Chapter 6

Internet Telerobotics

Dr. Mayez Al-Mouhamed
Professor, Computer Engineeing Department
King Fahd University of petroleum and Minerals

A reliable real-time client-server telerobotic system is presented using a *Distributed Component Framework* to promote software reusability, ease of extensibility, debugging, and data encapsulation. .NET remoting is used for automatic handling of the network resources and data transfer while isolating the components from network protocol issues. A client-server transfer of live stereo video provides the operator 3D views of the slave scene with augmented reality (AR) framework and services. Overall distributed framework and design independence improves the portability and modularity of the proposed telerobotic system. A multi-threaded execution is proposed for streaming of force, command, and for the transfer of live stereo video data. The proposed framework provides a useful integrated software and hardware environment to enhance man-machine interactions using stereo visualization and AR in real-time telerobotic systems.

6.1 Introduction

Robotic technology [28, 43] is enhancing surgery through improved precision, stability, and dexterity. Depth perception and haptic sensing [48, 27, 31] are needed at the surgeon's console. Needed media data have increasing sampling rate which requires tradeoffs between quality and sampling frequency. A central problem is how to design a man-machine interface for the implementation of effective telerobotic systems that extends human manipulative skills over a distance. For this a real-time framework that efficiently and pervasively integrates physical robot, sensors, software, and hardware is highly desirable.

Internet telerobots [21] (ITRs) can be driven by anyone and include robots that navigate undersea, drive on Mars, visit museums, float in blimps, handle protein crystals, etc. The study of ITRs enabled the assessment of communication delays and the analysis of supervisory control.

A WWW remote supervisory control architecture combining computer network in an autonomous mobile robot with collision avoidance and path planning is proposed in [37]. Sheridan[47] defines a model of supervisory control. The operator provides system commands to a human interactive computer that controls a task interactive computer which translates higher level goals into a set of commands.

Hu et. al.[29] proposed JAVA for network interfacing and video as well as the use of C++ for the robot controller for Internet-based telerobotic system. TCP/IP sockets [10] is one highly reliable method for sending information over the Internet using VxWorks real-time multitasking system. By prioritizing the tasks the user can control the order of task execution and the amount of CPU time allocated to each task.

A component-based distributed control for Internet tele-operations using DCOM and JAVA is proposed in [23] to explore the foot of a volcano using a mobile robot. JAVA is used for database connectivity and path planner GUI and DCOM for network connectivity. JAVA virtual machine (MS VM) is proposed to bridge the gap between JAVA and DCOM. The robot position feedback is provided by two paths, one from the GPS (Global Positioning System) data and the second one from the visual feedback.

A DCOM-based distributed component system [24] is proposed to integrate web technologies and telerobotics together with environmental constraints. The main feature is a DCOM/ActiveX based supervisory control server operating over the Internet. Operator views 3-D model, control paths, and issue commands through supervisory control. Ho [22] developed an Internet based telerobotic system using JAVA and VRML. JAVA-based frame is used as grabbing software to move an image from camera to main-memory.

A CORBA-based distributed robot object model [49] is proposed for mobile robotics. Object communication uses timely leased communication patterns associated with the availability of some resources. A distributed perception strategy [26] is proposed for internet robotics using the perception-motion process (behavior) and task planning (mission). The objective is a robot that can find neighboring Internet sensors for improved reliability. For this, behavioral objects are transported via Internet to favor local interaction which improves stability and synchronization when Internet becomes unreliable. A distributed robot architecture [33] is proposed for modularity for integrating learning aspects in a mobile robot. The robot functions are designed as hardware agents whose resources and notifications are managed by an agent manager. Behavioral functions are flexible programs that are created by task knowledge learning and managed by an agent manger. An environment model is used for tasks, skills, and objects. Distributed objects communicate using CORBA publish-subscribe mechanism.

The classical remote procedure call is redesigned for resilience, transparency, and event-driven notification [38]. CORBA is used to connect objects across heterogenous processing nodes. Using CORBA event service two interaction models are proposed: (1) an event channel be operated using proxies for client and server for communication with multiple clients where active servers push events on registered clients, or (2) a notification scheme where a client dynamically subscribes to a set of events which define event priority and lifetime.

A subsumptive, distributed, vision-based robotic architecture is proposed in [15]. To improve system fault tolerance different behavioral strategies are coded into multiple loops (fine-to-coarse) where each loop adds to the competence level of the loops below. The loops are independently processed and their results ranked by an arbitrator using application-based criteria to decide which loop should instantly control the robot. The decoupling of video processing and communication is similar to proposed multithreaded execution for concurrent frame acquisition and transmission.

In this chapter a reliable, real-time, telerobotic framework using object-oriented distributed components technology is presented. .NET remoting technology is used to allow a software component to be developed in one computer and used in many different computers. All updates on an instance of a component (object) can be seen in all connected computers as if it is a local component. Stereo vision greatly enhances the operator's efficiency but imposes severe requirements in terms of bandwidth to transfer real-time video data in a client-server environment. A client-server framework for grabbing and live video transfer is proposed.

The organization of chapter is as follows. First, the robotic aspects of the multi-threaded distributed framework are presented. Second, the software system design is presented. Third, the live stereo video transfer is presented. Fourth, the results are briefly presented and a comparison to other implementations is presented together with a short conclusion.

6.2 A multi-threaded distributed framework

The aim is to extend natural eye-hand motion coordination through a computer network while preserving human manipulative dexterity in scaled working environments. The objective is to de-

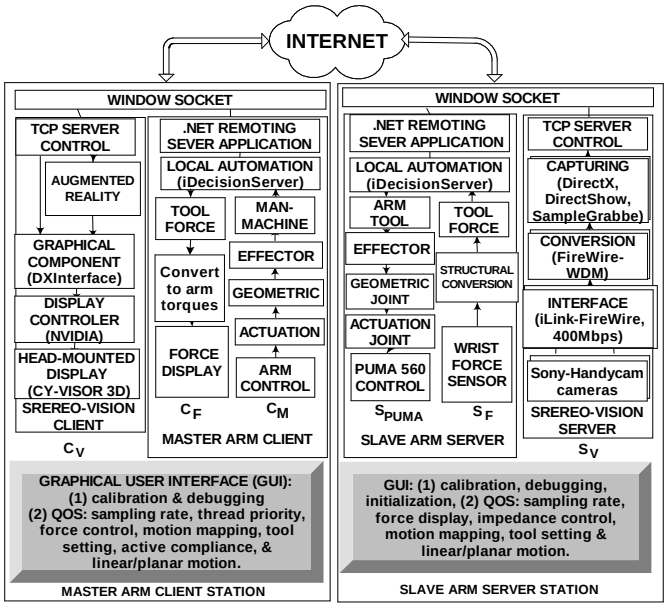

Figure 6.1: A layered representation of client-server telerobotic system

velop a multi-disciplinary telerobotic research environment integrating motion, vision, and haptic senses to experience telerobotic system interactions, man-machine interfacing, and computer aided teleoperation (CAT).

We present a *Multi-Threaded Distributed Framework* (MTDF) for Telerobotics. The proposed framework consists of components and patterns that establish how components interact with each other. The server station components (Section 6.3.1) are (1) a slave arm components that consist of a robot PUMA (S_{PUMA}) and a Force (S_F) components, and (2) a video (S_V) component (Section 6.6.2). The client station components (Section 6.3.6) are (1) a master arm component that consists of a Motion (C_M) and a Force (C_F) components, and (2) a video component (C_V). All client (server) components are concurrently run as independent threads on the client (server) computer. For example, one pattern consists of real-time thread S_V (also S_F and C_M) that is logically interconnected to C_V (also C_F and S_{PUMA}) to which it sends data through the network. A layered representation of proposed client-server telerobotic system is shown in Figure 6.1. In the following we present the robotic aspects of the proposed *Multi-Threaded Distributed Framework* (MTDF).

6.2.1 Server motion coordination

The kinematics of the slave arm is represented by means of three frames: (1) a fixed world frame (R_w) at arm origin, (2) an effector frame (R_e), and (3) a user defined tool frame (R_t). The controllable frame R_e is represented by its 3×1 position vector ($E_w(\theta)$) and its (3×3) orientation matrix ($M_w^e(\theta)$), where θ is the slave arm joint vector and w refers to R_w. The tool frame R_t is defined by its position vector T_t and its orientation matrix M_e^t of frame R_t with respect to frame R_e. The position of the tool point is defined by $T_w = E_w + M_w^e(\theta)M_e^tT_t$. The slave station receives a command to translate the tool frame R_t by ΔT_w and to rotate it by ΔM_t. The operator motion can be efficiently mapped onto the tool frame when the translation is specified in tool frame, i.e. ΔT_t. The new arm controllable position vector is:

$$\Delta E_w = M_w^t(I - \Delta M_t)T_t + \begin{cases} \Delta T_w & \text{Operator-tool} \\ M_w^t\Delta T_t & \text{Operator-world} \end{cases} \tag{6.1}$$

where $M_w^t = M_w^eM_e^t$. The new effector orientation matrix (controllable) becomes $\Delta M_e = M_e^t\Delta M_tM_t^e$. To avoid cumulative errors in the above equation Gram-Schmidt orthogonalization [1] is used to guarantee that the columns of ΔM_e represent unit vectors that are orthogonal. The PUMA reads current joint vector θ and computes effector position $E_w(\theta)$ and orientation $M_w^e(\theta)$.

The target effector position and orientation are $E_w^+ = E_w(\theta) + \Delta E_w$ and $M_w^{e+} = M_w^e(\theta)\Delta M_e$. The inverse kinematic model $\theta^+ = G^{-1}(E_w^+, M_w^{e+})$ provides the joint vector θ^+ that moves the tool by the commanded translation ΔT and rotation ΔM. θ^+ is sent to the slave arm motion controller. Incremental change in operator hand frame R_{op} is superimposed on tool frame R_t. For example, when R_{op} is tilted the remote tool frame R_t is tilted by the same angle.

6.2.2 Server active compliance

The force sensor consists of two parallel plates p_1 (frame R_e) and p_2 (frame R_s) interconnected by three elastic links. The force sensor used is developed by the authors [6]. The motion of p_2 with respect to p_1 is measured by a (1) translation vector ΔS_e, and (2) orientation matrix ΔM_e. The sensor structure allows finding ΔS_e and ΔM_e as functions of the six sensing signals. The sensor frame R_s is located between R_e and R_t. An external force applied to the tool causes a deflection vector $\Delta T_e = \Delta S_e + (\Delta M_e - I)M_s^t T_t$ to the tool frame origin as well as a change ΔM_t in R_t orientation as $\Delta M_t = M_t^s \Delta M M_s^t$. Since $M_e^t = \Delta M M_s^t$ the tool deflection vector is $\Delta T_t = M_t^s \Delta M^{-1} \Delta T_e$.

Active Compliance (AC) control consists of converting the measured force into an incremental motion for the slave tool. The force (F_t) and moment (C_t) vectors are computed using vectors ΔT_t and $M_t^s \Delta M M_s^t$. Using the passive compliance matrices for linear (K_l) and rotational (K_r) motion of the tool we compute the force $F_t = (f_x, f_y, f_z)^t = K_l \Delta T_t$ and moment $C_t = (c_x, c_y, c_z)^t = K_r \Delta M_t$ vectors. F_t and C_t are used to: (1) display the reflected force feedback at the client station, and (2) implement active compliance mechanism as a force control strategy as shown in Section 6.3.4.

6.2.3 Assistance functions

Server teleoperation assistance functions can be activated by buttons at the client station. The assistance functions are:

1. Relative or world mapping: operator motion is mapped to: (1) world frame, or (2) tool frame in tool manipulation tasks.

2. Floating tool mapping: operator motion is dynamically mapped to the slave tool frame by defining the tool frame position and orientation at some point of interest for the task. This may greatly reduce the number of iterations done by the operator to set up the manipulated object in a given position and orientation.

3. Planar or linear motion: constraining some tool motion axes to linear or planar motion and leaving the other axes under direct operator manual control.

4. Force Control: implements active compliance by continuously sensing the force exerted on the tool, evaluates a proportional force error based on a desired force, and converts the error into a position increment to reduce the force error. Here the user may select setting up active compliance over a sub-set of tool axes while other axes are kept under position control. In this case the selected components of computed force F_t and moment C_t vectors are feedback as elementary tool translation $\Delta T = AF_t$ and rotation $\Delta M = BC_t$, where A and B are two 3×3 matrices that determine the selected axes.

6.2.4 Client CAT support

The client station have a set of button-controlled teleoperation functions which are: (1) real-time rendering of the operator motion, (2) indexing, (3) space scalability, and (4) impedance control. In the following we present each of these functions.

Real-time rendering of the operator motion and display of force feedback are implemented as follows. There are two major inputs: (1) the joint vector read from position sensors and (2) force data coming from the remote side. Arm kinematic model $G_M(\theta)$ allows computing the current operator

hand position vector X^+ and orientation matrix M^+, where θ is the arm joint vector. Using last references X and M, it computes the variations $\Delta X = X^+ - X$ and $\Delta M = M^t M^+$ with respect to reference. The client sends the above computed variations to the slave arm as an incremental motion command for the slave tool frame. The client also outputs the received force feedback as master arm motor torques to display the force feedback on operator hand.

The indexing function is defined as follows. In direct teleoperation the variation in operator hand position and orientation $(X^+ - X, M^{-1}M^+)$ is evaluated and transmitted to server. During indexing the system continuously sets up the reference to current (X, M) and disable transmission to slave arm.

The scalability function is defined as follows. The increment in master position vector (ΔX) and orientation matrix (ΔM) are scaled-down before being transmitted to the server. For this the variation in the operator hand orientation matrix (ΔM) can be seen as made of three euler angles, i.e. $\Delta M = R_x(\alpha_x)\ R_y(\alpha_y)R_z(\alpha_z) = R_{xyz}(M)$, where R_u is a rotation matrix about axis u and R_{xyz} is the product of three rotation matrices sets for ΔM. Since ΔM is known, we inverse the above equation and find the three angles as $(\alpha_x, \alpha_y, \alpha_z) = R^{-1}_{xyz}(\Delta M)$. Using an operator-controlled scale factor s, the scale function becomes:

$$(\Delta X, \Delta M) = ((X^+ - X) * s, R_{xyz}((R^{-1}_{xyz}(\Delta M)) * s))) \tag{6.2}$$

To avoid singularities, the three Euler angles are computed for the variation in the operator orientation matrix ΔM. Due to speed of operator motion and real-time mapping, the computed angles are small which avoids the singular area.

The impedance control is implemented as follows. The master arm dynamics [14] determine the motor torque τ as $\tau = D(q)\ddot{q} + C(q, \dot{q}) + G(q)$, where q joint vector, $\dot{q}$ joint velocity, $\ddot{q}$ joint acceleration, $D(q)$ is the inertia matrix, $C(q, \dot{q})$ is the coriolis and centrifugal coefficients, and $G(q)$ is the gravity term. We compute terms $C(q, \dot{q})$, $G(q)$, and $D(q)$ which enables finding the motor torque:

$$\tau = \alpha\ddot{q} + \beta\dot{q} + C(q, \dot{q}) + G(q) + \tau_{ff} \tag{6.3}$$

where term $\alpha\ddot{q} + \beta\dot{q}$ is generated based on the operator motion, terms $C(q, \dot{q})$ and $G(q)$ are used to compensate for dynamic effects and gravity, and τ_{ff} is the force feedback. The overall dynamic motion equation becomes:

$$\tau_{ff} = (D(q) - \alpha)\ddot{q} + \beta\dot{q} \tag{6.4}$$

where term $D(q) - \alpha$ represents the reduced master arm inertia (impedance) and $\beta\dot{q}$ is a motion damping factor. The values of the parameters α and β are experimentally determined based on a performance/stability criterion.

6.3 Software system design

6.3.1 Server side components

The server components are: (1) PUMA Component S_{PUMA}, (2) Force Sensor Component S_F, and (3) Decision Server Component. In addition to these components, we also have three interfaces known as (1) Proxy Robot Interface (2) Force Sensor Interface, and (3) Decision Server Interface which will be presented subsequent sections.

6.3.2 PUMA component

S_{PUMA} acts as a software proxy of the robot for which commands are issued to the component as they are issued to the robot. Some important *public methods* exposed by S_{PUMA} include *ConnectRobot* that connects server to slave robot, *InitializeRobot* sends a program to robot that repeatedly moves

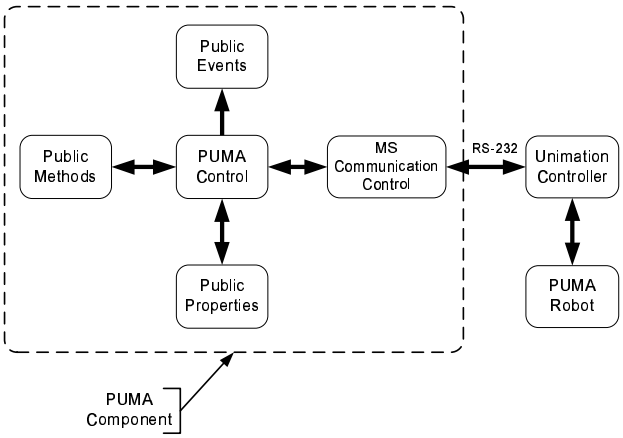

Figure 6.2: PUMA component and its interface to robot arm

the robot in an incremental fashion. Some properties are: (1) reading robot angles, (2) computing the position vector and orientation Matrix of robot hand frame, (3) setting up specification of robot tool frame, (4) setting up the communication between server and robot, etc. The events invoked by S_{PUMA} include: (1) Data received from PUMA, (2) some errors occurred with PUMA, (3) Robot moved to a new location, and (4) PUMA status changed.

6.3.3 Force sensor component

The force sensor component S_F reads the robot wrist force sensor and creates a stream of reflected force feedback directed to the master station. S_F is implemented as a separate thread, the priority of which can be adjusted during runtime to allow for the management of CPU usage. In .NET remoting technology, events can be issued from one station and received in another, i.e. automatic transfer of some data. Events do contain certain parameters, for example MouseClick(3,5) event indicates a mouse click event at position at 3 and 5. Here a new instance of S_F creates a new thread and waits until the sensing is triggered. After the reading has started, it informs the parent application (PA) of the availability of a new force packet. The PA uses an event handler at the higher level of application hierarchy. The event directly transfers the force information to the client. Similarly the component also provides *StopReading()* function to abort the force sensing thread. Sensing can be triggered again using *StartReading()*.

There are three public properties exposed by S_F. The *SensorThreadPriority* sets the priority as one out of five OS levels. The *TimerValue* sets a time interval between two successive readings. The *ThresholdValue* activates the force event only when there is noticeable change in one of the force values.

6.3.4 Decision server component

DecisionServer is a component that provides a supervisory control such as active compliance, impedance control, and assistance functions. The DecisionServer is a server abstraction layer to allow: (1) active compliance control, (2) supervisory commands like space scalability and indexing. A four-layer hierarchy including DecisionServer is shown in Figure 6.4, where physical layer refers to robot and force and lowest layer refers to the user interaction level.

6.3.5 Server side interfaces and .NET remoting

Proposed telerobotics is based on object oriented distributed application. The Decision server interface allows the client to receive the events fired by the DecisionServer instance on server side. An interface is a set carrying definitions of public methods and properties. It serves as a contract [2]

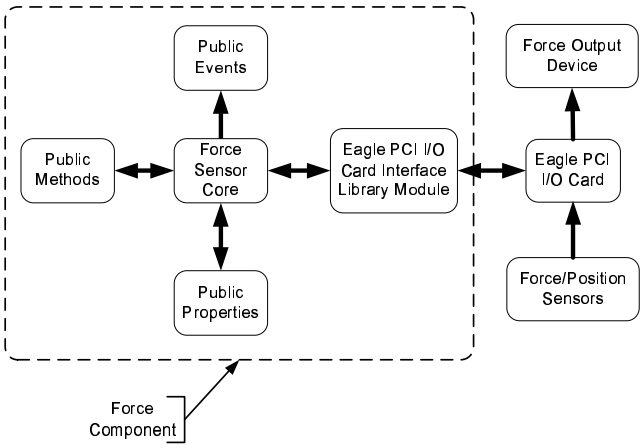

Figure 6.3: Force sensor component and its interface to sensor

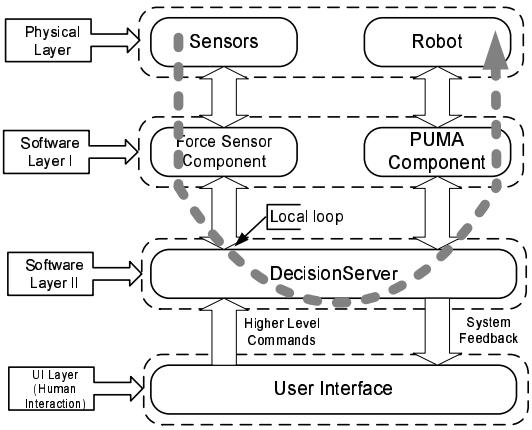

Figure 6.4: Component hierarchy on the server side

for any component that implements this interface. Any client that accesses or executes the methods of a component on the server needs an access to the server assembly or component. By giving a reference to an interface that the server component implements, we can hide the actual component or assembly from the client which improves overall system security.

To access references to both the PUMA and Force Sensor components, two interfaces are defined: *IProxyRobot* and *IForceSensor*. Further we define another interface *IDecisionServer* which inherits both the *IProxyRobot* and *IForceSensor* interfaces. Using this approach we are able to define a unified set of public members (methods, properties and events) that are required to be implemented in the form of DecisionServer component on the server side. The integrated scheme incorporating all the components on server side is shown in Figure 6.5.

.NET remoting is used to publish an instance of DecisionServer component on the network which is uniquely identified to potential clients. A client can get a reference to this instance through an *IDecisionServer* interface. .NET remoting enables accessing objects using SOAP (Simple Object Access Protocol). Any object in the distributed application can be referenced using the above scheme as if it was available on the same machine.

6.3.6 Client side components

The client contains the *IDecisionServer* interface that allows referencing the server side component through .NET remoting as shown in Figure 6.6. In addition to *IDecisionServer*, there are instances of .NET remoting and client Graphic User Interface (GUI).

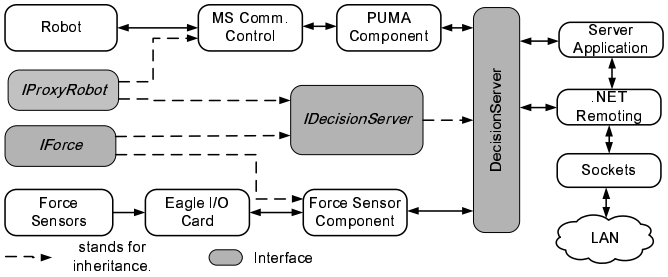

Figure 6.5: Integrated scheme - server side

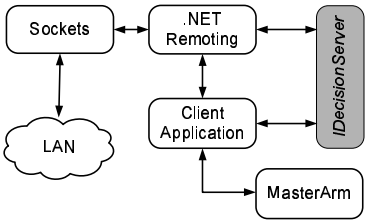

Figure 6.6: Integrated scheme - client side

6.3.7 Decision server interface

The DecisionServer is inherited from IDecisionServer and in turn from IProxyRobot and IForceSensor interfaces. .NET remoting is responsible for making socket calls to the client and we may choose either network protocol for these requests.

DecisionServer provides a mechanism for remote execution of any event handler program. The client assembly must be known to the DecisionServer which violates object oriented philosophy and reduces security. For this *Shim Classes* are used as agents to forward DecisionServer events over to the *IDecisionServer* interface. Shim classes are thin assemblies visible to both the server and the client. DecisionServer invokes the event which is received by an event handler hooked by shim classes. This event handler then calls the event handler of the client (*IDecisionServer*).

Care must be taken while receiving events from the server and writing event handlers for them because these are synchronous events which means that the thread that is invoking the event on the server side will be blocked until all the event handlers for this event are executed. To send a single TCP packet, eventually a system call is needed, which is an atomic operation, e.g. it cannot be pre-empted due to a higher priority packet. So manipulating threads during the invocation of the events may cause deadlocks in the distributed client-server environment.

6.3.8 MasterArm component

Public methods used by MasterArm are used to: (1) start and stop reading the master arm position (Inherited from *Force* component), and (2) write the given force data to the master arm in a separate thread. Now we describe some of the public properties. One property computes the change in position vector and orientation matrix after the position data ready event is fired. A property is used to find/set whether a master arm is engaged or not to control computation of arm kinematics. A get/set property is used to indicate whether to provide force feedback to the master arm or not. Other properties are also used to compute the local impedance control function. A block diagram of the *MasterArm* is given in Figure 6.7.

6.3.9 Integrated scheme of client-server components

The integrated scheme incorporating all the components on client and server side is shown in Figures 6.5 and 6.6. The DecisionServer is inherited from *IDecisionServer* and in turn from IProxyRobot

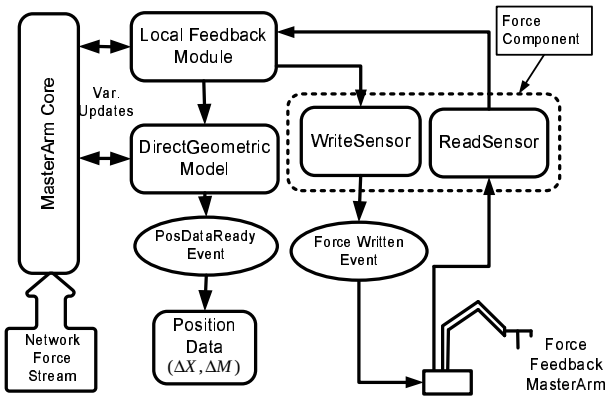

Figure 6.7: MasterArm component

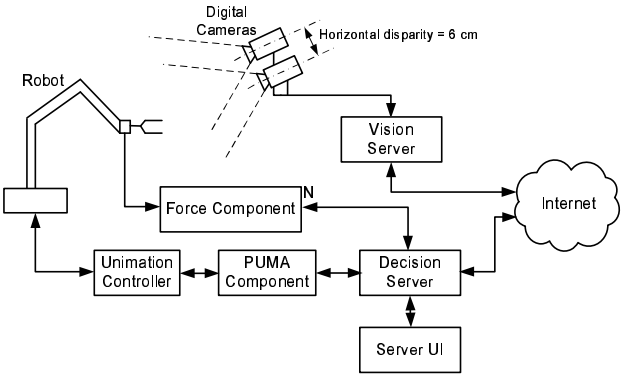

Figure 6.8: Server side of the distributed framework

and IForceSensor interfaces. In order to cause an event handler subprogram to be called (invoked) on client side for any event invoked by DecisionServer, we must provide DecisionServer, access to the client assembly. This introduces severe deployment limitations as described in Section 6.3.7. Here DecisionServer invokes the event which is received by an event handler hooked by shim classes. This event handler then calls the event handler of the client (*IDecisionServer*). This allows hiding the server and client assemblies from each other.

6.3.10 A multi-threaded distributed telerobotic system

The distributed approach leads the logic of the system be distributed in different software components. The multi-threaded aspects stem from the simultaneous threads running on the server carrying out: grabbing of stereo video data, reading force sensors, sending control signals to the robot, reading the feedback from the robot servo controller, and sending and receiving all of this information to one or more clients. Two cameras generate pictures which are sent to the client using the vision server. The operator may issue commands to the DecisionServer which in turn makes use of PUMA and Force Sensor components. The client side uses the GUI as well as master arm to issue commands to the slave arm on remote side. The vision client receives the synchronized video data using windows sockets and provides a stereo display.

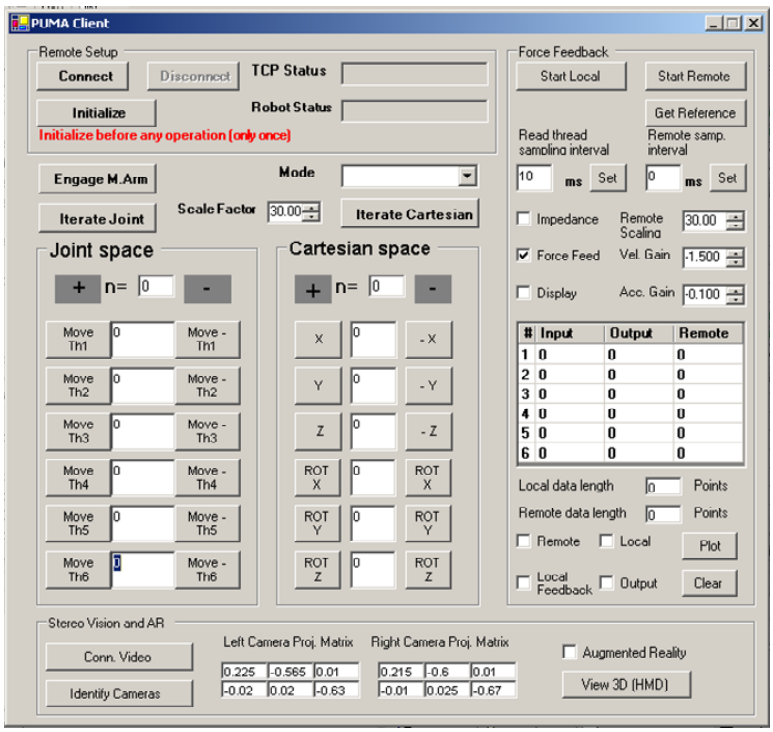

Figure 6.9: Client Side Graphic User Interface

6.4 Stereo vision for telerobotics

Stereo vision enhances operator's efficiency during tele-manipulation [41]. A client-server framework for live transfer of video data is implemented using Microsoft Visual C# and Microsoft DirectX which provides COM interfaces for graphics related functionalities such as DirectShow. The later provides an interface for capturing and playback of video data. The server captures two stereo images and upon a request from the client it sends video data to the client using windows socket. SampleGrabber (DirectShow) is used to capture video frames coming through a live video transfer as shown in Figure 6.10.

The client establishes a graphic display, establishes a connection with server, and displays the incoming pictures using a head-mounted display (HMD). A graphics device interface (GDI) of Windows graphical environment is used to (1) communicate between the application and the device driver, (2) perform the hardware-specific functions that generate output, and (3) display the received picture (see Figure 6.11). Microsoft bit block transfer API called *BitBlt() or Blit* is used to copy an image from one device context to another.

6.4.1 A distributed framework for relaying stereo vision

Synchronous windows sockets are used for video transfer at server station (see Figure 6.10). To maximize video transfer rate two thread-based schemes are used: (1) a single buffer with serialized transfer, and (2) double buffer, concurrent transfer.

In the single buffer with serialized transfer, the SampleGrabber component of DirectShow [2] uses

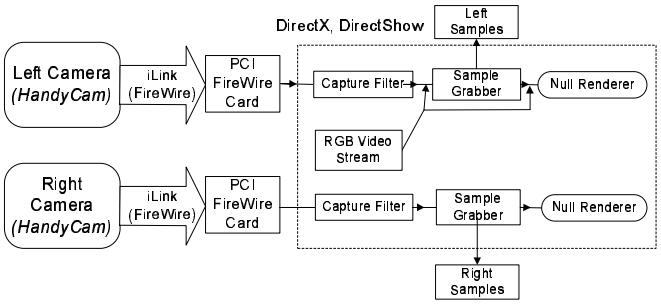

Figure 6.10: Sample grabber and camera interfacing at the server

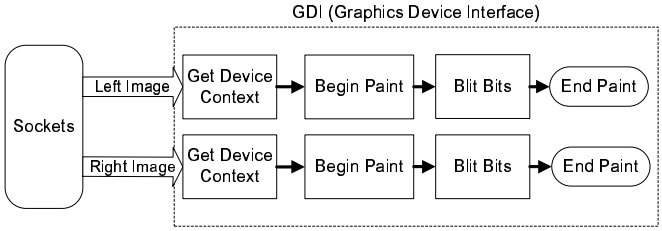

Figure 6.11: Stereo vision display on client side

a callback function to inform the completion of one video frame at server. Two thread instances of SampleGrabber running at the same time to capture the video coming from two cameras. The data is copied by SampleGrabber to some global memory buffer to be sent to the client through sockets. After hooking of callback function onto SampleGrabber, FilterGraph (DirectShow) starts the video capturing. The last step of server socket is to transfer the video data. The server waits for a request of a picture from client to start video data transfer.

On the client, the GDI is set to draw the received pictures. The server flushes the previous bitmap buffers, grab left and right images using callback functions, create a bitmap information header for these images and transfer to the client. The client gets the buffer size (TCP stream), prepares the bitmap buffer, receives the bitmap information header, copies the bitmap data from the sockets into the buffer, requests for new picture, and draws the 3D stereo picture.

In the double buffer scheme, concurrent transfer allows pipelining the execution of video capturing and live video transfer as shown in Figure 6.12. For this, two buffers are used, one for each stereo frame on the server. When a picture is received, the camera callback function is invoked which accesses a variable shared by multiple threads indicating whether the buffer was copied in the previous callback of this very camera. The camera status is used to synchronize the stereo frames for the left and right pictures. It is updated after copying the data to the buffer. If both cameras are ready, updating the buffer status enables the sending thread to transfer the content of this buffer over to the client. In case the second camera has not finished copying the picture to the buffer, buffer status is not updated.

The sending thread is responsible for receiving requests from the client. It checks the buffer status to determine the buffer, creates bitmap headers, and retrieves the buffer size. If the information has not already been sent to the client, it is sent. Otherwise, the server continues with the sending of buffer data only. The client proceeds in the same manner as with single buffer approach except that it does not receives the bitmap information header and buffer size with each stereo frame. It retains the bitmap information header and buffer size to properly display and read the required number of bytes from windows sockets.

6.4.2 An augmented reality system for telerobotics

Augmented reality consists of overlaying the graphics on real stereo images. A camera model is used to project 3D points on 2D image plane. The full perspective transformation between world and

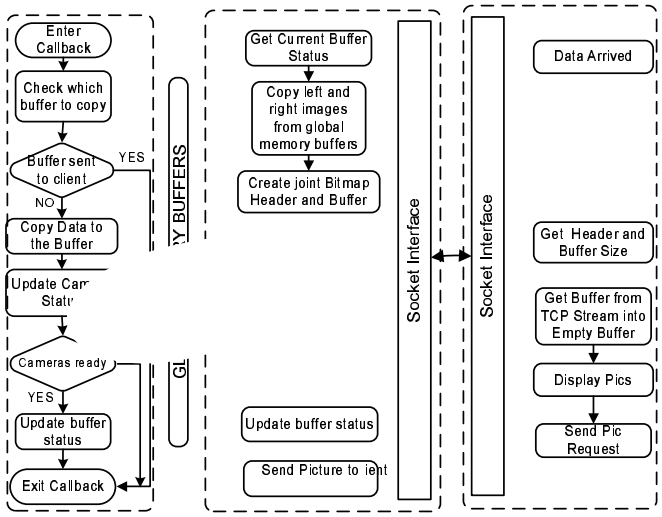

Figure 6.12: Live stereo video transfer using the optimized scheme

image coordinates is conventionally analyzed using the pinhole camera model and camera calibration proposed in [18]. When scene depth is small as compared to distance from camera to scene, the weak perspective projection is a linear approximation model [25]. In this case image plane coordinates and their pixel addresses can be related by an affine transformation. A GUI was designed to support the user carrying out camera identification by pointing to four non-coplanar reference points forming a frame of reference. This allows finding left and right camera projection matrices. The fiducial points should be 20 cm distance from frame origin and camera set at no less than 1.5 m from the scene.

Superimposing graphics using DirectX API

The image data retrieved from the *StereoSocketClient* comes in bitmap format. It can be displayed using the *DXInterface* component, and HMD. *DXInterface* is designed using *DirectX* Application Programming Interface (API) [40]. Off-screen surfaces are used to superimpose real or virtual images as well as drawing and blitting. The *BackBuffer* service of *DirectX Device* allows indexing and fast switching of primary and off-screen surfaces. The Hardware Abstraction Layer (HAL) and graphics processor control flipping the addresses of front and back buffers. Without image shearing or tearing, the next image drawn on the HMD comes from the previous back buffer. While the previous front buffer is now back buffer and is ready to be used for the coming video frame. New image is acquired from the network while the drawing of the current image is in progress. In short, the stereo video is updated on local display in a page-by-page format and not pixel-by-pixel which reduces time delays.

Component framework

In this sub-section the AR stereo vision client and server components are presented.

Client side components

Listed below are the components providing stereo vision and AR functionality on the client side:

1. The StereoSocketClient generates compatible bitmaps from the binary video and receives the live video transfer sent by the vision server developed using MFC (Microsoft Foundation Classes) client-server setup.

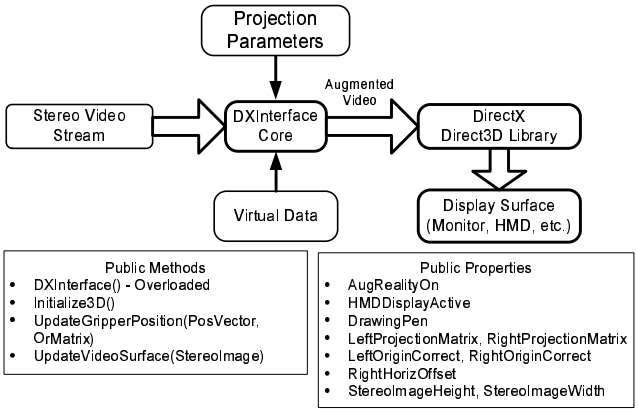

Figure 6.13: An overview of DXInterface Component

2. IdentifyCamera computes camera projection matrices using a set of four non-coplanar points as vector edges of a frame [18]. The GUI uses the images provided by StereoSocketClient in computing the left (M_l) and right (M_r) projection matrices.

3. RobotModel locates the future position of the robot gripper based on the current command. *RobotModel* acts as a local proxy of the PUMA robot.

4. DXInterface (Figure 6.13) handles (1) augmentation of real video, (2) synchronization of real and virtual data, (3) projection on video surface, and (4) page flipping.

Server side

Server side acquires and sends the stereo image data through windows network sockets. However, only the client side is responsible for the major AR functions. The server side for the stereo video client-server framework is used in the AR framework.

The complete augmented reality system

All of these components have been combined together to form a complete AR system on the client side. The system provides the AR functionality through the following steps:

1. Read operator motion using the *MasterArm* component.

2. *MasterArm* provides motion parameters to *IDecisionServer* and *RobotModel* components.

3. *IDecisionServer* executes the incremental move command on remote *DecisionServer*.

4. *RobotModel* provides the new 3D position of gripper to *DXInterface* component.

5. *DXInterface* acquires a frame of remote scene from *StereoSocketClient* as well as two projection matrices from *IdentifyCamera* at the system initialization.

6. *DXInterface* projects a ball at the gripper position in 2D image and sends both images to HMD display controller.

7. When the *IDecisionServer* receives the *OnMove* event from the remote side, the angular position of the robot is sent to the *RobotModel* to update the local model.

An architectural overview of the AR system present on the client side, is given in Figure 6.14.

It is important to update the local robot angles upon the invocation of *OnMove* event from the server side because there may be some differences between the move command arguments and the current position of robot due to mechanical and mathematical round off errors. Also it is to be noted

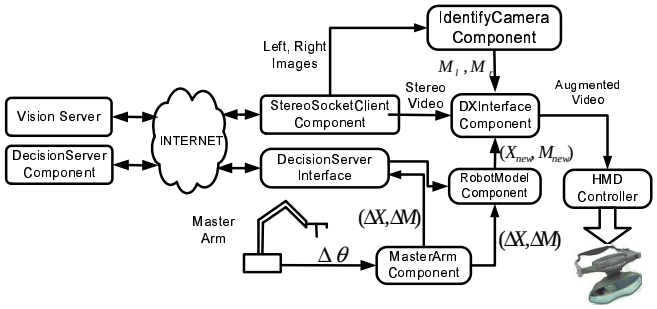

Figure 6.14: Block diagram of complete AR system on client side

that the *IDecisionServer* uses *.NET remoting* for network streaming of component data and force data while Vision Server and *StereoSocketClient* use raw windows sockets to live transfer of binary stereo video data.

The system has the ability to remember the identification of cameras and other projection related data across different runs by preserving these values to the permanent memory in a special format. So, the identification is required only when the cameras or the objects have been moved from their previous locations.

6.5 Discussion of system software architecture

Extensive performance evaluation and experimentation were carried on for the propose telerobotic system. Brief results are reported in this section.

Analysis of telerobotic delays through three campus routes was carried out while streaming of video, force, and commands. A sampling rate of 120 Hz is achieved for force feedback and 50 Hz for operator commands. Stereo video transfer operates at a rate of 17 fps. Total reference delays for force and stereo are 8 ms and 83 ms, respectively. Overall round-trip delay is 183 ms (5.5 Hz) when slave arm is operated at 10 Hz. More details can be found in [9].

The effectiveness of the framework and concurrent execution of its various computing and communicating threads has been assessed in the experimentation of the following tasks: (1) peg-in-hole insertion, (2) assembly of a water pump, (3) operating drawers, (4) pouring of water, and (5) wire-wrapping. The above experiments [3] involve the completeness, modularity and flexibility of proposed telerobotic framework when rich and heterogeneous sensory data (video, force, and command) was exchanged between client and server. A summary of results is as follows: (1) teleoperation tools are very effective and need to be developed, (2) advanced motion coordination reduces teleoperation time and operator mental effort, (3) active compliance at server station is more effective than operator reaction using force feedback. More details can be found in [5].

Even et. al [16] describes a computer aided telerobotic system as an integrated planning scheme including interactive 3D modelling, programming, and execution. The proposed telerobotic framework may provide the above system a pervasive network connectivity and mobility as well as a structured software framework for implementing real-time interactions.

In [7] a client-server framework is designed using VB 6.0 and custom protocol with TCP ActiveX controls. This scheme exposes TCP read/write operations to application, custom protocol, TCP ActiveX control, and Windows Sockets etc. While in a distributed setup, the components directly communicate with each other through windows sockets using .NET remoting. This difference is achieved by using the distributed component based approach in place of TCP based custom protocols.

An object oriented distributed application (OODA) could also use JAVA that provides RMI as remote calling function, graphics services, and 2D and 3D support. Hu et. al.[29] proposed JAVA for network interfacing and video as well as the use of C++ for the robot controller for Internet-based Telerobotic Systems. Ho[22] used Java-based framework for frame grabbing as compared to the use of DirectShow in our scheme. Compared to [24], .NET framework is directly used for all

GUI development as well as the core system components making it a unified solution. This avoids the use of middleware services like MS VM within the framework. JAVA and CORBA are intended to be cross-platform environments thus requiring lot of JIT (just-in-time) compilation and virtual machines to interpret code on different operating systems. In addition they provide no support for hardware-accelerated graphics APIs that are critical for live video visualization on PCs.

Compared to [26], the behavior in proposed telerobotic framework is represented by the local position, force, and active compliance loops and the planer is being the human operator and AR system. Compared to [15], in proposed framework the coarse correction made by the remote teleoperator adds to the competence of fine active compliance loop during the period of contact with the environment. Compared to [38], in proposed framework there are three servers and three clients at any given time. Object interconnection is implicitly done using .NET remoting.

We proposed an object-oriented distributed component framework and .NET remoting for (1) remote procedure call, (2) hiding details of network interface, (3) an automatic notification and data messaging mechanism between client and server. .NET does not require components registration thus breaking the interdependency in the development phase. To develop off-the-shelf components and programs we used Windows 2000 because of support provided for hardware accelerated graphics, .Net remoting, and thread scheduling and the priority needed for multi-threaded execution. .NET may be used for off-the-shelf real-time applications as it casts off every possible overhead. .NET has embedded type signatures which allow component debugging across different languages, a missing feature in JAVA and CORBA. Process variables like real-time sensor data and robot-states are relayed to the client-side using implicit inter-component communication.

In conclusion, a real-time *Distributed Component Telerobotic Framework* has been described using object-oriented distributed programming paradigm with Visual C #, .NET remoting, DirectX, and TCP sockets. .NET remoting is used to automatically handle the network resources and data transfer while isolating the components from network protocol issues. This approach frees the programmer from defining custom protocols for client-server interaction. It also provides flexible deployment environment without pre-registration of components on host machine which is an advantage over DCOM. The framework is implemented as a modular multi-threaded system for live transfer of stereo video, transfer of master-slave commands, and streaming of force feedback. The functionalities of master and slave arms are independently designed. A multi-threaded execution with DirectX graphical tools has been used to promote concurrency in grabbing, transmitting, receiving, processing, and displaying image data using HMD technology. The client station provides components that support AR tools like camera model identification, and graphical slave arm components. The proposed framework is useful to enhance man-machine interactions with 3D perception and AR, and to serve as an integrated environment to develop telerobotics.

6.6 Performance of networked telerobotics

A real-time telerobotic system consisting of client station (operator) and server station (slave arm) interconnected by a computer network has been implemented using a distributed component framework. To minimize overall delays a multi-threaded execution is proposed for pipelining of information processing and real-time transmission. Thread engineering allowed pipelining stereo grabbing and live transfer of stereo video data. Different scenarios are statistically analyzed to relate the effect of thread manipulation to overall time delays. Analysis of telerobotic delays through three campus routes with different network loads is presented. A sampling rate of 120 Hz is achieved for force feedback and 50 Hz for operator commands when network load is below 80%. Copying stereo images from cameras to memory is done in 24 ms. Stereo video transfer operates at a rate of 17 fps. Total reference delays for force and stereo are 8 ms and 83 ms, respectively. The environment interaction delay is 183 ms (5.5 Hz) when slave arm is operated at 10 Hz. However, short instantaneous traffic irregularities may cause deviation and scattering from above reference rates. A qualitative analysis of contact forces arising in the contact of a rigid body, a spring, and a human muscle tissue is presented.

The performance of networked teleoperation systems is based on timely streaming of highly-demanding dynamic media to interface human operator to the actuators and sensors of a remote robot. A central problem is communication delays [30]: satellite links, for example, often have round-trip delays that last from a fraction of a second to several seconds. In telesurgery [28] delays can greatly slow down task execution, as the surgeon must pace the procedure to wait to see the effects of commanded motions. Delays in teleoperation with force feedback [46, 45] can cause instability of the robot control system, although various techniques can help to minimize this problem.

Real-time network and protocol transmission delays, jitter [41], and processing times need to be reduced to ensure guaranteed quality of service for robot commands, stereo vision, and force feedback. In a computer network, the communication delays and traffic capacity vary with flow direction and irregularly change with traffic conditions.

Teleoperation [42] on packet switched LAN indicates that when packet size is increased from 64 to 1024 bytes, the delay is increased from 5.6 ms to 13.4 ms due to computational overheads. LAN performs well even in the presence of traffic caused by other users until the total network congestion where delays become unpredictable. The operator performance is quite insensitive to a fairly small data loss. Transmission delay causes a decrease in operator performance almost linearly. Jitter produces a disturbance in velocity.

In Internet telerobotics using TCP/IP sockets [10] and VxWorks real-time multitasking system, reliable connectivity can be established but with delay jitter in the arrival rate of originally synchronous packets. Packet inter-arrival delay varies from a few ms to a few seconds and average delay for a small packet ranges from 50 ms to 100 ms over the US. Asynchronous packets do not preserve their chronological order. TCP/IP can be reasonably used for packet with 256 bytes with a 10Hz sampling rate.

In [20] a telerobot is implemented using TCP/ATM in which two LANs are connected to an ATM backbone. Specification of Quality-of-Service (QoS) includes application timing, criticality, clock synchronization, and reliability. The use of constant bit rate in ATM allows a tightly constrained transmission delay which is suitable for real-time applications. ATM is used to guarantee time service frequency at the computing nodes (QoS brokerage). The sampling intervals reported are 0.4, 0.3, and 0.2 s for TCP/IP, raw ATM, and TCP/IP over ATM, respectively.

A scattering transformation is proposed to overcome the problem of jitter in bilateral feedback systems [34]. To keep the time delay constant, data is sampled at constant rate and transmitted along with its sampling time t_s. The maximum delay T_m is estimated. When data is received at t, if its delay $t - t_s$ is below T_m then the data is released only when its delay becomes T_m which provide a constant delay.

To improve real-time performance of telerobotics a multi-threaded execution is proposed for live transfer of force, command, and stereo data. Different software optimizations are statistically analyzed. Delays are evaluated using three campus routes with different network loads. Thread engineering is use for minimizing overall transfer time of stereo video data. Specification of video and force packet times are presented and analyzed. Effects of non-deterministic surge in network load are presented. Using the above real-time telerobotic system, contact between the slave arm tip and environment is presented using kinesthetic force display.

The organization of this Section is as follows. First the network and system specification are presented. Second the video and force delays are statistically presented and evaluated. Thired, a qualitative evaluation of contact forces is presented. Fourth, a discussion and a comparison of achieved performance to other contributions is presented together with a conclusion.

6.6.1 Network and system specification

In this section the specification and configuration of the network, software, and computers used is presented prior to addressing the performance analysis of the proposed telerobotic framework. Detailed design aspects of proposed telerobotic system can be found in [8].

The client and server are run on two PCs having 2-GHz Intel P4 processors with 1GB DRAM

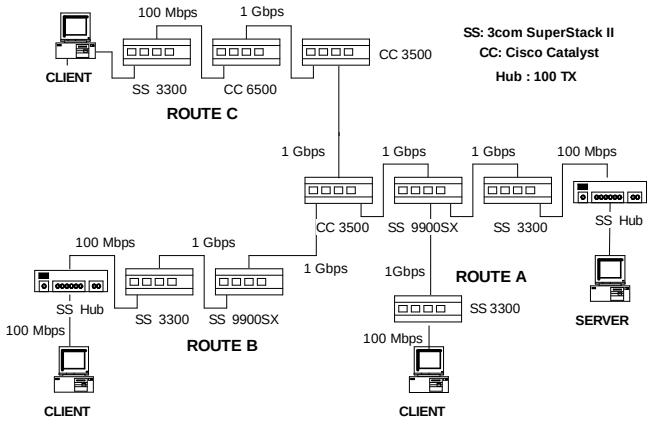

Figure 6.15: Campus network routes used between a fixed server and moved client

and 512 KB cache memory. Control of master and slave arms is done using Eagle PCI 30FG data acquisition cards. Each of client and server PCs is attached to a campus network by using a 100 Mbps NIC card (3com EtherLink XL PCI). The server PC is interfaced to two Sony Handycam digital cameras using a 400 mbps FireWire PCI (IEEE-1394) card. The client PC uses an NVIDIA GetForce4 Ti4600 as display adaptor to interface with an SVGA resolution Cy-visor DH-4400VP 3D head-mounted display.

Both client and server PCs run under MS Window 2000. The vision server software uses MS Visual C++ with .NET framework 1.1 under Microsoft development environment 2003. The imaging device used is Microsoft DV camera and VCR. The PUMA server is implemented using MS Visual C# with the above .NET framework.

Network delays are studied using three campus networks denoted by routes A, B, and C as shown on Figure 6.15. The server station is fixed in one location and the client station is moved to each of the three terminals in routes A, B, and C. The connectivity between the client and server stations is defined as follows. Refer to above figure for networking specification and component abbreviations. In route A, each packet travels across three L2/L3 switches (SS 3300 and SS 9900 SX) and a 100 Mbps hub (SS 100 TX). In route B, each packet travels across two L2/L3 switches using 100 Mbps input links (SS 3300), one L3 backbone switch (CC 3500) that uses 1 Gbps link at input and output, and two L2/L3 switches (SS 9900SX) using 1 Gbps links and two 100 Mbps hubs. In route C, each packet travels across three L3 backbone switches (two CC 3500 and one CC 6500) working as routers using 1Gbps links, three L2/L3 switches (SS 3300 and SS 9900SX) and a 100 Mbps hub.

In the next section analysis of telerobotic delays is presented in above three campus networks.

6.6.2 Performance evaluation

Evaluation is carried out at the following levels: (1) streaming force in presence of video, and (2) thread engineering and delays in live transfer of stereo video.

Streaming force in presence of video

The performance of streaming force feedback between the server and client stations mainly depends on (1) total delay and (2) inter-arrival rate.

Denote by NU the percentage of network utilization. We first present the performance obtained for route A and later we repeat the experiments for route B and C. The above network interconnects computing systems for faculty, administration, and PC labs and provide access to Internet. The reference NU during the running of proposed telerobotic system with the above utilization is measured as about 10% for 1 Gbps links and 80% for 100 Mbps links of routes A, B, and C.

The force feedback thread is responsible of reading the force sensor, computing the force and moments at the tool frame, and transmitting the force data to the client station. We expose the

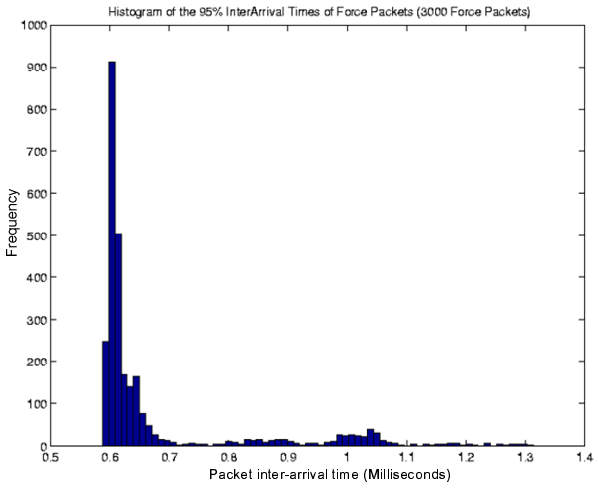

Figure 6.16: Distribution of inter-arrival times of force packets

running of the force thread to factors that may affect its performance such as concurrent CPU thread execution, network communication, and change in the route between the client and the server. Each force data packet is 48 bytes. Each video picture is 288×360 pixels and each pixel is 3 bytes. One picture is 0.3 MB. A stereo frame consists of 2 pictures or a data volume of 0.6 MB which requires a bandwidth of 5 Mbps/Frame for its network relaying.

The server throughput on network in the case of streaming only force packets is about 1 KHz as shown on Figure 6.16. Specifically for 90% of the cases the inter-arrival times of force data packets is below 1 ms. The average time to copy one stereo frame from the SampleGrabber to the DRAM is 24 ms. However, the video copying time is increased to 60.5 ms if we enable a thread to only read force information without network transfer. If the network transfer of force packets is enabled, the video copying time increases to 33.5 ms because when force packets are transferred on the network the internal processor resources are exclusively used by the stereo copying thread.

The transfer of force and video information leads to a force packet rate of 250 Hz due to sharing of CPU and network resources among the force and video threads. In 90% of the cases the inter-arrival time is below 4 ms. Although the video thread was continuously active the video transfer was ON and OFF in this case. For route A, Figure 6.17 shows the distribution of force packet inter-arrival times during active video transfer instance (ON and OFF) which corresponds to a reference time of 8 ms (mainly from 1 to 8 ms) but may occasionally increase up to 21 ms. Therefore, the reference server rate of force packets is 120 Hz and may occasionally drop to 47 Hz. Testing shows that processing time of force packets (reading, computing, and packing) on the server station is negligible compared to its transmission time for all studied routes. We conclude that a reference for the total force packet time is 8 ms. Inspection of the inter-arrival times of force packets in presence of active video thread (ON) shows that the worst case corresponds to arrival times in a range of 6 ms to 21 ms.

In the case of multi-streaming of force, command, and video packets the average inter-arrival time of force packets is 1.1 ms and all the population remains under 8 ms. The force thread is having more opportunity due to the presence of a command thread. Some peaks appear on the force packet arrival times which correspond to periods of active video transfer. The delays caused by the implicit software and network transfer overheads in using .NET remoting and Shim assembly for streaming of force data are quite comparable to network protocol delays for the same routes using traditional TCP socket for data transfer.

The use of .NET technology for streaming of force packets provides nearly the same inter-arrival delay for routes A, B, or C which is also comparable to the delay for one single hub using TCP sockets. The Gbps switches over all three routes normally operate with low NU values. Some subnet may occasionally reach the congestion level which is manifested by an NU exceeding 80% on at

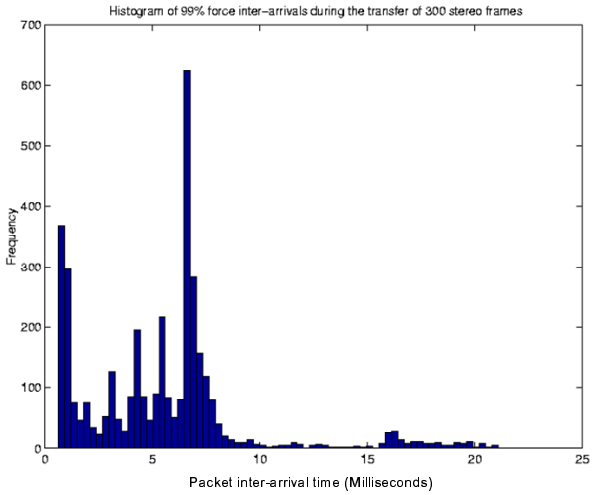

Figure 6.17: Distribution of arrival times of force packets during video transfer

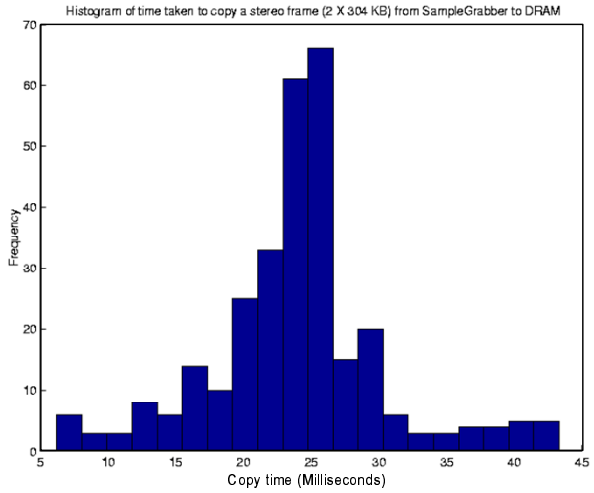

Figure 6.18: Distribution of copy times from SampleGrabber to main memory

least one 100 Mbps link of a given route. In this case the distribution of inter-arrival times of force packets, in the presence of live video transfer, becomes scattered and unpredictable in a range going from 8 ms to 30 ms.

Delays in live transfer of stereo vision

In this sub-section we study: (1) the delays in copying the video data from cameras to computer main memory, (2) performance of thread engineering for live video transfer in route A, and (3) video performance in routes B and C.

Copying from SampleGrabber to main memory

To measure the copying time on the server from SampleGrabber to main memory we consider two cases: (1) a single stereo thread, and (2) a stereo copying thread with a force thread.

In the case of a single stereo thread, the distribution of inter-arrival times of 300 video frames is shown in Figure 6.18. The mean value of 24 ms for which the 95% confidence interval falls below 25 ms.

In the case of a stereo copying thread from SampleGrabber to main memory with a force thread, the force is read as fast as possible without data transfer over the network. The mean copying time

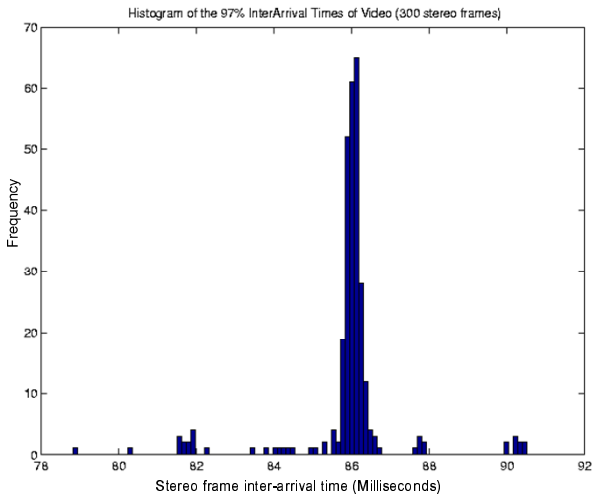

Figure 6.19: Distribution of stereo frames using the serialized transfer approach

increased from 24 ms to 60.5 ms and 90% of the data lies between 8 and 150 ms. CPU time sharing (active force thread) has significant effect on stereo copying on the server. This is a useful feedback to the need for real-time operating systems and parallel I/O streaming.

In the case of stereo copying thread with force thread reading and transferring data over the network the mean stereo copying times decreased to 33.46 ms. The improvement over a copying time of 60.5 ms is due to the release of CPU resources to the video copying thread due to the use of blocking socket. This indicates that the multi-threaded execution of three concurrent threads is left to the best effort of Windows OS.

Live transfer of stereo video for route A

Performance of live video transfer is evaluated as follows: (1) a single buffer with serialized transfer, and (2) double buffer, concurrent transfer. Synchronous windows sockets are used for the client server video interfaces.

A single buffer is used in the serialized video transfer on server. The sending thread waits for the two SampleGrabbers to write stereo frame data to global buffer in order to send it to client with disabled display. Figure 6.19 shows the distribution of inter-arrival times of 300 stereo frames which is a Gaussian distribution with a mean of 86.5 ms or 11.6 fps. The distribution of inter-arrival times of video packets in the presence of force thread is shown on Figure 6.20. The above two distributions show the effects of resource and network conflicts caused by concurrent threads. Although the average arrival times are nearly the same the distribution is scattered due to indeterminism in the CPU execution and pre-emption of both video and force threads.

The double buffer, concurrent transfer scheme is evaluated using a single buffer configuration on the server. Figure 6.21 shows the distribution of inter-arrival times of 50,000 stereo frames transferred between client and server with disabled display. Statistically this is a Gumbel distribution with a mean value of 59 ms and 90% of the data lying between 56 and 65 ms. A transfer rate of 17 fps is achieved. The maximum delay observed is 1299 ms which obviously is coming from network congestion and the minimum value is 53.5 ms. Time was saved in activating the SampleGrabbers and receiving the buffer ready notification. This proves that pipelining of video copying and transfer over the network is superior to traditional sequential processing demonstrated in serialized transfer.

Testing shows that a frame rate greater than 10 fps gives good viewing and refresh rate of 85 hertz eliminates any flickering. Some simple manipulation experiments, to move objects by looking at 3D scene on the computer screen wearing shuttering glasses and HMDs showed good depth perception of the viewer.

In summary, the lowest transmission delay for live video data transfer is observed in network

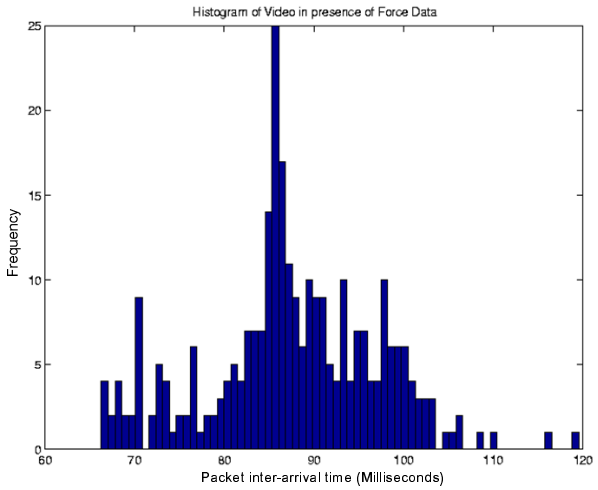

Figure 6.20: Distribution of video packets with active force for serialized transfer

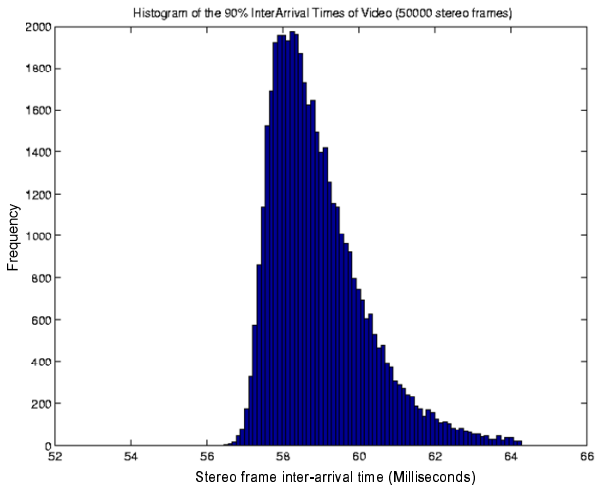

Figure 6.21: Distribution of inter-arrival times of stereo frames for route A

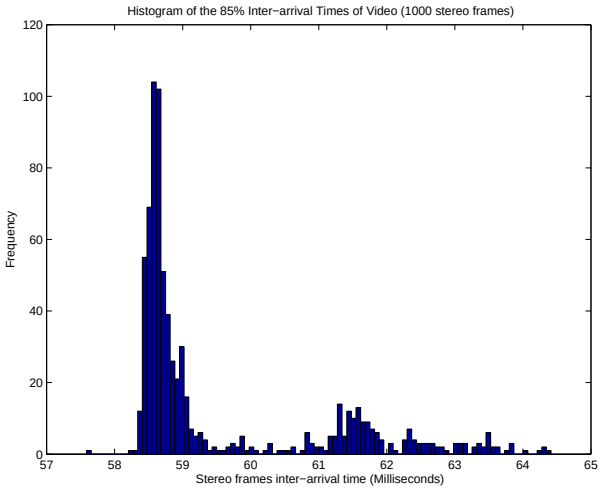

Figure 6.22: Distribution and scattering of inter-arrival times of stereo frames in route B in presence of force packet streaming

A. While copying a stereo video frame to main memory takes 24 ms network transfer of live video takes 59 ms, e.g. an arrival rate of stereo frames of 17 fps. In other words, the total time of stereo frames is 83 ms between server and client from reading cameras to HMD display. The average delays caused by route A are similar to that obtained when both client and server are interconnected to one single LAN segment (Hub) in route A. Testing shows that total operator motion computation and command time is negligible at the client. However, the fastest motion on the available PUMA system is reasonably at the 100 ms level given its mechanical constraints and serial interface. Therefore, the total round trip delays from force sensing and video capturing to executing operator command in response to above feedback is about 183 ms which corresponds to an interaction frequency of 5.5 Hz. The Gbps switches and links operate at a lower rate than their real capacity. The above parameters can be considered as system references when NU is below the 80% reference level for each used 100 Mbps link on route A. The lack of accurate synchronization between force and stereo frames has not shown to be critical in experimenting the system.

Occasionally an increase of NU above the 80% level for one or more of the 100 Mbps links (congestion) of route A leads the distribution of video and force inter-arrival times to be characterized as: (1) the dominant part of the distribution is still at the above reference values, and (2) some scattered distribution starts to appear in the region above the reference delays. At the above congestion level the width of the scattered region extends to 30% of the reference delays for the video. For example the scattering of the video distribution may extend to the range of 59 ms to 80 ms.

Live transfer of stereo video for routes B and C

In network B, it is noticed that when NU is below the 80% level for all 100 Mbps links the stereo and force frames preserve the same distribution pattern as in network A, but the whole distribution is slightly shifted with an increase in the average inter-arrival times to 60 ms for the stereo data. The increase in the reference delay of 1 ms is due to buffering of video data when hopping from one switch to another. The distribution of video packets for route B is quite similar to that of route A shown on Figure 6.21.

Route C includes three major gigabit L3 switches, three L2/L3 switches (two 100 Mbps and one 1 Gbps), and one hub. Campus route C is used as an exit route for accessing the Internet which means that traffic burstiness (irregularity) is the highest for this route. Testing shows that route C has comparable average NU to that of other studied routes but with the highest degree of traffic irregularity. Here more than 90% of the population of stereo frames deviate by no more than ± 0.5 ms from the reference delays of route A. Similarly, when NU is below the 80% level for all used 100

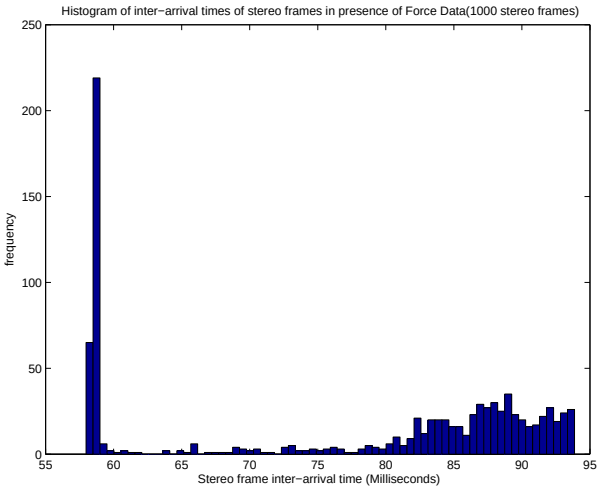

Figure 6.23: Distribution and scattering of inter-arrival times of stereo frames in route C in presence of force packet streaming

Mbps links of route C the dominant part (90%) of the distribution remains in the range of 58-60 ms (stereo) and some scattering appears in the region 60-80 ms. Rare bursty traffic during which NU rapidly fluctuates between 10% and 20%, even for short time, causes the distribution to become rather uniform and unpredictable and may extend to the range of 60-120 ms for stereo frames. The start of distribution scattering of inter-arrival times of video packets for routes B and C is shown on Figures 6.22 and 6.23, respectively.

A flow-control strategy for live stereo vision is to minimize overall stereo delays by considering a variety of cases between the following two extremes: (1) sending large packets without compression to save processing time, or (2) sending compressed video to save overall data transmission time. In the proposed framework uncompressed stereo video transfer showed to have less overall delays as compared to compressed video transfer over the studied routes. For telerobotics, one strategy is to develop a task-aware compression method for compressing background data and transfer of uncompressed, small volume, higher resolution, region-focused, video data that is essential for the current task. The operator may specify a relevant tool region to be dynamically tracked and transmitted with higher resolution and refreshing rate. A tracking algorithm detects motion in the relevant region and guides a selective compression algorithm. Thus by controlling the size of the uncompressed region and its resolution the user may set up a variety of scenarios between the above two extremes.

Nowadays network routers, buffers, switches, and links do not incur noticeable delays. Here telerobotic real-time packets hoping from one switch to another do generally accumulate a small amount of delay regardless of used route. Packets with smaller payload are less sensitive to surge in network load than packets with larger payloads. The problem of unpredictable arrival times of real-time packets is concerned with the instantaneous traffic condition rather than with delays accumulated in network switching and buffering. The non-deterministic nature of network load even for short period of time may cause severe degradation over the reference delays. Although this is the exception, the unpredictable inter-arrival times of real-time packets may significantly degrades the quality of teleoperation. In this case, there is need for a resilient telerobotic flow control to ensure smooth performance degradation under severe load conditions. One approach is a flow-control that activates remote emergency agents, at the slave site, to ensure task safety and continuity under excessive delays. At the client station, a virtual environment may supply the operator station with kinesthetic and visual feedback to provide interaction continuity based on task locality, environment model, and history information.

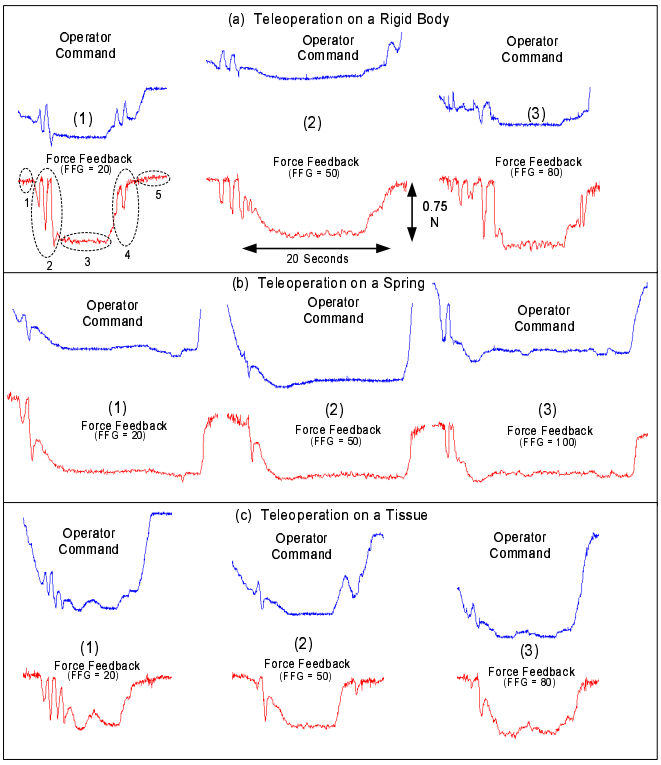

Figure 6.24: Operator commands and force feedback during contact with a rigid body (a), a spring (b), and a tissue (c)

6.6.3 Qualitative evaluation of force feedback

Force feedback [46, 19, 35] is widely known to enhance performance in telerobotics. In telesurgery, providing force feedback display to the surgeon proved to (1) reduce tissue trauma and (2) increase operation safety [14]. The proposed telerobotic system is used to provide Cartesian bilateral coupling between a PUMA 560 slave arm and a locally designed master arm having 6 dof position and 3 dof force display

Telerobotics focuses on remote interaction with the environment [28, 43]. Analysis of contact forces as well as modelling issues based on motion damping and impact velocity is presented in [39, 32]. Displaying haptic and force feedback allows the operator to navigate in a space constrained by physical effects like environment contact or others. This proved to be effective when the force feedback is applied at natural human inputs (hand) leading to the most ergonomic cognitive reaction of operator.

Force-reflecting teleoperation was proposed earlier in [44]. To quantify the transparency of proposed telerobotic system we study the slave arm tool transition from free space to contact state. The objective is to minimize forces arising during contact between robot tool and environment.

The operator moves the master arm along a vertical direction which causes similar motion to the slave arm tool. A real-time force data stream is transmitted from a wrist force sensor and transmitted from server to client station (master arm) where it is displayed on the operator hand using the master arm. Details about motion and force mapping can be found in [8].

Figure 6.24 shows the force interaction during contact between the tool and (1) a rigid body (case a), (2) a spring (case b), and (3) a human muscle tissue (case c). Following the contact the operator was asked to maintain a known constant force on the target for a short time. Each of the above three cases is further sub-divided into three experiments (from 1 to 3) each corresponds to a different setting of the force feedback gain (FFG) which is used to scale up and down the force feedback measured at the slave arm wrist and displayed on the operator hand. The force feedback received by the client is plotted before being scaled by the FFG coefficient. FFG is intended to

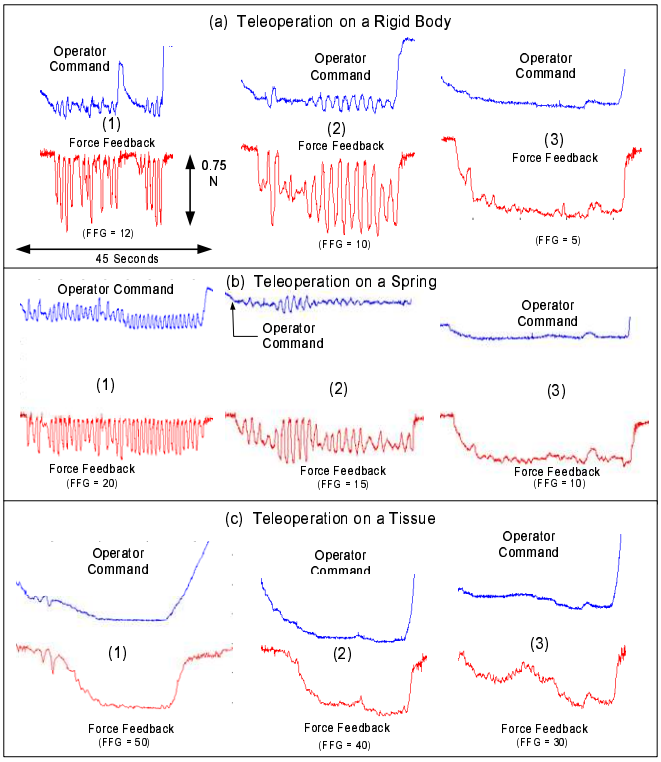

Figure 6.25: Extended command-force interactions of pre-contact for a rigid body (a), a spring (b), and a tissue (c)

adjust the displayed force to a proper sensitivity level for the operator in connection with overall system stability.

In each of the nine instances of Figure 6.24 the operator command and associated force feedback are plotted in the upper and lower parts, respectively. During 20 s the operator is to bring the slave arm tool into contact with the target and to maintain a constant force of 0.75 N.

Each contact operation has 5 phases which are (1) contact-free, (2) pre-contact, (3) contact, (4) pre-release, and (5) release. These are shown on Figure 6.24-(a)-1. The operator receives no force feedback when the tool is in free space. The pre-contact phase starts when the tool hits the target. We note that in both pre-contact and pre-release phases the teleoperation system is subject to vibrations which are displayed to the operator using scaling factor FFG. The vibration frequency depends on: (1) stiffness of the target, (2) value of FFG, and (3) a system interaction time of 183 ms or round-trip delay (Section 6.6.2). At the slave arm station, the rate of sensing-to-reaction or operator tool interaction is 5.5 Hz. Stiff targets produce prompt bouncing contact forces and therefore produce higher vibration frequency. The vibrations for the rigid object are greater and faster than those of the spring or the tissue.

Similar effects are also observed for higher values of FFG as shown for each instance of figure parts (2) and (3). In the master arm both (1) the operator and (2) master arm motor can generate forces that control the dynamic of corresponding linkage (motor, wires, pulleys, and operator). The motor excitation is being the reflected force feedback.

Figure 6.25 shows an extended pre-contact periods for each of the above cases and material. The reason for the vibration is that when the operator is starting the pre-contact phase the first contact leads to (1) a force feedback applied to master arm motor, (2) transmitting motor torques to operator through the dynamic of the linkage, and (3) producing a force bouncing (as the force feeling) from the operator hand back to the slave arm. This process continues to transmit contact forces from the scene and return a bouncing force from the operator until the elastic system between the target and operator hand is closed up by the operator engaging the slave closer and closer to target which amortize the above bouncing. Note that the release phase is similar to the pre-contact

phase. A high FFG gain may drive the telerobot out of control as shown in Figure 6.25-(a-1), (a-2), (b-1), and (b-2). Stable contact for the rigid and spring objects requires the use of lower FFG gains.

Note the difficulty caused by the visco-elastic nature of the tissue when the operator attempted to maintain a constant force contact. The tissue shape deformations causes instabilities even in the contact state as shown on Figure 6.24-(c-1), -(c-3), and Figure 6.25-(c-2), and -(c-3)). The tissue is more subtle because its visco-elastic deformation may cause pre-contact phases to occur in the middle of a contact phase. Here the operator felt the reaction and attempted to re-engage the contact state. There are other important reasons for instability in Internet teleoperation which are due to transmission delays [19].

Generally the contact phase is characterized by a good fidelity of force feedback. The master arm can display a force of 20 N and the values of FFG used are in the range of 1 to 100. A stable post-contact force factor of 1:100 is used. The operator succeeds in maintaining the target force for the desired duration. The FFG can be set up by the operator to reduce the possibility of errors as well as to increase operation safety in telesurgery. However, excessive FFG values lead to unstable operations. It was observed that at high FFG the elasticity feature of the spring was transmitted into physical constraints on the operator motion. Contact phases are all quite stable regardless of target stiffness when moderate FFG is used (1:20) except for tissue. For example pressing on the spring induced an opposing force on the operator hand giving the operator the feeling of a quasi spring. Teleoperation with weight exposes the operator to gravity effects if needed.

6.6.4 Discussion of networked telerobotics

A telerobotic client-server framework [4, 7] is proposed for VB 6.0 and TCP ActiveX platforms. Read/write operations using TCP based custom protocols take 20-40 ms because of the software layers involved such as application, custom protocol, TCP ActiveX control, and Windows sockets etc. Transmitting a command signal of 48 bytes between the client and server stations takes 55 ms. In the proposed framework a similar force packet takes about 8 ms in the presence of both video grabbing and video transfer threads.

In a typical scenario when both client and server use .NET based components with TCP channels, optimized data transfer is reported [40]. TCP Channel uses a default binary formatter which serializes the data in binary form and uses raw sockets to transmit data across the network. This method is ideal if the objects are only deployed in a closed environment.

Internet-based telerobotic System [29] can be implemented using JAVA for network interfacing and video and C++ for the design of the robot controller. In a LAN setup, a transfer rate of 9-12 fps with time delays less than 200 ms for a single image of size 200×150 pixels is reported. Video images are compressed using JPEG technique. The Java-based [22] frame grabbing software takes one second for an image to move from camera to main-memory as compared to a mean value of 24 ms obtained by proposed multi-threaded approach using DirectShow. The reported video transfer rate is 1 frame every 3 seconds for a single image of 16 bit color depth over the Internet.

In the proposed approach, a multi-threaded execution is proposed to allow pipelining of (1) processing and (2) network transfer times. In comparison to the above results, the proposed stereo video client-server system transfers on a campus network uncompressed stereo frames of 288×360 pixels at a reference rate of 17 fps and a total time of 83 ms. Other achieved sampling rates are 120 Hz for force feedback and 50 Hz for operator commands. This approach allows understanding some of the limiting factors in telerobotics and to develop engineering solutions for direct teleoperation tasks involving contact forces.

In conclusion, a telerobotic system transmitting live motion commands, force feedback, and stereo video has been evaluated on three campus routes with different load conditions. To minimize real-time delays a multi-threaded programming has been used to restructure sequential processing into concurrent threads that are executed in a pipelined fashion to parallelize the CPU processing with network transmission. Pipelining of grabbing stereo data with live transfer over the network allowed (1) copying a stereo frame from cameras to memory in 24 ms, and (2) live stereo video

transfer at a rate of 17 fps. When network load is below 80%, the reference sampling rates for force feedback and operator command are 120 Hz and 50 Hz, respectively. The total delays for force and stereo are 8 ms and 83 ms, respectively. The slave arm is operated at a 10 Hz rate which leads to a round-trip delay of 183 ms or 5.5 Hz. Inherent network routers do incur negligible delays to above reference rates. However, short traffic burstiness may cause noticeable scattering in the above reference rates. As future direction, we proposed a task-aware video compression guided by a vision algorithm which tracks a relevant region that is transmitted with higher resolution and refreshing rate than background data. A resilient flow-control is also proposed to activate emergency agents at slave station to ensure task continuity and safety during periods of excessive delays.

6.7 Telerobotic experimentation

A multi-threaded distributed telerobotic system was previously described. It consists of a client station (operator) and a server station (slave arm) interconnected by a computer network. The system is evaluated using a set of teleoperated experiments which are (1) peg-in-hole insertion, (2) assembly of a small water pump, (3) operating drawers, (4) pouring of water, and (5) wire-wrapping. Direct teleoperation is evaluated using following schemes: (1) stereo vision, (2) vision and force feedback, and (3) vision with active compliance. Space indexing and scalability tools are also used. Mapping of operator hand motion and force feedback to a convenient tool point reduces operator mental load and task time due to highly-coordinated motion and ease of understanding of force feedback. Operator is logically manipulating the remote tool both in motion and force and feels the exerted forces as they were exerted in the hand. With active compliance all tasks are done in the least task times and with the least contact force. Stereo vision may alone be used but with large peak forces and extended task time. Force feedback has nearly equal task time as compared to active compliance but with a noticeable increase in contact forces. Force feedback and active compliance are critical tools for extending human eye-hand motion coordination and dexterity to remote work in hazardous, hostile, inaccessible, and small-scale environments.

Telerobotics aims at extending human natural eye-hand motion coordination over an arbitrary distance and an arbitrary scale. The objective is to replicate manipulative skills and dexterity to a remote work place. Human psychomotor skills have evolved over million of years. The design of effective man-machine interfacing and the transmission delays are two major limiting factors.

A 3D virtual reality model [17, 16] of the environment is used to develop model-based assistances and mixed control modes in repeatedly performing a sequence of short modelling, programming, and execution. A virtual arm is teleoperated, accessibility is checked, valid paths control the slave arm, and feedback from slave arm is used to controls the virtual arm. The system is used in unfastening 12 nuts of a tap cover, lifting up a cover using gantry crane, inspecting the tap, and lifting down the cover and fasten it again.

An event-driven virtual reality (VR) [11] is used to model the environment to ease the task of programming, planning, and teleoperating a remote robot. Once the VR assemblies are set up placed, the real links update their recorded trajectory. This approach is useful to resolve conflicts among multiple robots while reducing communication bandwidth.

In [19] a sensor-based motion-planning is proposed for teleoperation in deep space. Bilateral control of a graphic slave arm operating on a 3D graphic environment is used to select an approximate sequence of fine motions. The operator is provided with graphic animation using kinematics, dynamics, friction, and impact forces used in a closed loop control provides the operator the feeling of repulsive forces. A 3D collision prevention scheme is used. The sequence is sent to a slave arm supervised by a sensor-based motion-planning algorithm and applied to peg-in-hole assembly. Accurate graphic and physical models of slave arm and environment are needed.

Using a pre-planed insertion path, adaptive impedance control (AIC) [36] is used to reduce jamming forces by finding the desired position adaptively to follow the optimal path from the current position and environmental constraints. For peg-in-hole operations, the scheme may correct slight horizontal misalignment due to uncertainties.

In [12] a generalized bilateral control is proposed with scaling, indexing, impedance control, and shared control. The scheme reduces contact forces by setting up soft stiffness at the slave arm and large damping at the master arm. To reduce teleoperation payload, gravity, and damping cause by force feedback a Cartesian mapping [51] is proposed. The remotely-sensed task wrench or computer generated virtual task wrench can be sent to the active hand controller to let the user feels the tasks wrench. The hand controls (1) wrench reflection mapping, (2) force reference, and (3) wrench reflection. The amount of work in peg-in-hole insertion is reduced to about one third than without force information.

A direct bilateral teleoperation (DBT) with kinesthetic force feedback is very useful even with a round trip delay of 6 seconds [30]. Peg-in-hole insertion with 0.4 mm clearance and slope tracing are still feasible using a stable PD controller without the use of scattering theory to guarantee stability. An anthropomorphic space robot is evaluated using kinesthetic and stereo HMD [50]. The operator position is mapped to slave arm both is position and velocity. Evaluation of a drill task indicates less contact forces with equal task time when either visual or kinesthetic force is used with stereo vision.

A mixed of direct and task oriented modes [13] are activated using a set of visualization and manipulation tools with some force monitoring to improve safety and accuracy in micro/nano spaces. To avoid collision high-level motion commands are used due to electrostatic forces and possible sticking.

A telerobotic framework is evaluated using direct teleoperation with following schemes: (1) stereo vision, (2) vision and force feedback, and (3) vision with active compliance. Indexing and scalability tools are used. The proposed system is used to carry out a set of experiments involving contact with the environment. Operator hand motion is mapped scheme to remote tool both in position and force. Strategy for task effective execution is presented for each experiment. Analysis of operator interaction with the environment, task time, and peak and average contact forces is presented. Comments on the global performance of each scheme is presented together with a comparison to other contributions.

The organization of this chapter is as follows. First, the experimental tasks are presented. Second, the used tools and strategy for the above experiments are presented. Third, the global experimental results are presented. Fourth, the obtained results are compared to other contributions and a conclusion is presented.

6.8 Experimentation: task description and specification

In this section we describe the experimental part of a multi-threaded distributed component framework for telerobotics [8] and [9]. In the following sub-sections experiments are presented with their geometric and mechanical specifications.

6.8.1 Peg-in-hole insertion

The objective is to expose the proposed framework to an operation involving the following aspects: (1) teleoperation with kinesthetic force feedback display at the master arm or with active compliance at the slave station, (2) logical mapping of operator hand motion to a floating tool frame (TF) attached to a peg, (3) use of available computer-aided teleoperation (CAT) tools. Insertion deals with grasping of a peg, moving it to the top of hole, detecting contact with the hole, and inserting the peg in hole. The geometric dimensions, clearance, and chamfer geometry are shown on Figure 6.26-(a). For smooth insertion, the peg tip was rounded and the hole was chamfered. To avoid damaging the robot and sensors the hole was attached to a 1 kg base for which the sideways movements are permitted in response to a lateral force exceeding 8 N.

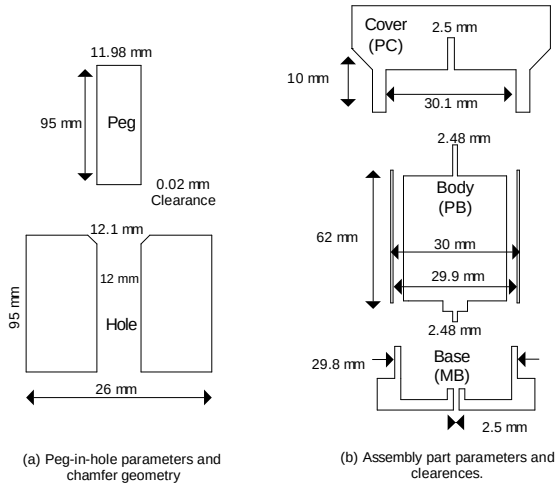

Figure 6.26: Features and parameters of the peg-in-hole and pump components

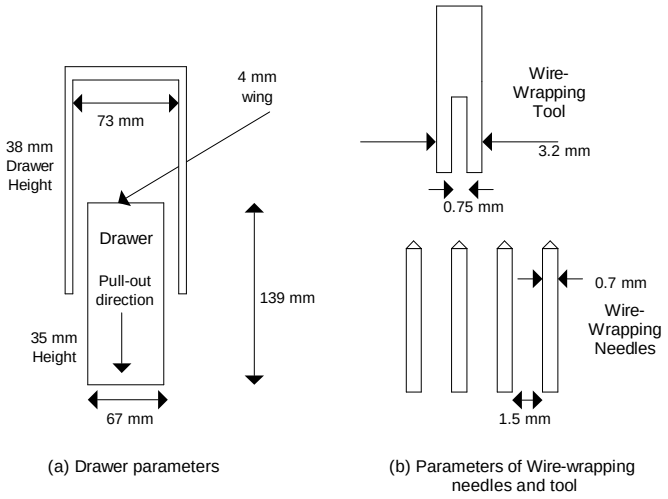

Figure 6.27: Dimensioning parameters of drawers and wire-wrapping experiments

6.8.2 Assembly of a pump

This assembly requires a high-degree of eye-hand motion coordination with balanced dependence on both vision and force feedback. In the studied case the assembly operation requires one or two objectives to be met at the same time. The mechanical tolerance of the parts is relatively moderate as compared to that of the peg and the hole. These are shown on Figure 6.26-(b). A car water pump is used to carry out assembly and disassembly operations. The pump consists of three cylindrical parts: (1) a plastic cover (PC), (2) a metallic base (MB), and (3) a pump body (PB) which contains a motor to be assembled in the middle of the above two parts. The motor shaft axis appears on both top and bottom sides of PB. Initially MB is attached to a fixed platform for which the sideways movements are permitted in response to a lateral force exceeding 8 N. MB can be tilted by up to an angle of 10 degrees with respect to horizontal plan. The task is to grasp PB, move it to top side of fixed MB and carry out part mating of PB and MB. Then the above operations are repeated to assemble PC on the top on PB-MB compound.

6.8.3 Operating drawers

The task of operating drawers requires a high-degree of eye-hand motion coordination with moderate use of force feedback. The used drawer has a small vertical wing at its end to block its entire retrieval

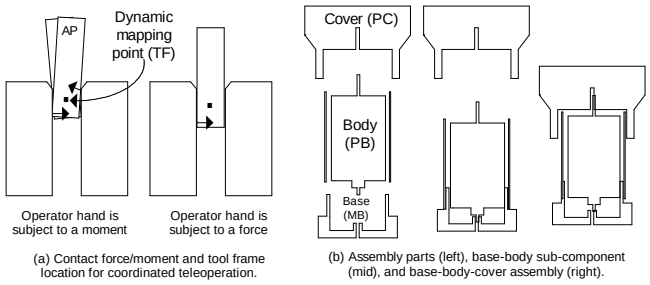

Figure 6.28: Strategy used for peg-in-hole insertion and assembly of a water pump.

by only sliding it outward. Once it reaches the blocking position at its end, the removal of the drawer requires (1) tilting it upward to free its bottom, and (2) sliding it downward to free its top. Figures 6.27-(a) shows the specification of the drawer used.

6.8.4 Pouring

The objective is to expose the proposed framework to the following aspects: (1) teleoperation with fine trajectory and time control, (2) extensive use of eye-hand motion coordination, (3) perception of 3D scene and scene depth, (4) evaluation of functional and ergonomic aspects of proposed CAT tools. This task deals with grasping of a small cup that contains colored water using the slave arm gripper, moving it to the neighborhood of an empty cup, and pouring the water in the target cup.

6.8.5 The wire-wrapping operations

The objective is to evaluate performance of proposed teloperation system in a scaled-down slave arm space. The task is to insert the head of the wire-wrapping tool into a series of needles of a wire-wrapped electronic circuit. The operation must repeat from one needle to next in a row of 5. Figures 6.27-(b) shows the specification of the wire-wrapping gun and needles.

6.9 Experimental methodology

In this section each task will be analyzed with respect to tools used during its teleoperation, methodologies used to ease its achievement, and obtained results. Video clips on this task are available at [3]. In the following subsections the strategy and analysis of carrying out each of the tasks are presented.

6.9.1 Peg-in-hole insertion

The peg-in-hole insertion consists of searching an unconstrained motion path in a space constrained by the jamming F/M. The peg is held by the slave arm gripper. To start, the peg is held in the axial direction of the hole to the best of operator and the 3D vision system. The displayed 6D force feedback represents the forces exerted on the slave arm tool to which the peg is firmly attached. Here 3D vision perception provides coarse information while displayed force feedback is critical to search unconstrained motion directions based on correcting both peg-hole axial and rotational mismatches.

Using the operator-arm interface, a static mapping (S-1) is to link the operator hand to the arm-peg (AP) attachment point. At the server, this mapping also enables computing the external force and moment (F/M) in a frame of reference centered at AP. At the client, the computed external force is displayed at the operator hand using the master arm. Here the stereo vision is not very helpful in making fine translational or rotational motion corrections. Since the operator hand is logically mapped to the peg at point AP, a single F/M contact component corresponds to a subset of coupled F/M being sensed by the operator at AP which might defeat the operator action to nullify above F/M by simple hand motion. We found that it is difficult for a human to comprehend an F/M compound, applied to hand, as opposed to a single F/M component. Therefore setting the

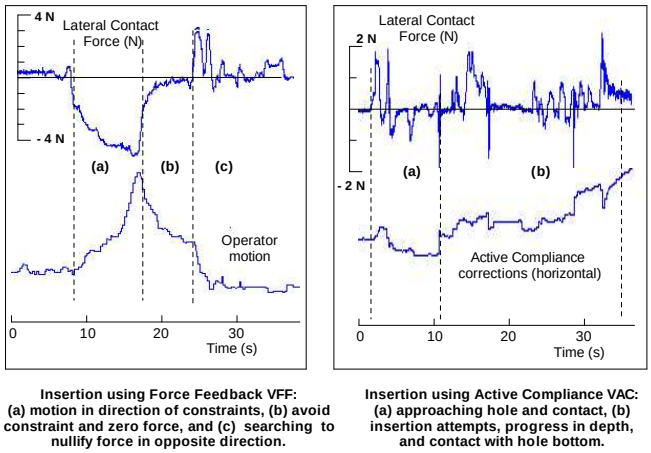

Figure 6.29: Insertion using force feedback (a) and active compliance (b).

motion mapping should be guided by the need to uncouple contact F/M in an attempt to reduce the operator mental load and operation time. For this mapping S-1 was abandoned due to lack of efficiency.

Another mapping (S-2) consists of initially setting the mapping point at the edge of the peg and dynamically compute the new mapping

point by locating it in the middle of peg part that is already inserted in the hole as shown on Figure 6.28-(a). This can be evaluated using (1) the horizontal plan at the top of the hole which is taken as reference for zero depth and (2) current peg depth. This strategy aims at capturing the jamming F/Ms where they are exerted on the peg and display them on the operator hand to favor direct corrections of both peg-hole misalignment errors (moment) and translational errors (force). Hence the objective of this mapping is to logically map the operator hand at a point where it is: (1) effective to capture the mechanical constraints such as the jamming forces, and (2) easy to make necessary correction through motion mapping. Since the above point is dynamically re-mapped to the operator hand motion, thus, the operator rotational and translational corrections are likely to reduce the above constraints due to the one-to-one mapping of the jamming constraints and the corrective motion done by the operator. The scalability function is used here to scale-down the operator motion in all directions to allow fine (1) motion correction in the horizontal plan, and (2) position control of the force exerted by the peg on the hole. Visual monitoring is also used to appreciate the progress in the insertion.

Figures 6.29-(a) and (b) show performance of peg-in-hole insertion using teleoperation with stereo vision and (1) force feedback (VFF), or (2) active compliance (VAC). The upper and lower plots correspond to displayed force feedback and operator motion command, respectively. These interactions are exchanged through the network. In step (a) of VFF, the operator searches an unconstrained motion path in a space constrained by a contact force (-4 N), e.g. a wall effect. In step (b), operator changes direction and reduces lateral contact force which allows the peg to go deeper in the hole. In step (c), a different contact force appears and the same cycle is repeated until completion of insertion.

The third approach (S-3) consists of a supervisory corrective motion done by the local force regulation and the remote slave arm. This solution is similar to the second mapping in terms of measurement of mechanical constraints at the above floating TF but instead of forwarding contact F/M to the operator, active compliance controller is activated at the slave station (shorter loop) which leads to superimpose locally computed peg motion corrections (rotational and axial corrections) to motion instructed by the operator. In this case the operator may limit his control of the peg to vertical direction with the corresponding force feedback. The space scalability function is used here to scale-down the operator motion in the horizontal plan (10:1) with a unit scale in the insertion direction which allows the operator to control the vertical force the peg is exerting on the

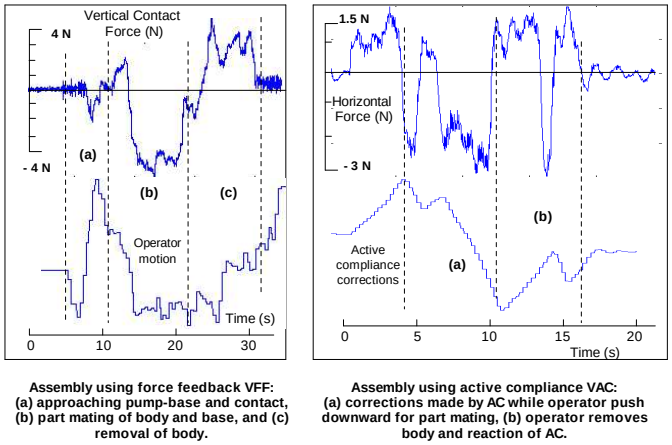

Figure 6.30: Assembly using force feedback (a) and active compliance (b).

hole.

In Figure 6.29-(b) an active compliance control is set at the server. The upper and lower plots correspond to displayed contact force measured at the server and the motion correction made by active compliance controller, respectively. These interactions are local to the slave station. The operator applies a downward force (step (a)) while active compliance control searches a horizontal position and orientation (step (b)) that reduces contact F/M components. Due to proper mapping, F/M components are likely to be uncoupled from each other and corrected independently from each other. This results in the lowest exposure to contact forces.

6.9.2 Assembly of a water pump

The assembly plan is as follows. PB is grasped and moved to the vicinity of MB. Operator carries out axis alignment to the best of the available 3D depth perception. The steps are shown on Figure 6.28-(b). Part mating requires meeting two constraints which are (1) force contact of the motor shaft axis and insertion in the middle hole of MB, and (2) part mating of both lateral cylinders of PB and MB while maintaining axes alignment. The above constraints must be met in a sequential order starting with the best possible configuration that can be achieved using 3D stereo vision and later combining both force feedback and visual information. Similar operation is carried out for assembling PC on the top on PB-MB compound.

The assembly strategy consists of using a balanced mixing of visual and force feedback in addition to space scalability to maintain some geometric directions and keep correcting other references. Specifically the visual feedback is used to establish a proper geometric setting in the pre-positioning phase. The operator space mapping in the horizontal plan is scaled down, for example by a factor of 10:1, to maintain the part positioning and to limit potential motion in the horizontal plan. The vertical axis is left with unit scale under operator control. This approach allows preserving axis alignment (first constraint) of the parts during the part mating operation (second constraint). It allows the operator exerting fine force control in pushing one part into another while monitoring the results. In the case of large positioning errors or axis misalignment during the part mating operation, the tool is lightly lifted up (failure) and the space scalability is increased (for example to 3:1). Correction of part position and orientation are made before attempting again the part mating phase. Force feedback is critical in carrying out the part mating in which the part is subject to a soft down-ward push under careful visual monitoring using zoomed stereo vision for the early detection of potential mismatch. In summary, successful part mating is based on a combination of fine force feedback and 3D depth perception in addition button-controlled tools like indexing and scalability.

Figures 6.30-(a) and (b) show performance of assembly tasks schemes VFF and VAC, respectively. Under VFF scheme, in step (a) PB is moved by the operator to MB where a contact force is detected. Pre-positioning and part mating are performed in step (b). Notice the force feedback (wall effect)

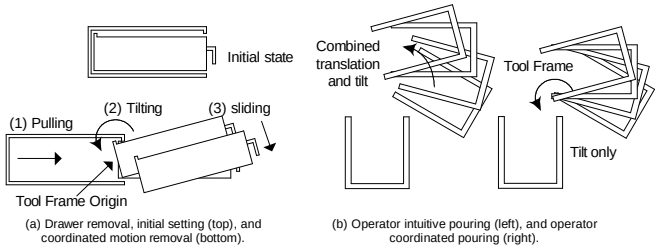

Figure 6.31: Strategy used for operating a drawer and pouring water

displayed on the operator. In step (c) PB is extracted from the assembly with a release force feedback and return to zero force once in free air. The fluctuations in force are caused by the friction.

In Figure 6.30-(b) the sensed contact force is used by the active compliance to carry out corrections of position and orientation of PB while the operator attempts the part mating. Part mating is performed in step (a). Notice the resulting force feedback when the part hits the bottom of MB. In step (b) PB is extracted from the assembly with an additional release force feedback and return to zero force once in free air. The contact forces involved have less magnitude and time than those of the VFF scheme.

6.9.3 Operating drawers

The drawer is pulled up until its bottom wing reaches the blocking point. During the above operation motion scalability can be used to scale down the operator motion in all directions except the pulling direction to maintain directional motion. The blocking end is detected using both visual and force feedback.

To ease the operator task TF must be located at end of the drawer so that the needed tilt operation can be made using a rotation about the horizontal axis of TF. Figure 6.31-(a) shows the initial closed drawer (up) and the steps for its opening and entire removal (down). For this, the GUI, AR tools, and master arm are used to point to the new TF origin O_{TF}. This allows relocating TF. Now the operator hand motion maps logically to TF and the contact F/Ms are now computed with respect to TF before being forwarded to the operator or locally used in the slave active compliance loop. This provides one of the best possible logical mapping from the operator hand to manipulated object so that the needed action is projected on one single axis at the new location of TF. In other words, the operator feels the contact between the wings with the environment as if the drawer is held by the operator. By tilting the hand in the upward direction the front side of drawer is tilted up which frees the drawer bottom that can now be shifted downward before becoming entirely free. During the above operations the displayed contact forces are very helpful in detecting potential contacts that may result from errors in the location of TF and implied operator motion.

6.9.4 Pouring

The pouring operation consists of (1) grasping, (2) traveling, and (3) pouring. Grasping requires the slave arm to move down to a pre-apprehension configuration prior to grasping of a cup (FC) filled with colored water. This is done while continuously trying to center the jaw to middle of FC at its mid height to avoid potential collision. During grasping the indexing function is frequently used to maintain the master arm within a small operator dexterity area whenever the motion requires moving along a path consisting of a long translation or rotation. Traveling requires lifting FC and moving towards the target cup (TC) while maintaining the slave gripper in a horizontal plan and progressive setting up of FC orientation when approaching TC. Pouring requires setting up a proper pre-pouring configuration in the vicinity of TC. Now FC must be tilted while keeping its top above TC. This normally consists of a rotation and a translation as shown in the left part of Figure 6.31-(b). To reduce the workload on the operator it is more interesting to relocate the mapping function

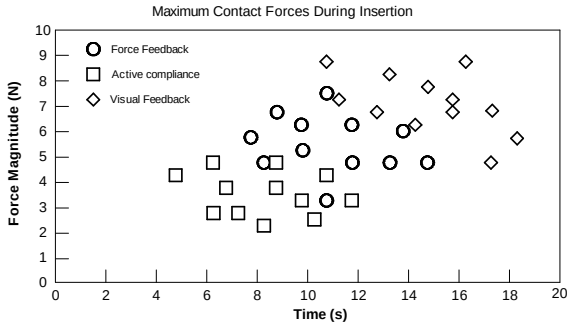

Figure 6.32: Maximum insertion forces with respective task times

of the slave tool at one of the top lateral point of TC which becomes the origin of the new TF. In this case tilting the operator hand leads to directly tilting the new TF about one axis of above frame as shown in the right part of Figure 6.31-(b). We note that placing TF origin (and orientation) at the above critical point contributed in reducing the task time by about 40% as compared to default setting of TF at gripping point. In addition, it produces a predictable trajectory while being an ergonomic teleoperation tool.

6.9.5 The wire-wrapping operations

The GUI is used to set up (1) scale level of force feedback, and (3) the camera zooming level. The space scaling is directly controlled by the operator index. The converging setting consists of scaling the operator motion by a factor of 30:1, the force feedback by a factor of 1:10, and stereo camera zooming by a factor of 1:40. The operator moves the wire-wrapping tool (WWT) while aligning its axis with the circuit needle using 3D vision and carries out the insertion. The operator needs to feel the vertical force component to avoid damaging the needle during contact because actually only a small central hole in WWT must fit the needle as shown on Figure 6.27-(b). In this case the operator carries out corrections of axis mis-alignment, and (2) insert WWT head in the needle. The distance between two needles is about 1.5 mm. The above task was successful in making five successive insertions in a line in 30 seconds. The operator adaptation to teleoperation to small scale appeared to be smooth and simple. Multiple zooming views is useful to avoid changing the zooming level whenever axis alignment needs correction.

6.10 Experimentation: results and discussion

In this section we present the global results of using the proposed telerobotic system in performing the above teleoperation tasks, the used CAT tools, and recommendations on how to improve the man-machine interfacing in the telerobotic system.

6.10.1 Analysis of teleoperation tasks

In this section we present the results of using the proposed telerobotic system in performing peg-in-hole insertion and assembly operation. These tasks are selected because they require effective interaction between the operator and the remote task involving fine motion, force feedback, and stereo vision. The results are limited to the period of interactions with the environment such as the insertion phase in peg-in-hole operation and part mating phase in the assembly operation. Both phases follow the contact detection phase. We study the following teleoperation schemes in which the operator has control of a 6 DOF master arm and provided with (1) only 3D visual feedback (V), (2) 6 DOF kinesthetic force feedback with 3D visual feedback VFF, and (3) visual feedback with active compliance VAC operating at the slave site.

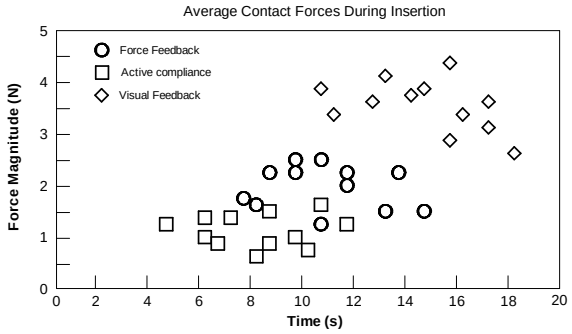

Figure 6.33: Average insertion forces with respective task times

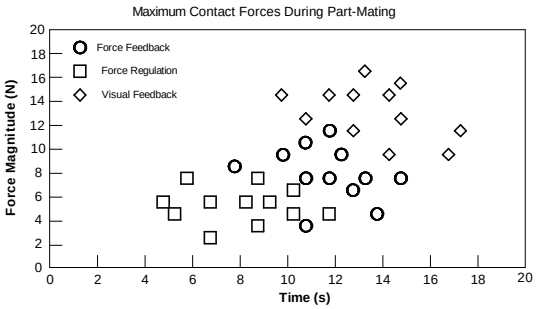

Figure 6.34: Maximum assembly forces with respective task times

Each of the insertion and part mating phases was carried out by 12 operators, each carried out successively schemes V, VFF, and VAC. Each scheme was carried out 12 times in total for each of the above two phases. Each operator was allowed to experience the task at least 10 times before recording the data. The operators are aware of the need to minimize contact forces to reduce potential damage and improve operation effectiveness. The collected data refers to maximum and average F/M magnitudes as well as the corresponding completion time of a given operation for a given operator. Combined results for all operators are presented. In total we have 72 plots for two phases. The F/M vectors are all computed at the origin of reference TF and sampled at 50 Hz on slave arm station. Both force and moment components are similar in terms of concluding remarks.

Figures 6.32 and 6.33 show the maximum and average force exerted during peg-in-hole insertion with respective operation times. The operators were satisfied with the quality of stereo vision provided during the experiments. Scheme V allows completion of the insertion but with the largest contact forces and completion times as compared to VFF and VAC. Occasionally V scheme may produce less contact forces and possibly less duration than the other two schemes. Stereo vision is one critical operator augmentation in telerobotics. However, the use of only visual feedback for any operator indicates that another critical feedback is missing as both peak and average forces dominate as compared to the other schemes. The average force indicated some dependence on the operator speed and overall performance for a given operator. The ranking of any given operator performance is mainly the same in each scheme.

Figures 6.34 and 6.35 show the maximum and average force exerted during assembly of the pump with respective operation times. Similar to the insertion case, the VAC scheme outperforms the V and VFF but with less deviations both in maximum and average contact force. Table 6.1 shows the ratio of maximum and average force and task time of schemes V and VFF over scheme VAC for both insertion and assembly tasks. For the insertion, teleoperation with active compliance VAC allowed carrying out the tasks with the least task times and contact force. For the insertion, scheme V allows completion of the insertion but with significant increase in contact forces and task time as compared to VAC. Teleoperation with VFF slightly increases task time but with a noticeable increase in contact forces as compared to VAC. VFF produces the largest variation in time from one

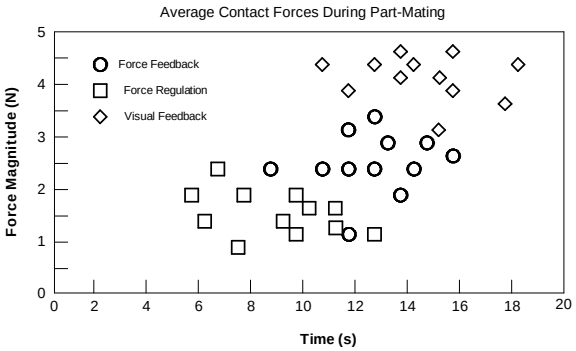

Figure 6.35: Average assembly forces with respective task times

Comparison of force and task time in insertion and assembly

	Force Magnitude		Task Time
Operation(ratio)	Maximum	Average	duration
Insertion (V/VAC)	1.93	3.85	1.76
Insertion (VFF/VAC)	1.45	1.89	1.33
Assembly (V/VAC)	2.33	2.74	1.55
Assembly (FF/VAC)	1.27	1.63	1.36

Table 6.1: Peak force and task duration for V and VFF vs. VAC

operator to another. This shows that active compliance loop at the server station is better prepared to correct contact forces than the remote human. This shows the efficiency of supervisory approach and its local active compliance that continuously searches to nullify the external F/M by correcting tool position and orientation at current TF. Occasionally VAC gets higher times due to temporary blocking caused by excessive vertical force commanded by the operator.

For the assembly tasks, teleoperation with VAC is still ranked first but with less advantages as compared to the case of the insertion. The reason is probably due to operator ability to combine force feedback with 3D perception in the critical phases of the part mating.

In both insertion and assembly, VAC contributed in reducing peak and average contact forces as compared to both V and VFF schemes especially in the case of the insertion. VAC equally reduced task time for each of the FF and V schemes in both of insertion and assembly tasks.

Telerobotics needs advanced supervisory tools that embed complex force and position control with dynamic motion/force mapping. Effective man-machine interfacing is needed to integrate the above tools in a simple and natural way at the operator index and stereo space. The stereo space visible to the operator and its AR capabilities need to support the above integration. The connectivity between master and slave arms can be temporarily switched off and master arm used as a 3D pointer. A variety of active compliance (AC) scenarios can be associated graphical features in the operator stereo space. These can be selected and grouped in AC compounds (CACs) as task-oriented and operator favored tools. To activate a specific CAC at the task point of interest, the operator can point to a given CAC in stereo space, drag it, change its orientation, drop it at a desired tool point, and control its activation. This allows composing optimized AC mechanisms and efficiently activating them at the right location and orientation with respect current task.

6.11 Comparison to others

In virtual reality based teleoperation [17, 16, 11, 13] the operator plans an operation using a model, the plan controls a slave arm, and slave arm transmits back parametric feedback. The primary issue is operation safety. Mainly off-line approaches are used and teleoperation is carried out on a static environment with no dynamic interaction reported. However, in [19] graphic animation of robot kinematics, dynamics, friction, and impact forces used in a closed loop control provides the operator the feeling of repulsive forces which allowed to carry out peg-in-hole insertion. The proposed telerobotic framework provides direct-oriented teleoperation with CAT tools augmented with some supervisory control schemes to improve teleoperation effectiveness in real interactions with the environment.

We concur with [30] on the importance of kinesthetic force feedback in assembly operations. We extended direct teleoperation by using compliance control that makes the slave arm continuously searching to nullify F/M sensed on the current tool while the whole arm is being driven by the operator to take advantage of the above mechanism in current task. In comparison to [36] our proposed VFF and VAC schemes have similar effects in modifying task trajectory. The active compliance controller continuously searches corrections in tool position and orientation that reduce tool external F/M. Operator sets task-oriented compliance and leads the arm under compliance equilibrium to work location. Proposed VAC reduced peak contact forces and task time as compared to kinesthetic force feedback with vision in insertion and assembly tasks. VAC may also be useful as a task locality mechanism to ensure task continuity in delayed teleoperation.

As compared to [12] the proposed system also uses indexing and scale in addition to a tool-oriented dynamic motion mapping in position and force to carry out coordinated motion as a strategy to reduce operator cognitive load. This enables force reflection from current tool to the operator which increases the feeling of telepresence and enables teleoperation tasks to be completed more easily and with lower contact forces.

The wrench mapping of [50] is comparable to proposed tool motion and force mapping. However, our dynamic mapping scheme showed to be useful tool for many tasks where the point of interest is function of task state. Proposed mapping makes the operator logically mapped, in position and force, to remote object. The accomplishment of above experiments is fundamentally due to proposed dynamic mapping scheme which is estimated to be the most critical CAT tool in proposed telerobotics.

6.12 Conclusion

A set of assembly tasks has been used to evaluate a client-server telerobotic system which transfers motion, force feedback, and stereo vision over a network. The tasks are (1) peg-in-hole insertion, (2) assembly of a pump, (3) operating drawers, (4) pouring of water, and (5) wire-wrapping. The operator has been provided with (1) stereo vision V, (2) vision and force feedback VFF, and (3) vision and active compliance VAC. Active compliance is has been used as assistance to direct teleoperation in addition to indexing and scalability tools. A dynamic mapping of operator hand motion and force to a task-oriented tool point has been used to reduce operator cognitive load and task time. Scheme V allowed to complete the above tasks but resulted in the largest contact forces and task times as compared to VFF and VAC. In contact centric tasks, like insertion, VAC noticeably outperforms V and VFF and provides task quality control. In multi-objective tasks, like assembly, VAC and VFF are closer in peak and average force as well as in task times. However, VFF has variable delivery as it depends more on operator skills. Button-controlled indexing and scalability proved to be the most frequently used tools. Scalability was useful to operate in a 30:1 scaled down space as well as a linear dimension blocking tool. Force feedback and active compliance are critical tools for extending human eye-hand motion coordination and dexterity to remote work in hazardous, hostile, un-accessible, and small-scale environments. To augment teleoperation a master arm-driven graphical tool is proposed for composing compliance mechanisms that can be dragged, attached to a tool point, and activated

to ease man-machine interfacing.

6.12.1 Acknowledgement

This work is supported by King Abdulaziz City for Science and Technology (KACST) under research project grant AT-20-80. The authors thank Mr. M. Faheemuddin, Mr. M. Buhari, and Mr. S. Islam for their help in setting up campus routes. The authors acknowledge computing support from King Fahd University of Petroleum and Minerals (KFUPM).

Bibliography

[1] The Gram-Schmidt algorithm. *http://www.math.hmc.edu/calculus/ tutorials/gramschmidt/, 2002.*

[2] Microsoft's DirectX and .NET websites. *http://www.microsoft.com/directx/ and http://www.microsoft.com/net/, 2002.*

[3] Telerobotics video clips. *http://www.ccse.kfupm.edu.sa/researchgroups/ robotics1/, April, 2003.*

[4] A. Al-Harthy. Design of a telerobotic system over a local area network. *M.Sc. Thesis, King Fahd University of Petroeum and Minerals*, January 2002.

[5] M. Al-Mouhamed, M. Nazeeruddin, and Sayed Islam. Experimentation of a multi-threaded distributed telerobotic framework. *Submitted for publication*, December 2005.

[6] M. Al-Mouhamed, M. Nazeeruddin, and N. Merah. Design and analysis of force feedback in telerobotics. *Submitted for publication*, December 2005.

[7] M. Al-Mouhamed, O. Toker, and A. Al-harthy. A 3D vision-based man-machine interface for telerobotics. *IFAC Inter. Conference on Intelligent Control and Signal Processing*, April 8-11 Portugal, 2003.

[8] M. Al-Mouhamed, O. Toker, and A. Iqbal. A multi-threaded distributed telerobotic framework. *To appear in IEEE T. on Mechatronics*, December 2005.

[9] M. Al-Mouhamed, O. Toker, and A. Iqbal. Performance evaluation of a multi-threaded distributed telerobotic framework. *Submitted for publication*, December 2005.

[10] W.J. Book, H. Lane, L.J. Love, D.P. Magee, and K. Obergfell. A novel teleoperated long-reach manipulator testbed and its remote capabilities via the Internet. *Proc. of the IEEE Inter. Conf. on Robotics and Automation*, 3(6):1036–1041, 1997.

[11] Bailin Cao, G.I. Dodds, and G.W. Irwin. An event driven virtual reality system for planning and control of multiple robots. *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, 1999. IROS '99*, 2:1161–1166, Oct. 1999.

[12] Tan Fung Chan and R.V. Dubey. Design and experimental studies of a generalized bilateral controller for a teleoperator system with a six dof master and a seven dof slave. *Proc. IEEE/RSJ/GI International Conference on Intelligent Robots and Systems '94. 'Advanced Robotic Systems and the Real World', IROS '94*, 1:415 – 422, Sept. 1994.

[13] A. Codourey, M. Rodriguez, and I. Pappas. A task-oriented teleoperation system for assembly in the microworld. *Proc. 8th International Conference on Advanced Robotics, 1997. ICAR '97*, pages 235–240, July 1997.

[14] J.P. Desai and R.D. Howe. Towards the development of a humanoid arm by minimizing interaction forces through minimum impedance control. *Proc. IEEE International conference on Robotics and Automation, ICRA*, 4(2):4214 – 4219, 2001.

[15] G. N. DeSouza and A.C. Kak. A subsumptive, hierarchical, and distributed vision-based architecture for smart robotics. *IEEE Trans. on Sys., Man, and Cyber.-Part B: Cybernetics*, 34:5:1988–2002, 2004.

[16] E. Even, E. Fournier, and R. Gelin. Using structural knowledge for interactive 3D modeling of piping environments. *Proc. of IEEE Inter. Conf. on Robotics and Automation*, pages 2013–2018, Apr. 2000.

[17] E. Even, P. Gravez, E. Maillard, and E. Fournier. Acquisition and exploitation of a 3D environment model for computer aided tatleloperation. *Proc. of the 1999 IEEE Inter. Workshop on Robot and Human Integration*, pages 261–266, Sep. 1999.

[18] O.D. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig? *In Proc. 2nd European Conference on Computer Vision, LNCS 588, Springer-Verlag*, pages 563–578, 1992.

[19] N. Funabiki, K. Morishige, and H. Noborio. Sensor-based motion-planning of a manipulator to overcome large transmission delays in teleoperation. *Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics, 1999. IEEE SMC '99*, 5:1117–1122, Oct. 1999.

[20] F. Goktas, J.M. Smith, and R. Bajcsy. Telerobotics over communication networks. *Proc. of IEEE Conference on Decision and Control*, 3:2399–2404, 1997.

[21] K. Goldberg and R. Siegwart, editors. *Beyond Webcams: an Introduction to Online Robots*. The MIT Press, Cambridge, MA, USA, 2001.

[22] Teresa Ho. System architecture for Internet-based teleoperation systems using java. Master's thesis, Department of Computing Science, University of Alberta, Canada, 1999.

[23] Y.E. Ho, H. Masuda, H. Oda, and L.W. Stark. Distributed control for tele-operations. *Proc. of the 1999 IEEE/ASME International Conf. on Adv. Intelligent Mechatronics*, pages 323–325, September 1999.

[24] Y.E. Ho, H. Masuda, H. Oda, and L.W. Stark. Distributed control for tele-operations. *IEEE/ASME Transactions on Mechatronics*, 5(2):100–109, June 2000.

[25] N. Hollinghurst and R. Cipolla. Human-robot interface by pointing with uncalibrated stereo vision. *Image and Vision Computing*, 14(3):171–178, 1996.

[26] X. Hou and J. Su. New approaches to internet based intelligent robotic system. *IEEE International Conf. on Robotics and Automation*, pages 3363–3368, 2004.

[27] R.D. Howe. A force-reflecting teleoperated hand system for the study of tactile sensing in precision maniplulation. *IEEE Inter. Conferance on Robotics and Automation*, pages 1321–1326, May 1992.

[28] R.D. Howe and Y. Y. Matsuoka. Robotics for surgery. *Annual Review of Biomedical Engineering*, pages 211–240, 1999.

[29] Huosheng Hu, Lixiang Yu, Pui Wo Tsui, and Quan Zhou. Internet-based robotic systems for teleoperation. *International Journal of Assembly Automation*, 21(2):143–151, May 2001.

[30] T. Imaida, Y. Yokokohji, T. Doi, M. Oda, and T. Yoshikwa. Ground-space bilateral teleoperation experiment using ETS-VII robot arm with direct kinesthetic coupling. *Proc. of the IEEE Int. Conf. on Robotics and Automation,ICRA 2001*, 1:1031 – 1038, June 2001.

[31] H. Iwata. Artificial reality with force-feedback: Development of desktop virtual space with compact master manipulator. *ACM SIGGRAPH Computer Graphics*, 24 (4), 1990.

[32] W. Kim, B. Hannaford, and A. Bejczy. Force reflection and shared compliant control in operating telemanipulatord with time delay. *IEEE Trans. on Robotics and Automat.*, pages 176–185, Apr 1992.

[33] S. Knoop, S. Vacek, R. Zollner, C. Au, and R. Dillmann. A CORBA-based distributed software architecture for control of service robots. *IEEE/RSJ Inter. Conf. on Intelligent Robots and Systems*, pages 3656–3661, 2004.

[34] Kazuhiro Kosuge, Hideyuki Murayama, and Koji Takeo. Bilateral feedback control of telemanipulators via computer network. *Proc. of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3:1380–1385, Nov. 1996.

[35] D. Lawrence. Stability and transparency in bilateral teleoperation. *IEEE Trans. on Robotics and Automat.*, 9(5):624–637, Oct 1993.

[36] Sukhan Lee, Ming-Feng Jean, Jong-Oh Park, and Chong-Won Lee. Reference adaptive impedance control: a new paradigm for event-based robotic and telerobotic control. *IEEE/RSJ International Conference on Intelligent Robots and Systems, 1998*, 2:1302 – 1307, Oct. 1998.

[37] R. C. Luo and T. M. Chen. Remote supervisory control of an autonomous mobile robot via world wide web. *Proc. of IROS '97*, 2:1163 – 1168, 1997.

[38] M. M. Arnoretti, S. Bottazzi, M. Reggiani, and S. Caselli. Evaluation of data distribution techniques in a CORBA-based telerobotic system. *Intl. Conf. on Intelligent Robots and Systems*, pages 1100–1105, 2003.

[39] D.W. Marhefka and D.E. Orin. A compliant contact model with nonlinear damping for simulation of robotic systems. *IEEE Trans. on Sys., Man, and Cybernetics, Part A*, 29(6):566–572, 1999.

[40] Microsoft. MSDN library. *http://msdn.microsoft.com/default.asp, 2002.*

[41] Xi Ning and T.J. Tarn. Action synchronization and control of Internet based telerobotic systems. *Proc. of IEEE Inter. Conf. on Robotics and Automation*, 1:219–224, 1999.

[42] F. Paolucci and M. Andrenucci. Teleoperation using computer networks: Prototype realization and performance analysis. *Proc. of the 8th Mediterranean Electrotechnical Conference, MELECON'96*, 2:1156–1159, 1996.

[43] A. Rovetta, R. Sala, Xia Wen, and A. Togno. Remote control in telerobotic surgery. *IEEE Trans. on Sys., Man, and Cybernetics, Part A*, 26(4):438–444, 1996.

[44] S.E. Salcudean, K. Hastrudi-Zaad, S.P. Tafazoli, S. DiMaio, and C. Reboulet. Bilateral matched impedance teleoperation with application to excavator control. *Proceedings of the IEEE International Conference on Robotics and Automation, Leuven, Belgium*, pages 133 – 139, May 1998.

[45] S.E. Salcudean, N.M. Wong, and R.L. Hollis. A force-reflecting teleoperation system with magnetically levitated master and wrist. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1420–1426, 1992.

[46] S.E. Salcudean, N.M. Wong, and R.L. Hollis. Design and control of a force-reflecting teleoperation system with magnetically levitated master and wrist. *IEEE Transactions on Robotics and Automation*, 11(6):844–858, December 1995.

[47] T. Sheridan. Supervisory control. *G. Salvendy (Ed.) Handbook of human factors and ergonomics*, pages 1295–1327, 2002.

[48] L. Stocco and S.E. Salcudean. A coarse-fine approach to force-reflecting hand controller design. *Proc. of the Inter. Conf. on Robotics and Automation*, 1:404–410, 1996.

[49] D. Wang, X. Ma, and X. Dai. Web-based robotic control system with flexible framework. *IEEE Inter. Conf. on Robotics and automation*, pages 3351–3356, 2004.

[50] L.E.P. Williams, R.B. Loftin, H.A. Aldridge, E.L. Leiss, and W.J. Bluethmann. Kinesthetic and visual force display for telerobotics. *IEEE Inter. Conf. on Robotics and Automation, ICRA '02*, 2:1249–1254, 2002.

[51] R. L. Williams II. Cartesian control of force-reflecting hand controllers. *Proceedings of the Fifth National Conference on Applied Mechanisms and Robotics, Cincinnati OH*, Oct. 1997.