Arm Structure. The arm manipulator used for our experiments (see Figure 1) has 6 DOF. For the purposes of arm control, it can be considered as consisting of two logical parts: the major linkage (first three DOF) and the minor linkage (the wrist).

The major linkage, Figure 1, consists of three links with revolute joints, allowing the arm to operate in three-dimensional space. The arm's first link, l_1, rotates about the vertical axis, producing joint values θ_1. Each of the other two links in the major linkage, l_2 and l_3, rotates in a vertical plane; their joints are labeled θ_2 and θ_3, respectively. Joint values for the major linkage are limited by mechanical stops, $|\theta_i| < 2\pi$, which are set to correspond to the potentiometer limits in the master arm (Figure 2).

The wrist (Figure 1) is attached at the end of l_3 by means of a three DOF joint that allows it to rotate around the three principal axes. The last three DOF are labeled θ_4, θ_5, and θ_6, respectively.

Arm Interface. The interface is divided into two parts - the physical interface used to input motion control parameters (master arm and keyboard) and the virtual world consisting of the arm and its surroundings.

The physical interface, called the *master arm* (Figure 2), is provided to the operator to control the major linkage. The wrist is not intended to be controlled by the operator; it is to be handled by the algorithm, as described in Section 3). However, keyboard commands are provided to allow the operator to take over and/or fine-tune wrist positioning if needed. Movement of the master arm is directly translated into movement of the virtual slave arm on the screen.

For visual feedback, the operator is presented with a view of the virtual world in which the arm "lives". This view can be adjusted using the interface, which provides a large number of camera angles, zoom in-out etc. For computational efficiency, arm links in the virtual world are modeled as generalized cylinders.

Another visual feedback, to assist the operator in controlling the arm in the vicinity of obstacles, simulates a haptic interface. Namely, a small bright red "contact" sphere is shown at those points of the virtual arm body where it comes in contact with an obstacle. Further, should the operator continue moving the virtual arm "into the obstacle" after the initial contact, a "skeleton" arm detaches from the virtual arm (which is frozen at the contact point) and follows the master arm motion. The operator would then realize that a contact took place, and would move the arm back to the point of contact (at which point the skeletal arm "reconnects" with the virtual arm) and try some other motion to avoid the obstacle. This feedback takes place only if the algorithm (which is activated upon a contact with an obstacle, see Section 3) fails to find a solution and to guide the arm in a safe way.

3 The Algorithms

Control of the major linkage. As mentioned in the Introduction, the goal of this algorithm is to assist the operator with controlling the major linkage. The algorithm is based on a greedy (best-first) search [4, 5], with some modifications that try to take advantage of the human operator's ability to see the "big picture".

As the operator moves the virtual arm using the master arm, a number of sub-goals, $g_1 ..., g_m$ (where g_m is the final arm configuration), are defined - one for each point sampled by the input device. The faster the operator moves the master arm, the wider the gap between g_i and g_{i-1} (since the sampling rate is constant). When the algorithm is not active, the trajectory between g_{i-1} and g_i is linearly interpolated in the arm joint space [7].

If the sub-goal g_i falls outside an obstacle but the linearly interpolated path between g_{i-1} and g_i crosses an obstacle, the algorithm is automatically activated. The objective is to produce a local path that would take the arm from the last "safe" location (somewhere between g_{i-1} and g_i) to g_i. Clearly, this path will not be linear in joint space. The other constraint is that the path must be found in real time – i.e. the operator should not notice delays in arm response time and preferably should not be aware a search is going on. This severely limits the maximum depth of the search (i.e. how many possible moves can be examined). Because of this, the maximum depth of the search should be experimentally adjusted according to the amount of processing power available while maintaining the real time requirement. The deeper the search that the system can afford, the freer the operator will be from local control.

In order to find the path from g_{i-1} to g_i, the algorithm performs a greedy search of the nearby environment. A greedy search operates by minimizing the estimated cost to reach a goal. It examines the possible moves at any given point in time and chooses the one which will take it the closest to g_i. If the move leads to a dead end or exceeds the maximum depth of the search, it backtracks to the last move and examines the next best successor. Invalid moves (ones that violate arm joint limits or result in collisions with obstacles) are excluded from this computation. "Closeness" to g_i is measured by the joint angle difference between the two configurations.

At any given step, the major linkage has 26 possible moves. For every link one can:

– increase the joint angle by minimum amount
– decrease the joint angle by minimum amount
– keep the joint angle unchanged.

The three possible moves per link result in 27 possible steps. One of these is the current configuration, so it can be eliminated. This means the search tree has up to 26 branches at every step; less if some branches are invalid. Therefore, a depth 5 search (which examines the next 5 possible moves in every direction) would have to explore up to 12 million moves. But, many of these moves may be redundant, and a good portion of the search may be taking place in an area that clearly (to the human operator!) has no solutions.

To eliminate some of those redundant search directions, we rely on the operator's ability to see the "big picture". Most notably, the 9 directions that deal with the first link rotating *away* from the sub-goal are never considered by the search. The assumption is that the operator has already determined that there is no need to rotate the first link away to reach g_i when approaching the obstacle. Based on preliminary experimental results, this assumption seems valid for many cases (see an example in Figure 3). Optimization of this kind reduces the search tree roughly by a third (to a maximum of 17 possible steps at any given point in time). Consequently, this allows the algorithm to search farther ahead within the same time constraints.

Furthermore, the algorithm will not examine positions which could not lead to g_i in the number of steps remaining. Consider a case where, 3 steps into the search with a maximum depth of 5 steps, we are about the examine a position which is 3 steps away from g_i (as determined by the distance heuristic). Clearly, this position can not lead to g_i since only 2 steps remain before the depth limit is reached. Therefore, it and its' successors need not be considered. While this seems like a minor improvement, in practice this type of search tree pruning will end up removing large portions from the bottom of the search tree, which is where the majority of the nodes lie. Hence we can expect significant time savings while employing this strategy.

The operator is notified when the algorithm is finished its work in two ways:

– if the algorithm has found a solution; then the operator will see the correct sequence of moves it took to complete the subtask.

– if the algorithm failed to find a solution; then a contact sphere and a skeleton arm will appear (as described in Section 2), signaling that the operator should backtrack and try a different path.

Note that since everything happens in real time, there will be no additional delays as a result of the algorithm's operation.

Under the greedy search, the arm tends to pursue one path until it finds a solution or reaches the maximum search depth. Compared to some other searches (e.g., those that take into account the path already traveled), this feature works well with hardware, since backing up the arm to pursue a different path is an expensive operation. One negative side here is that the arm can get stuck in local minima. But, the operator's ability to see the big picture allows them to avoid such cases by positioning the arm so as to take advantage of the known search parameters.

One improvement that is currently being tested is to allow the operator to override the direction of the search for any link (keyboard controls are added specifying the direction in which a link should search). The idea here is that the operator can see which directions of the search are more likely to succeed, and give the algorithm hints to lead the search in that direction. The big challenge is to make the interface effective in real time. This can be accomplished by using some basic assumptions about the operator's intent at any given moment, similar to the interface in [6]. The ultimate benefit of this modification would be to focus the search only on what the operator deems to be "promising" areas, and hence increase the likelihood of finding a solution.

Control of the minor linkage. The objective here is much simpler - just to keep the wrist from interfering with the motion of the major linkage. If possible, the wrist is kept in the final target configuration; if this changes as a result of the algorithm work, it tries to quickly return to this position.

With this objective in mind, the wrist algorithm operates in two modes. Normally the wrist's motion to the final target configuration is done by linearly interpolating in joint space of the wrist. This mode operates when the major linkage is moving normally and the wrist faces no collision with obstacles. Any time the wrist collides with an obstacle, the algorithm enters the collision prevention mode. The goal now is to find any wrist configuration that results in no collisions with the obstacle. This is accomplished by examining the neighboring 26 positions and choosing the first no-collision one found. If all positions result in a collision (a very unlikely event given that the immediately preceding step had no collision and the interpolated steps of the major linkage are very small), the operator is provided with the usual collision feedback prompting them to backtrack.

The only exception to this sequence is the case

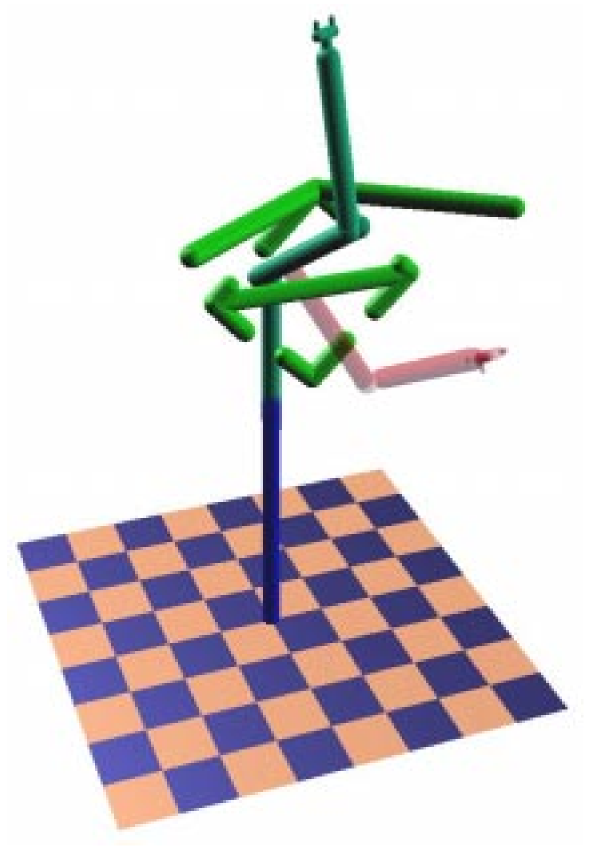

Figure 3: An example task where the major link algorithm performs particulary well.

when both the major and minor linkage are engaged in a collision simultaneously. In that case, the algorithm that controls the major linkage takes precedence and the wrist is frozen in its last known configuration and treated as part of the major linkage. The reason for this treatment is computational efficiency: having both algorithms running a 3D search concurrently would be a very expensive operation (a 6D search in real time). Finally, recall (Section 2) that the operator can override the wrist algorithm at any time with keyboard commands.

4 Experimental Results and Discussion

The set of experiments designed to test the effectiveness of the major link algorithm is currently in progress. Its format is very similar to the experiments done in [2, 3]. As of now, the algorithm has been tested on a few subjects. The results look promising - the most notable improvement is a very reduced number of collisions between the arm and the obstacles. Since the algorithm is local in nature and is overshadowed by the global decision making of the human operator, the improvements in the path length are likely to be modest.

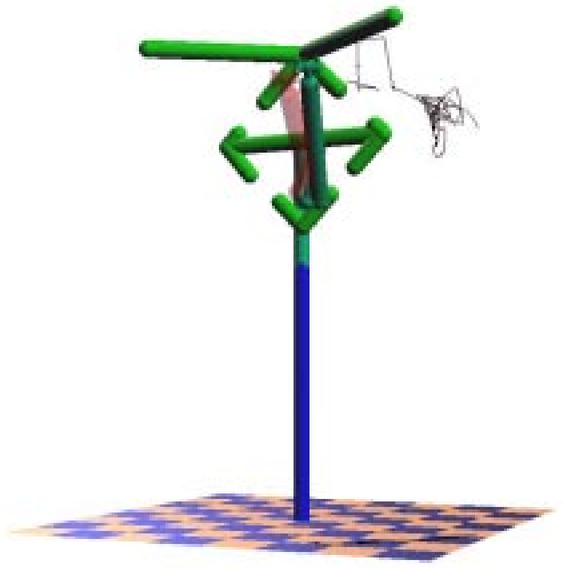

Figure 4: An example of unassisted performance by a human operator.

Figure 3 shows an example of a task that the subjects will complete as part of the experiment. Obstacles are lightly shaded, the arm is somewhat darker, and the target arm configuration is transparent. The task illustrates the types of obstacles and situations where the major link algorithm will be useful. Namely, the holes in the obstacles here are difficult to navigate with visual feedback. (And, as we know, haptic feedback for the entire arm body is still not feasible). On the other hand, the major linkage algorithm has no trouble navigating the holes if the user simply positions the arm near one hole and pulls the arm in the direction of the obstacle. The effectiveness of the wrist algorithm does not depend greatly on the shape of the obstacles.

Figures 4 and 5 show the performance in one section of the task of Figure 3, for the cases of unassisted and algorithm-assisted operation, respectively. The path of the tip of link l_3 (the point connecting the wrist with the major linkage) is shown as dark lines on the right side of the figures.

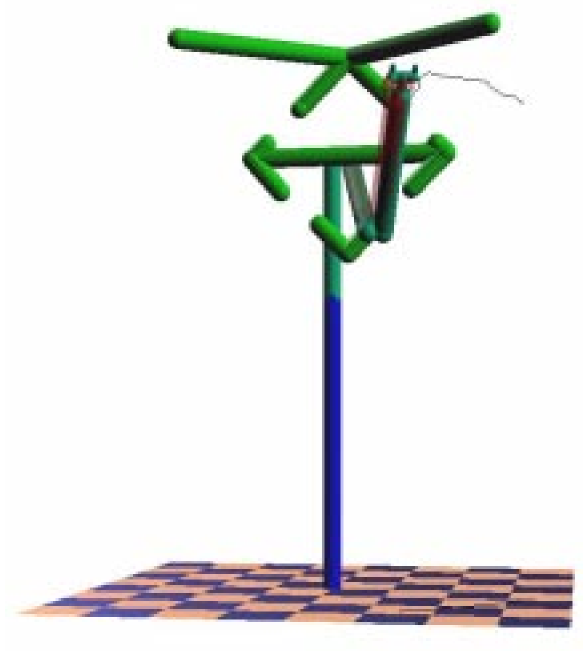

Figure 5: An example of algorithm-assisted performance by a human operator.

As seen in Figure 4, the unassisted operator had difficulty navigating the major linkage through the narrow opening between the obstacles. The pattern exhibited is familiar: hit the obstacle, backtrack, hit the obstacle again, backtrack and so on, until a solution is found, not rarely through blind luck. Hence, the path produced contains many jagged edges followed by a final straight line when a path was found. This movement resulted in about 20 collisions between the arm and the obstacle before it was completed.

On the other hand (Figure 5), when assisted by our major link algorithm, the same operator produced a much smoother path, with no collisions at all! The operator also spent a significantly less time to finish the task, and had an overall shorter path length. The latter benefit was not large enough if viewed as part of the overall performance in Figure 3, but was significant enough in the local task of Figures 4 and 5.

5 Conclusion

This paper addresses the question of augmenting human intelligence by machine intelligence in teleoperation tasks, in a team-like manner and in real time. The two algorithms proposed deal with some of the more difficult problems faced by the operator in such tasks. The first algorithm is concerned with precision navigation of the arm's major linkage in the vicinity of obstacle edges or holes. The second algorithm allows the operator to ignore the wrist and focus on the major linkage; collision-free motion of the wrist is being handled by the computer. Preliminary results indicate good performance of the system.

References

[1] F. Liu, Multivariate Analysis of Human Performance in Motion Planning, University of Wisconsin Robotics Lab Technical Report RL-97003, May 1997.

[2] I. Ivanisevic, V. Lumelsky, Operating in Configuration Space Significantly Improves Human Performance in Teleoperation, *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, April 1999.

[3] I. Ivanisevic, V. Lumelsky, A Human-Machine Interface for Teleoperation of Arm Manipulators in a Complex Environment, *Proceedings of the 1998 IEEE International Conference on Intelligent Robots and Systems*, October 1998.

[4] S. Russell, P. Norvig, Artificial Intelligence A Modern Approach, Prentice Hall, 1995.

[5] J. Noyes, Artificial Intelligence with Common Lisp, D.C. Heath and Company, 1992.

[6] R. Zeleznik, K. Herndon, J. Hughes, Sketch: An Interface for Sketching 3D Scenes, *Proceedings of the 1996 International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, August 1996.

[7] T. Lozano-Perez, Spatial planning: a configuration space approach, IEEE Transactions on Computers, Vol. 32, No. 3, February 1983.