Virtual Fixtures as an Aid for Teleoperation

Zoran Stanisic
Software Engineer
International Submarine Engineering Ltd.
Port Coquitlam, BC, Canada V3C 2M8
tel: (604) 942-5223
email: zstanisi@cs.sfu.ca

Eric Jackson
Program Manager, Control Systems
International Submarine Engineering Ltd.
Port Coquitlam, BC, Canada V3C 2M8
tel: (604) 942-5223
email: ejackson@ise.bc.ca

Shahram Payandeh
Associate Professor
Experimental Robotic Laboratory
School of Engineering Science
Simon Fraser University
Burnaby, BC, Canada V5A 1S6
tel: (604) 291-4290
email: shahram@cs.sfu.ca

Abstract

This paper presents the implementation of Virtual Fixtures in a teleoperation system featuring both manual and supervisory control modes. The fixtures improved the speed and precision of operation, reduced the operators workload, provided an alternative for path planners in supervisory control, and increased reusability of commands issued to the system. They also provided a link between supervisory and manual control modes, allowing the operator to easily switch from one to another.

Introduction

In general, control paradigms used to operate remote manipulators represent a spectrum from high level supervisory control to low level manual control [1]. High level supervisory control refers to operating the manipulator by specifying infrequent high level symbolic commands, while low level manual control assumes continuous teleoperation using a hand controller or similar master device. The robot control system developed for the CSA and described in this paper is designed with space teleoperation in mind and supports a wide range of control levels from high level supervisory to low level manual control. Virtual fixtures are a recent addition to this telerobot control system project.

Virtual fixtures are a task-dependent aid for teleoperation; they assist the operator by providing relevant visual and force information. Among problems that virtual fixtures are intended to address are: high operator load, complex and unpredictable algorithms for supervisory control, task performance degradation due to communication time delay, and limited reusability of paths generated by manual control.

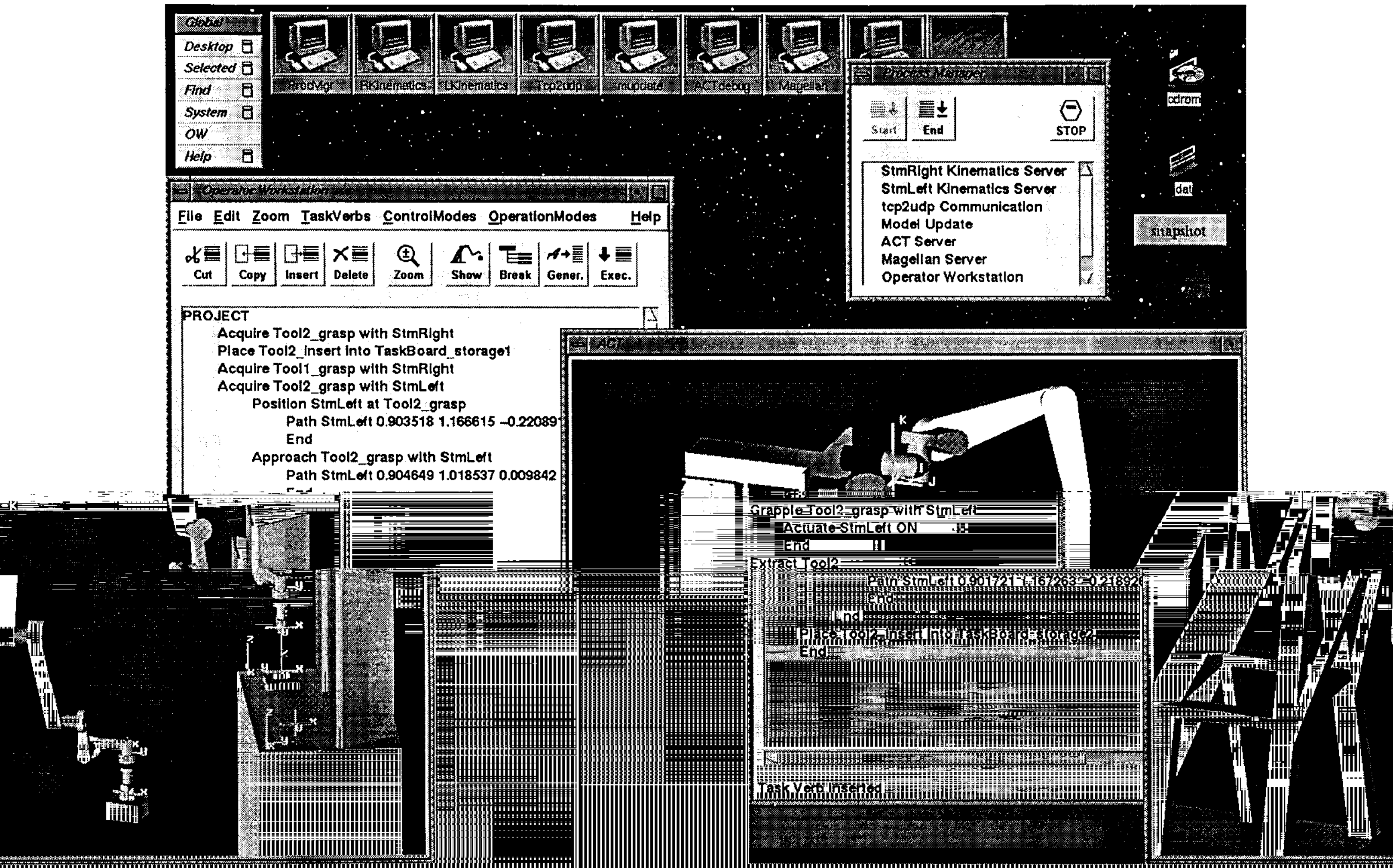
Previous implementations of virtual fixtures showed that they can increase speed and precision of operation, reduce operator workload, and reduce the effects of communication time delays [3,4]. These results were observed on manually controlled robotic systems. Because our robotic control system includes a range of higher level control functions such as extensive world modelling and work task modelling, these properties were used in our implementation of virtual fixtures. Our implementation allows the operator to use a variety of user interface modes, e.g. using manual control as an alternative to path planners when these algorithms provide unsatisfactory results.

In the system described, the control system sets up virtual fixtures in the environment based on the world model information and the data in the work task knowledge base, thus simplifying the manual teleoperation task. During a manual control task, virtual fixtures automatically generate symbolic commands appropriate to the tasks performed. Because each planned path contains task information that goes with it, we are able to significantly increase reusability of these paths. Using this approach, we improved the performance of supervisory and manual control modes, improved the operator's access to these modes, and improved the reusability of operator commands.

System Description

The teleoperation system described in the paper has been developed by International Submarine Engineering (ISE) Ltd. for the Canadian Space Agency (CSA) under STEAR contracts. The control system can be divided in two parts: the remote site consisting of dual 7dof manipulators and low-level control systems, and the local site which implements high-level control features and user interfaces (called the Operator Workstation).

The Operator Workstation software features extensive world modelling of robots and the environment. [Figure 1] The 3D model (shown in the right hand side of the screen) is used to help the operator visualize the environment; it can display either simulated information or feedback from the slave system. The model is also used by



StmRight Kinematics Server
StmLeft Kinematics Server
tcp2udp Communication
Model Update
ACT Server
Magellan Server
Operator Workstation

```
PROJECT
    Acquire Tool2_grasp with StmRight
    Place Tool2_insert into TaskBoard_storage1
    Acquire Tool1_grasp with StmRight
    Acquire Tool2_grasp with StmLeft
        Position StmLeft at Tool2_grasp
            Path StmLeft 0.903518 1.166615 -0.22089
            End
        Approach Tool2_grasp with StmLeft
            Path StmLeft 0.904649 1.018537 0.009842
Grapple Tool2_grasp with StmLeft
    Actuate StmLeft ON
    End
Extract Tool2
            Path StmLeft 0.901721 1.167263 -0.21892
            End
        End
    Place Tool2_insert into TaskBoard_storage2
    End
```

ation screen

Figure 1: The Operator Workst

rol algorithms, e.g. by path planners for calculations.

eleoperation control system we are ng is based on the NASREM control architecture[1]. Consequently, it features e work task modelling (task knowledge necessary to support supervisory control [Figure 1] Some example work tasks d in the knowledge base are shown in the ical task list in the main user interface (at the left hand side of the figure).

the con
distance

The te
describi
system
extensiv
base),
modes.
modelle
hierarch
window

As mentioned earlier, the system supports a number of control modes, ranging from high level supervisory control to low-level manual control. During supervisory control operation, the operator issues high level commands or "task verbs" (such as *acquire* and *place*). The system decomposes these commands into lower-level symbolic commands (E-Moves), and then interprets them into numeric commands (by invoking path planner algorithms). The decomposition of the task verbs

controlling a robot simulation, and will be used interchangeably with the task execution). The assistance provided by virtual fixtures may consist of force clues applied to a force-reflecting handcontroller, additional motion commands applied to the manipulator itself, or of imposed restrictions to the robot movements. When the teleoperation system features force-reflecting handcontrollers, virtual fixtures operate by providing force clues to the handcontrollers [4].

heir operation is implemented using is performed based on the knowledge kept in the Otherwise, t

mands and movement restrictions, case we will consider.

able 1: Task Verb knowledge

	Description
s	conditions that must be satisfied prior to the task execution
	conditions that cause the failure of the task execution
ns	conditions that must be satisfied to complete the task execution
	knowledge on how to break down the task into lower level tasks
	the algorithms performing the task preparation for supervisory control, e.g. path planners
	task-dependent control parameters, e.g. impedance control settings
e	force clues, motion commands, and geometry (motion restrictions) used for virtual fixture control

tures represent a task-dependent verlayed onto the environment, and ached to an object in that environment. , a virtual fixture can be attached to a iect to guide the robot gripper to the le point. As a geometry, the fixture ctions to the robot's movements, that the robot does not collide with rther, the fixture applies additional ands to the robot to attract it to the nt, therefore simplifying the ocess.

l in the virtual fixture definition is nt', and this notion has had a major our implementation: we incorporated

task knowledge base and the information from the world model.

[Figure 1] An example breakdown is shown in the hierarchical task list for the *acquire* task verb. This breakdown is done in two steps. The command is first broken into E-Move commands: *position*, *approach*, *grapple*, and *extract*. Secondly, each of these task verbs is interpreted (by invoking path planner algorithms) in order to obtain numeric commands suitable for execution by the lower level control system.

The robots can also be controlled in manual control mode via a hand controller device. The devices used in our system are a 6dof mouse, or a set of two 3dof joysticks. The input from the hand controller is transformed into Primitive level commands suitable for execution by the Robot Controller (effectively a stream of manipulator joint angles).

Besides providing supervisory and manual control modes, our teleoperation system attempts to combine the best features of both of these control modes by using the notion of virtual fixtures. We can define the Virtual Fixture Control mode as an extension of the manual control mode, where virtual fixtures are placed in the environment based on the knowledge typically related to supervisory control.

Virtual Fixtures

Virtual Fixtures can be defined as active, task-dependent software agents, whose purpose is to assist the operator during the preparation or execution of a task (task preparation refers to

motion com which is the

Ta

Property
preconditions
failure conditions
postconditio
breakdown knowledge
algorithms
parameters
virtual fixtur

Virtual Fix geometry o typically atta For example movable obj object's grapp imposes restri thus ensuring the object. Fu motion comma grapple poi positioning pro

The key word 'task depender influence on o virtual fixtures into our task knowledge base. [Table 1] The task knowledge base consists of a number of task verbs, each of them containing information relevant to preparation or execution of the associated robotic work task. The virtual fixture data is added to the task verb knowledge base, and describes ways in which the system can assist the operator during the execution of the task using a virtual fixture control paradigm. Some responsibilities of the virtual fixtures in our system are:

- generating additional movements of the robots, e.g. homing motions
- restricting motion or DOFs of the robots
- setting low-level controller parameters, e.g. impedances
- providing cooperative control of the dual robots

Let us consider how fixtures assist the operator during the execution of the *position* or *approach* task (involves setting all 6 degrees of freedom of the robot end-effector). The operator normally provides the handcontroller input, while the fixture provides additional motion commands. [Figure 2] These motion commands are presented using the arrows (note that the figure only shows commands that relate to achieving the proper position of the end-effector in the workspace, while the commands that relate to achieving the proper orientation are left out).

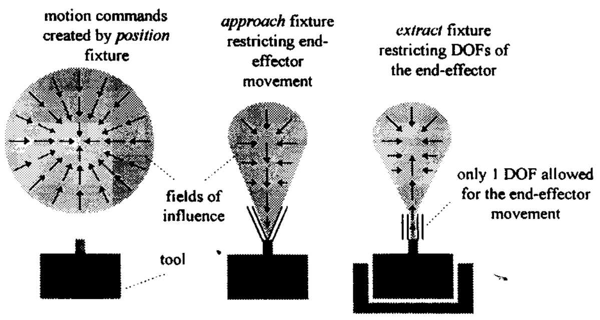
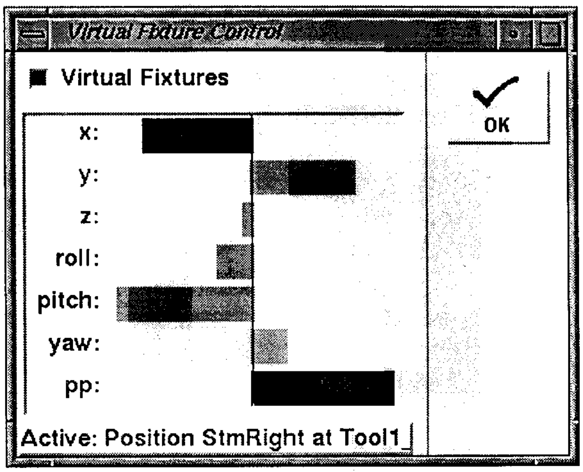

motion commands created by *position* fixture
approach fixture restricting end-effector movement
extract fixture restricting DOFs of the end-effector
fields of influence
only 1 DOF allowed for the end-effector movement
tool

Figure 2: Different clues provided by virtual fixtures

To avoid conflict with the operator's commands, the fixture considers only the degrees of freedom not specified by the operator at that moment. Thus, if the operator releases the handcontroller, the fixture will provide motion commands for all DOFs of interest. [Figure 3] As an example, the Virtual Fixture Control dialog box shows the handcontroller input from the operator (the dark bars), and the motion commands generated by the active virtual fixture (light bars). In this case, the operator sets the x coordinate and pitch plane angle, while the fixture controls other coordinates.



Active: Position StmRight at Tool1

Figure 3: Interacting with the operator during task preparation

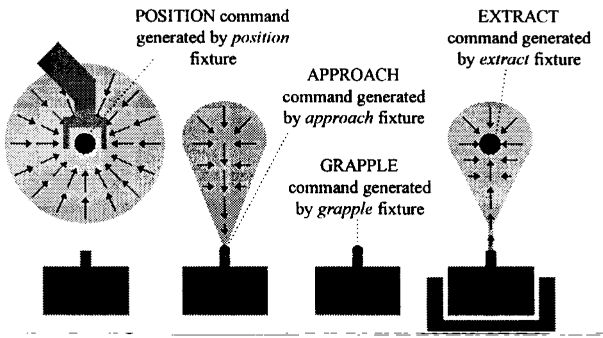
For some tasks it is beneficial to restrict the manipulator's motion (because of the interaction between robot and environment). For the *approach* task, we want to avoid collisions with the target, while allowing the operator to quickly approach the grapple position. For the *extract* task we want to simplify the extraction process, thus we restrict the degrees of freedom not involved in the process. [Figure 2] The restriction of motion or degrees of freedom is graphically shown by the double bars (only the restrictions related to position coordinates are shown).

In our system, the virtual fixtures are also responsible for setting low-level control system parameters. For example, we want a different type of robot control algorithm when the robot is in free space compared to when the robot is in contact with the environment. Based on the task knowledge, the virtual fixtures set the appropriate control parameters for the task being executed.

Finally, virtual fixtures are used to simplify the simultaneous control of dual robots. Suppose we want to perform a task where the coordinated motion of both robotic arms is required (e.g. to carry a heavy object). The virtual fixture knows that both robots need to be moved at the same time, and that additional constraints exist on the robots (they are coupled). In this case the virtual fixture uses the handcontroller input provided by the operator to generate setpoints for both robots, taking the task constraints into account. As a result, the operator is able to simultaneously control both robots using only one handcontroller device. After the coordinated task is finished (one of the robots ungrapples the object), the system goes back to operating only one robot at the time.

Operational Issues

the execution of the task using the virtual fixture control mode. When the task is ended (the system automatically performs this check based on sensory information and the postcondition knowledge of the task verb), the virtual fixture is removed or deactivated. The fixture can also be removed if the preparation of the task fails (based on the failure conditions).


POSITION command generated by *position* fixture
APPROACH command generated by *approach* fixture
GRAPPLE command generated by *grapple* fixture
EXTRACT command generated by *extract* fixture

e 5: Command generation by *acquire* fixture

used as an aid for manual control, the work not known in advance (by the control). Activation of virtual fixtures in this case reactively based on the task preconditions , one fixture is activated for each task preconditions (defined in the task ge base) are satisfied. When the task is r the failure conditions are satisfied, the s deactivated.

used as an alternative for path planners, a aid for manual control, virtual fixture combines the supervisory and manual approaches. When used as alternative for nners, the fixtures make sure that the y generated path plans are placed within traints of the work task. Also, during the on process, task knowledge is used to the operation (by placing appropriate ixtures).

manual control virtual fixtures can new high level commands (task verbs) to the actions that were specified by the during the work task. This task verb on is performed when a virtual fixture is ed (i.e. when the task postconditions are). [Figure 4] Consider the *acquire* task

The virtual fixture control mode can be used within two contexts: when providing an alternative for path planners during supervisory control, or when providing a task-dependent aid for manual control. Our system currently implements only the first case because of its reduced operational overhead. However, it is easy to use one instead of the other, and the major difference lies in the activation and deactivation of virtual fixtures.

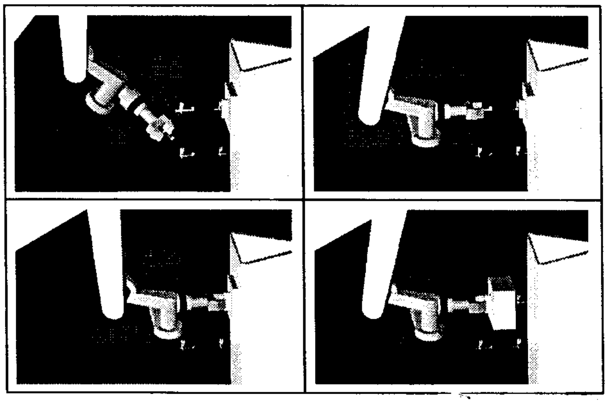


Figure 4: The acquire process

When used as an alternative for path planners in supervisory control, the task that we want to execute is known in advance (it is specified by the operator). The appropriate fixture is placed in the environment (activated) when the operator starts

Figur

When task is system is done Basically whose knowled ended o fixture is

Whether or as a control control a path pla manually the cons generati simplify virtual f

During generate describe operator generati complet satisfied consisting of the *position*, *approach*, *grapple*, and *extract* subtasks. Whenever a subtask is finished, the appropriate task verb is assigned to the path that was specified by the operator. [Figure 5] The positions of the end-effector at which these subtasks were completed are visually represented by black dots. When all four subtasks are complete, the *acquire* task verb is generated, and the previously generated subtasks are assigned to it.

System Evaluation

One of the benefits of virtual fixture control as compared to manual control is reduced operator workload. The fixture participates during the execution of the task, and offloads part of the work from the operator. For example, the operator could take care of setting the position coordinates, while the fixture could set the orientation. This would be an example of shared control of the manipulator.

In some cases, the fixture would be able to perform the task appropriately without any input from the operator. Thus the control responsibility would switch between the operator and the virtual fixture, which is an example of traded control.

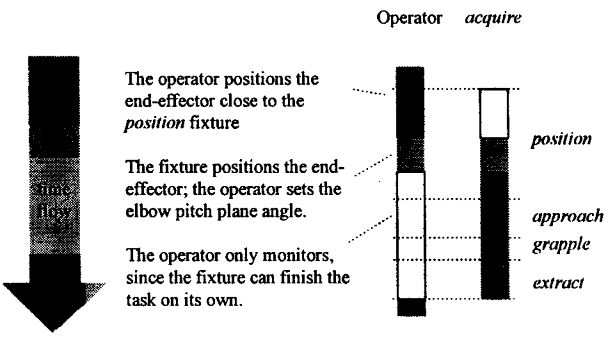

Operator
acquire
The operator positions the end-effector close to the *position* fixture
The fixture positions the end-effector; the operator sets the elbow pitch plane angle.
The operator only monitors, since the fixture can finish the task on its own.
time flow
position
approach
grapple
extract

Figure 6: *Acquire* fixture reducing operators' load

[Figure 6] Consider the real example of the *acquire* task performed using virtual fixture control. The operator is mostly involved in the first phase of the process (generating a collision free path). Once the end-effector is close to the fixture, a period of shared control exists where the operator sets the pitch plane angle (the redundant degree of freedom for the 7dof arm) if necessary, followed by autonomous fixture operation in finishing the task.

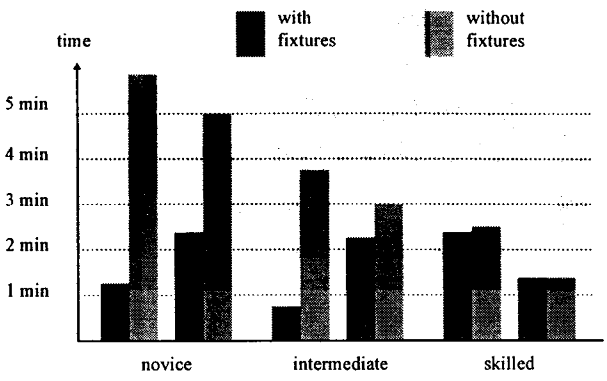


Figure 7: Performance measurements for the Position task

Virtual fixtures can also increase the speed and precision of robot operation. We conducted a simple test where several operators with skills ranging from novice to skilled performed a positioning task at their own pace. The task was first performed with virtual fixtures and then without them, and the duration of the operation was measured.

[Figure 7] The results indicate that the virtual fixtures significantly improved the operation speed for novice and intermediate operators. Further, all the operators considered virtual fixture control easier than manual control.

Conclusions

The benefits of implementing virtual fixtures in our teleoperation system are multiple. With virtual fixture control we obtained a performance increase and reduced operator workload compared with ordinary manual control.

The virtual fixture control provided a scheme where manual control can be used as an alternative to path planners, while preserving the task information typically related to supervisory control. It also provided automatic command generation when used instead of manual control, thus establishing a closer relation between the supervisory and manual control paradigms.

Finally, we increased the reusability of paths generated with manual control. Since during the operation of the manipulator we obtained not just numeric paths, but their corresponding symbolic commands as well, we can easily check whether those path plans can be reused when that command is executed again.

Because of the initial modelling required for the virtual fixtures, they are appropriate for repetitive tasks, or those where speed and precision are essential. An example of future work can be an application of virtual fixtures to remote laparoscopic surgery.

References

[1] J. S. Albus, H. G. McCain, R. Lumia. "NASA/NBS Standard Reference Model for Telerobot Control System Architecture", National Bureau of Standards Robot Systems Div., Dec 1986.

[2] R. C. Arkin. "Three-dimensional Motor Schema Based Navigation", Proceedings of the NASA Conference on Space Telerobotics, vol. 1, pp 291-299, Pasadena, CA, Jan-Feb 1989.

[3] L. B. Rosenberg, "The use of Virtual Fixtures to enhance telemanipulation with time delay", DSC-Vol. 49, Advances in Robotics, Mechatronics, and Haptic Interfaces, ASME 1993.

[4] C. P. Sayers, R. P. Paul. "An Operator Interface for Teleprogramming Employing Synthetic Fixtures", Presence, Vol. 3, No. 4, pp. 309-320, Fall 1994.