QoS Management of Supermedia Enhanced Teleoperation via Overlay Networks*

Zhiwei Cen, Matt Mutka
Dept. of Computer Science and Engineering
Michigan State University
East Lansing, MI 48824, USA
{*cenzhiwe,mutka*}*@cse.msu.edu*

Yang Liu, Amit Goradia, and Ning Xi
Dept. of Electrical and Computer Engineering
Michigan State University
East Lansing, MI 48824, USA
{*liuyang4,goradiaa,xin*}*@egr.msu.edu*

***Abstract*— In supermedia enhanced Internet based teleoperation systems, the data flowing between the operator and the robot include robotic control commands, video, audio, haptic feedback and other media types. The difference between an Internet based teleoperation system and other Internet applications are that (1) there are many media types involved in teleoperation systems and each of them has a particular Quality of Service (QoS) requirement; and (2) some media types are very latency sensitive. Overlay networks have been proposed to improve the QoS of teleoperation applications. However efficient use the overlay network resources and the distribution of these resources optimally to all supermedia streams remains an important problem. This paper aims to provide a framework of QoS management for teleoperation systems over overlay networks. The validity and performance of the system is evaluated using the PlanetLab overlay network.**

***Index Terms*— Teleoperation, Supermedia, Overlay networks, Quality of Service.**

I. Introduction

In an Internet based teleoperation system, the operator controls the robot through a communication channel established by Internet routers and switches. A supermedia enhanced teleoperation system [1] allow control commands, audio, video, force feedback and other sensory information to be transmitted between the operator and the robot. In supermedia enhanced systems, a larger variety of media are involved. All the media types are to be transmitted through a shared path in the Internet. Each supermedia type has a different quality of service requirement.

However, the Internet is not designed to be a reliable communication channel. Random time delay, packet loss, network buffering effects and disconnections are all intrinsic characteristics of the best effort service model of the current Internet. Of all the unpredictable elements of the Internet, time delay is one of the biggest challenges in real-time teleoperation systems. High latency can stall a robot in the middle of a task. To build a reliable teleoperation system, we may proceed in two directions:

1) Improve the control mechanism of the teleoperation system to accommodate the unpredictable nature of the Internet. The traditional control system assumes a dedicated communication channel between the operator and the robot. In order for the system to work over the Internet, some of its assumptions must be changed.
2) Improve the quality of service of the Internet to make it close to the dedicated communication channels used by the traditional system. One of the possible ways to help build a stable control system is to reduce the end-to-end latency of the communication channels.

Event based teleoperation systems [1] were presented to address the problem in the first direction. The performance difficulty caused by the Internet based teleoperation system is a result of using time as a reference for different system entities. In the event-based control approach, a non-time based reference is used. An event reference is a monotonically increasing parameter to synchronize the operator and the robot. Through the event reference, low level sensing and control modules are integrated with high level task scheduling and action planning. The system is able to cope with the uncertainties caused by the Internet and provide event transparency to the system user. Fung, et al. introduced a Task Dexterity Index (TDI [1]) of the robotic task to reflect the dexterity of robotic task online. A Task Dexterity Index (TDI) is generated for each data stream using a fuzzy logic system to describe the bandwidth requirement of the robotic task. Later, this TDI may be used in the bandwidth allocation algorithm to rank the data streams that need to be transmitted. Event based control provides a reliable control mechanism for teleoperation system. However, in order to make the teleoperation system reliable and efficient, something must be done in the second direction.

There are extensive research efforts dedicated to provide QoS aware transport service over the Internet. Some of the most notable work, including Integrated Service (IntServ) and Differentiated Services (DiffServ), aim to change the best-effort service model to a QoS aware infrastructure. However, IntServ and DiffServ are not deployed over major networks due to a variety of reasons. Many other research efforts aim to improve QoS levels for multimedia applications over the current best effort Internet [2] [3]. However these transmission mechanisms do not solve issues in supermedia transmission due to the following reasons:

1) A teleoperation system involves several kinds of media

*This work is partially supported by NSF Grant HNI-0334035.

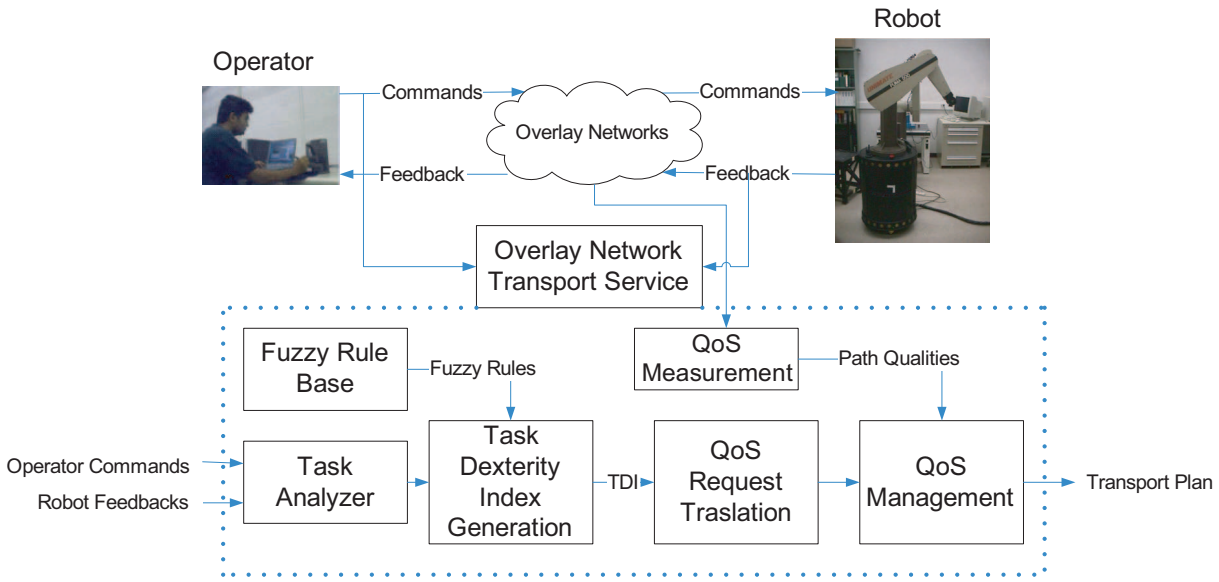

Fig. 1. Usage of Overlay Networks to Improve the QoS

types, such as video, audio, control commands, haptic information, and code uploads. Different types of supermedia stream have different QoS requirements. Video and audio streams may be unreliable but need to have an adaptive real time transmission service. Control commands and haptic information need a timely and reliable transmission service. Code uploads need reliable transmission service but are not time sensitive.

2) Supermedia streams have dynamic QoS requirements associated with the task execution process. The priority of a supermedia stream will change dynamically when when the robotic tasks change. For example, video streams may have higher priorities when the robot is approaching an object or trying to circumvent obstacles, while haptic feedback may become the highest priority when the robot starts to manipulate the object.

In Supermedia TRansport over Overlay Networks (STRON, [4] [5]), each media stream is chopped into a series of messages. The message is encoded using a class of digital fountain encoding algorithms [6]. Given p packets for a certain message, STRON encodes the p packets into αp packets, where α $(\alpha \geq 1)$ is the stretch factor. The encoded data packets are scheduled to be transmitted over multiple overlay paths. In order for the QoS of the paths to be independent, the overlay paths are preferably disjoint [7], [8], [9]. As soon as the receiver collects p distinctive encoded data packets, the decoding algorithm can reconstruct the original message. The usage of overlay networks to improve QoS of supermedia enhanced teleoperation systems can be illustrated in Figure 1. The experiments show that STRON was effective in reducing end to end latencies of teleoperation systems. However, considering that various supermedia streams have different QoS requirements, and all need to be transmitted over the same set of overlay paths, it is important to design a QoS management framework to assign the networking resource to the supermedia streams efficiently and optimally. This is one of the major problems we are going to solve in this paper.

Other related research on overlay networks include ON [10], QRON [11], SON [12], and PlanetLab. There is much work that deals with bandwidth QoS control and resource allocations. Most of the work is related with network based Quality of Service approaches [13] [14]. The transcoding method is used to adaptively change the encoding schemes according to the network conditions [15] [16]. For the supermedia transport framework, the main concern is to solve the competition for bandwidth among different supermedia streams since the networking is beyond the control of the end systems. Thus the bandwidth sharing and control mechanisms should be end-system based. Some of the existing end-host QoS control methods try to control resources by adjusting TCP parameters [17] [18]; others presented new transport protocols [19] [20]. However none of these methods can be applied to supermedia transmission due to the various types of media involved and the changing QoS requirements of these media types.

The purpose of this paper is to provide a reliable and efficient teleoperation system QoS management framework. In this framework, latency sensitive supermedia streams can be encoded using redundancy codes and transmitted over multiple overlay paths. A QoS management module ensures the important supermedia streams will have higher priority while sharing the overlay bandwidth with other media streams. The receiver will collect the first received data from the multiple paths and ensure reliable and fast delivery of the application data. The networking routes and encoding redundancy may be adjusted dynamically to meet the QoS requirements of the supermedia streams in face of networking performance degra-

dation. TCP Friendly Rate Control (TFRC [2]) is used as the congestion control mechanism for each overlay connection, which ensures that the supermedia traffic remains friendly to other Internet traffic.

The rest of this paper has the following structure. In section II, the architecture of the transport service is presented. We discuss the experimentation and simulation methodology and results in section III. Section IV provides the conclusions.

II. Design of the Overlay Network Transport Service

A. Task Dexterity index Generation

In order for the overlay transport services module to allocate the networking resources optimally, a mechanism is needed to model the dexterity and resource requirement of robotic tasks. This mechanism is task dexterity index. An on-line task dexterity analyzer is proposed to analyze task dexterity requirements by Pan, et al. [21]. The operator commands, the information from the mobile manipulator and the sensory information are taken as inputs and a collection of task indicators is generated online, which represent the actual conditions of the task being performed. Using a fuzzy inference scheme, the task analyzer maps the task conditions represented by task indicators into TDI, which represents task dexterity requirements. The task indicator vector $\mathbf{u} \in R^M$ (set of all task indicators) is defined as:

$$\mathbf{u} = \{u_1, u_2, \cdots, u_M \mid \forall u_i \in U, \quad i = 1 : M\}^T. \qquad (1)$$

where, M is the number of task indicators. The following is an example of the collection of task indicators which could be used for the tele-operated mobile manipulator:

- End-effector absolute motion on position:

  $$\Delta P_{arm} = \sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}$$

  where Δx, Δy, Δz are the position transformation on x, y, and z axes in the world frame.
- End-effector absolute motion on orientation:

  $$\begin{aligned} \Delta\Theta_{arm} &= Abs(\arccos 0.5(Tr(R_{err}) - 1)) \\ R_{err} &= R_{des} R'_{act}. \end{aligned}$$

  Here R_{err} is the product of the desired rotation matrix and the transpose of the actual rotation matrix, Tr is the trace of a matrix and Abs is the absolute value.
- Local curvature k and torsion τ of the trajectory, which are described by the previous and current commands.
- Operator commands. The characteristics (say acceleration and deceleration) of commands can indicate task conditions and the operator's intention.
- Status of the mobile manipulator (such as end-effector open/close, the external force, *etc.*).
- Sensory information about the environment (say obstacles, working environment temperature, *etc.*).

The Task Dexterity Index is used to mathematically describe the task dexterity requirements. The index is generated by a fuzzy mapping, which takes the task indicators as input variables:

$$Index = f(\mathbf{u}) = \frac{\sum_{j=1}^{M} a_j g_j(u_j)}{\sum_{j=1}^{M} a_j}. \qquad (2)$$

where a_j is the relative weight of task indicator u_j in the task, and g_j is the mapping function of the task indicator u_j for the task. Equation (2) maps the task indicator value into the extent of the task dexterity requirement. The relative weights are assigned based on the physical meaning and importance of a particular task indicator.

B. Transport Service Architecture

The system architecture is shown in Figure 2. The design details of Sender-side Overlay Agent and Receiver-side Overlay Agent can be found in [4] [5]. At the sender side, the supermedia streams are classified according to their roles in the teleoperation system by a classifying system. Each class of stream has its own QoS requirement. For teleoperation applications, the supermedia streams have three classes. Class 1 is for small but high frequency data streams, such as temperature and force feedback. These data streams have a low bit rate but whenever a data packet is available it requires timely delivery. We may not apply redundancy encoding to supermedia of this class and simply transfer copies of the data over several overlay network paths. Class 2 also works for low bit rate and high-frequency supermedia streams, such as the force feedback when multiple force detectors are present. For class 2 supermedia streams, the data generated during each sample interval cannot fit into one packet. Hence redundancy encoding is needed to provided a reliable transmission. However, since the amount of data being sent is not large, a high redundancy level may be used to ensure reliability and timeliness. Class 3 is designed to transmit high bit rate supermedia streams that do not have a strict requirement on reliability. Examples of class 3 supermedia streams include video and audio. A certain level of redundancy is used to transmit class 3 data over multiple overlay paths. We may also differentiate video and audio streams since sometimes audio streams are more sensitive to data loss than video streams. Differentiated treatments of each supermedia class are implemented in the QoS Management Module (QMM).

The Sender-side Overlay Network Agent (SONA) is responsible for transporting each class of streams according to its QoS specifications. The SONA consists of four submodules: the Disjoint Path Search Module (DPSM), the Network Measurement Module (NMM), the Optimal Path Selection Module (OPSM), and the Encoding and Transport Module (ETM).

The Disjoint Path Search Module (DPSM) runs a disjoint overlay network path searching algorithm [7] [8] [9] through the connected overlay nodes. A higher disjoint degree of the overlay paths ensures that if one of the paths experiences

congestion or service outage, the performance of other paths will be minimally influenced.

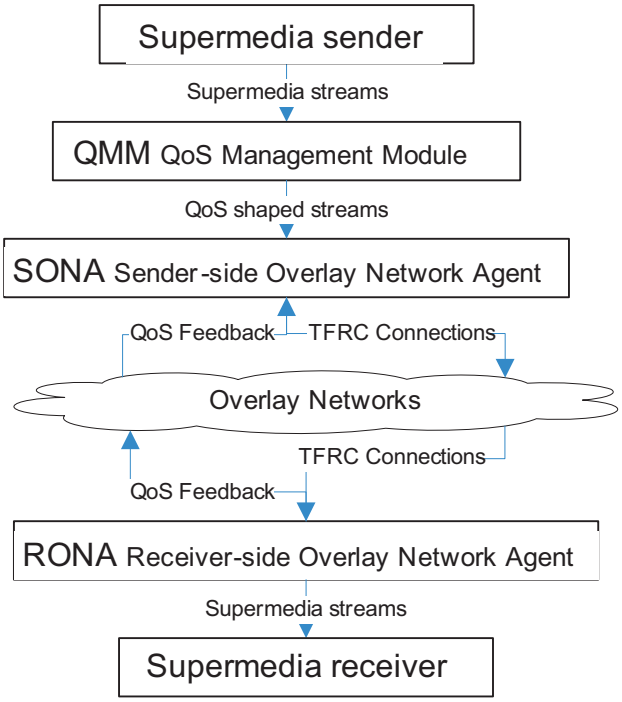

Fig. 2. The Architecture of the Overlay Network Transport System

The Network Measurement Module (NMM) is usually deployed in overlay nodes as a common service. Active measurements are launched periodically to probe the networking conditions. The measurement results needed by SONA include, the available bandwidth, the single trip delay, and the packet loss rate. Some research efforts have been dedicated to making accurate measurements of the network. In our framework, we use *Pathload* as the available bandwidth measurement tool. *Pathload* [22] takes advantage of the fact that the one-way delays of a periodic packet stream show an increasing trend when the stream rate is larger than the available bandwidth. The measurement of single trip delay and packet loss could be trivial except that half of the round trip time is taken as the single trip delay for simplicity.

The objectives of OPSM are to find the optimal disjoint path set to be used as active transmission paths and decide the amount of data sent over each active path. The redundancy encoding parameters are also decided in OPSM. The design of the OPSM is discussed in detail in [4].

The Encoding and Transport Module (ETM) encodes the supermedia streams using using redundancy encoding and passes the encoded packets to the overlay transport service. TCP Friendly Rate Control (TFRC [2]) is used as the transport control mechanisms for each overlay path. TFRC provides a congestion control mechanism that is more suitable to multimedia applications than TCP while at the same time remains TCP friendly. TFRC uses an equation to decide the optimal sending rate of the sender. Different equations can be used. The TCP throughput equation used in this framework is based on the TCP Reno equation from [23]

$$R = \frac{P}{t_{RTT}\sqrt{\frac{2bp}{3}} + t_{RTO}(3\sqrt{\frac{3bp}{8}})p(1+32p^2)} \quad (3)$$

where

- R is the transmit rate in bytes/second.
- P is the packet size in bytes.
- t_{RTT} is the round trip time in seconds.
- p is the loss event rate, between 0 and 1.0, of the number of loss events as a fraction of the number of packets transmitted.
- t_{RTO} is the TCP retransmission timeout value in seconds, which is set to $4R$ in practice.
- b is the number of packets acknowledged by a single TCP acknowledgment.

At the receiver side, the Decoding and Transport Module (DTM) of the Receiver-side Overlay Network Agent (RONA) decodes the packet streams received by the TFRC receivers and notifies the sender when enough packets have been collected for a certain message or a message is lost if a timeout occurs. The feedback to the sender also includes information telling the sender to what degree the redundancy in the encoding is effective so that the sender can adjust the redundancy level accordingly.

In the following section we discuss the design details of the QoS management module and the transport protocol.

C. QoS Management Module (QMM)

The available bandwidth B in terms of bytes per second of the sender is reported by the Network Measurement Module of the sender side overlay agent. The purpose of the QoS Management Module is to allocate B for each supermedia stream according to their requirements. For each supermedia stream i ($i = 1, \cdots, M$) we have the following parameters.

- Stream Priority
  For each supermedia stream, a heuristic priority number p_i is calculated and updated dynamically based on the robotic tasks of the teleoperation application. A good way to calculate p_i is using the Task Dexterity Index (TDI) as introduced in [1].
- Stream Weight
  The weight w_i denotes the possible percentage of bandwidth of this stream over the whole bandwidth. w_i is related with the type of stream. For example, a video stream should have more weight than a text stream.
- Stream Floor Rate
  f_i is the minimum rate to make the stream i workable. For some types of media, such as video, if the bandwidth available to this this type of media drops below a certain level, the media will simply not be useful anymore.
- Stream Sending Rate
  The allocated sending rate r_i of each stream is decided by the following algorithm.

If B satisfies all f_i, that is

$$B \geq \sum_{i=1}^{M} f_i \tag{4}$$

The remaining of the bandwidth is allocated according to the weight of the streams.

$$r_i = f_i + w_i \times \frac{B - \sum_{i=1}^{M} f_i}{\sum_{i=1}^{M} w_i} \tag{5}$$

If B cannot satisfy all f_i, that is

$$B < \sum_{i=1}^{M} f_i \tag{6}$$

The bandwidth is allocated according to the priority up to f_i for each stream. Sort the stream according to the priority in the descending order, we have

$$r_i = \min(f_i, \max(0, B - \sum_{j=1}^{i-1} f_j)) \tag{7}$$

Given the target rate r_i we need a packet scheduler to make sure the actual sending rate of each TM stream is r_i. We may use token bucket algorithm [24] to achieve this goal. For each TM stream i we have a token bucket B_i whose size is F_i. One token in the bucket corresponds one byte/second bandwidth the stream can use. The rate of token increase in the bucket is r_i and F_i is a number decided by w_i. F_i should at least be greater than the packet size. The stream can send a packet whose size is X if there are X or more than X tokens in the bucket, otherwise, the stream would have to wait until the bucket has the required number of tokens. Once a packet is sent out for a stream, a certain number of tokens are deducted from the corresponding token bucket. The token bucket algorithm is illustrated in Figure 3.

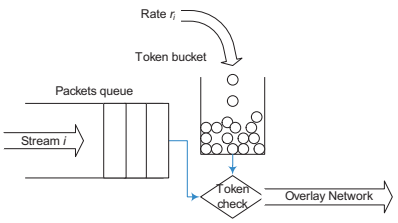

Fig. 3. The Token Bucket Algorithm

D. The Transport Protocol Design

The state transitions of the transport protocol are shown in Figure 4. The two major transport entities are the Encoding and Transport Module (ETM) at the sender side and the Decoding and Transport Module (DTM) at the receiver side. The ETM calls the path selection module when the system initializes and when a feedback is received. The path selection module produces a transport plan using the overlay network information and the feedback. According to this transport plan, ETM initializes the TFRC connections and encodes the messages to be sent. Before sending the encoded packets, ETM starts a timer. If the timer timeouts before feedback is received, ETM stops waiting and tries to send a new message using the old transport plan. The value of the timer may be specified in the form of δt, where t is the speculated transmission time found in the optimal path selection algorithm, and δ is an adjustable parameter. For certain classes of supermedia streams, a reliable transport service may be needed. In such cases, ETM will repeat transmitting the same message until a positive feedback is received.

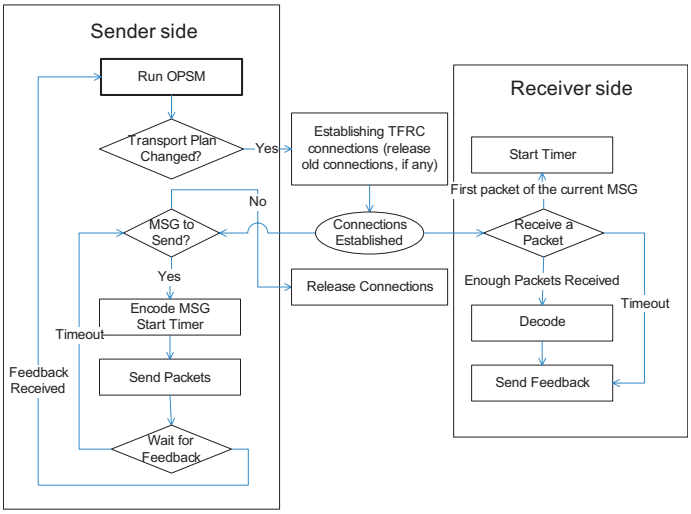

Fig. 4. The Transport Protocol State Transitions

The receiver side transport entity DTM starts a timer when the first packet of a new message is received. After enough packets were received for the current message (the number of effective packets required to decode the message is specified in the data packet), ETM decodes the message and sends a feedback to the sender. If the timer timeouts before enough packets were received, a negative feedback is sent back to the sender. Similar to the sender side timer, the value of the receiver timer may also be specified in the form of δt, where t is the speculated transmission time and δ is an adjustable parameter. However, the sender side feedback timer should be a bit longer than the receiver timer considering the single trip delay of the feedback packet.

Since the feedback only contains a small amount of data, no redundancy encoding is needed. The feedback may be sent over the primary overlay path (the overlay path with the best quality) through the TFRC connection of that path. When none of the overlay paths has satisfactory quality, the feedback may also be sent with duplicate copies over several overlay paths. Besides the existing TFRC connection, an additional TCP connection can be setup to transmit the feedback data.

III. Experimental Implementation and Simulation

In this section we present the results of using the overlay based QoS framework to accomplish an obstacle avoidance and object pickup task through teleoperation. The number of overlay paths is dynamically adjusted based on the TDI

during the teleoperation process. Also network simulator NS2 is used to simulate the behavior of the token bucket algorithm in shaping the bandwidth consumption of several supermedia streams.

A. Experiments of Teleoperation via the PlanetLab Overlay Network

Our teleoperation system mainly consists of two parts (Figure 6): the mobile manipulator (Nomad XR400 and Puma 560) and the haptic device (Phantom). The Phantom is used to send position commands to mobile manipulator and render force feedback from the mobile manipulator. Several kinds of sensors were installed in the mobile manipulator. In our experiment, laser sensors have been used to detect the obstacle and produce the TDI. In the experiment, the operator controls the mobile manipulator to pick up a material piece and place it onto the table after avoiding the obstacle in front of it. During the teleoperation experiment, TDI is produced online to reflect the dexterity of robotic tasks.

The experiment is designed to complete the teleoperation task through the PlanetLab [25](Figure 5) overlay network. PlanetLab is an overlay network used to design, evaluate and deploy geographically distributed network services. In the proposed approach for reliable and efficient teleoperation, the PlanetLab nodes serve as overlay network relay agents to transmit supermedia traffic through selected overlay routes in addition to the default IP routes determined by the routing protocols. By diverting the supermedia traffic into geographically distributed overlay paths, we expect that in the face of regional networking congestion or disruption, the alternative overlay paths will still be able to deliver the application packets to the destination efficiently.

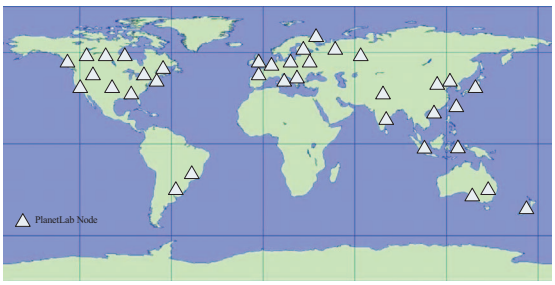

Fig. 5. The PlanetLab Overlay Network.

In the experiment we recruited five globally distributed PlanetLab nodes as overlay nodes. The number five is chosen because according to our previous experiments [4], five or six paths are generally enough to reduce the latency and its variance in most cases. More overlay paths would increase the performance marginally but the gain will be very limited. The same teleoperation task is carried out under two transport schemes 6. Both the robot and the operator are located in the United States. In both schemes an overlay node in Beijing, China (denoted as CHN) is always chosen in the paths. This is to simulate that the teleoperation task is performed between two distant locations (China and USA). In scheme (a), the operator is connected to the robot through a direct Internet path. In scheme (b), five overlay paths are served as an path pool (established by node $N0$ to $N4$ as in Figure 6. At a certain stage during the task, zero up to five overlay paths will be chosen as active overlay paths from the path pool depending on the task dexterity index. Supermedia traffic are transmitted through the default path and the active path pool based on the proposed QoS improvement approach. Periodical latency measurements are done over each path in the pool regardless of whether the path is active or not. All the overlay paths in the pool are ordered according to their QoS performance. This makes sure whenever one more active path is needed, the path with the best performance will be chosen.

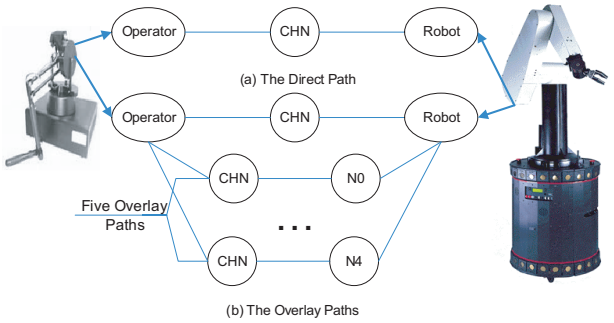

Fig. 6. The Two Transport Schemes.

Figure 7 shows the results of using six paths in the path pool (including the default path). As the TDI changes, the number of active overlay paths is automatically adjusted. As a result, the RTT tends to decrease when more active paths are used in the transmission. More obvious is that the variance of RTT is reduced significantly when the number of active paths increase. At the beginning of the experiment (before event sequence number 30), the mobile manipulator has picked up a material piece and was moving toward the destination. During most time of this period, the TDI is 0, which means the robotic task is not complex and the QoS requirement of the command stream is low. As a result only one default path of Overlay Network is allocated. When the robot meets the obstacle (starting from event sequence number 30), the operator controls the robot to avoid the obstacle. During this period, TDI is calculated to be 0.4 according to the laser sensor data. This TDI indicates that the robotic task is a little more complex compared to the robot moving task. The path number of the Overlay Network is then set to be 3 to meet the higher QoS requirement. After the robot avoids the obstacle successfully (after event sequence number 54), it moves toward the table and place the material piece on it (after about event sequence number 86). When the robot places the material piece onto the table, the TDI is calculated to be 0.8. This indicates the task becomes more complicated and the number of path increases accordingly.

The RTT comparison of using overlay paths and one direct path can be seen in Figure 8. In the overlay paths case, the RTT has an average of 0.3399 and a standard deviation of

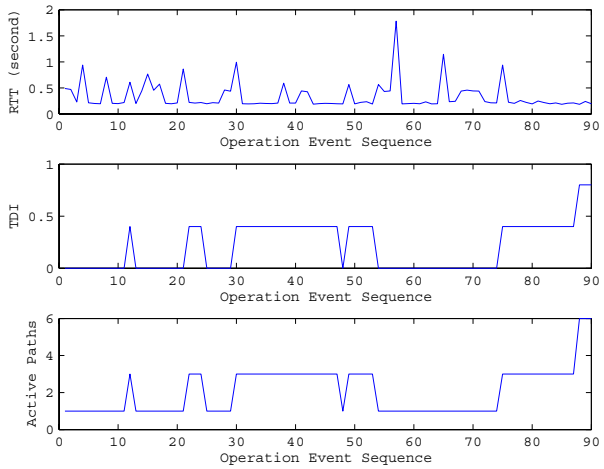

Fig. 7. The Results of Using Six Paths (including the Default Path)

0.2207. The corresponding statistics of the direct path case are 1.5462 and 0.7800. The results show that the additional overlay paths are very effective in reducing the RTT. More importantly, the variance of RTT is also reduced greatly.

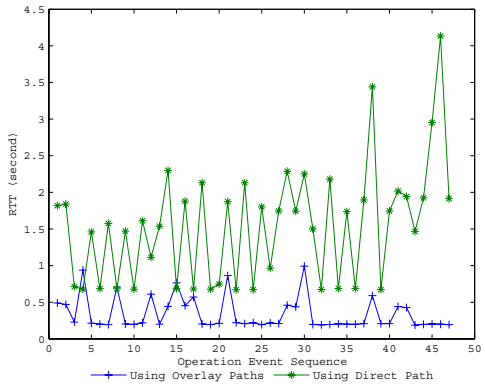

Fig. 8. RTT of the Two Schemes

The same teleoperation task is repeated three times in both direct path and overlay paths schemes. The robot base tracks (one track of each type of experiments) are shown in Figure 9. We can see that the overlay path experiment gives a smoother track than the direct path experiment. The changes of Y distance (one plot of each type of experiments) during the teleoperation process is shown in Figure 10. The experiment using overlay paths completes the task earlier than the experiment with the direct path only. The time parameters of the experiments are shown in Table I. The column "Near to Obstacle" means the time when the robot is nearest to the obstacle. The average finishing time for the direct path and overlay path is $6736.7ms$ and $3583.3ms$ respectively. The overlay path approach improves the performance by 47%.

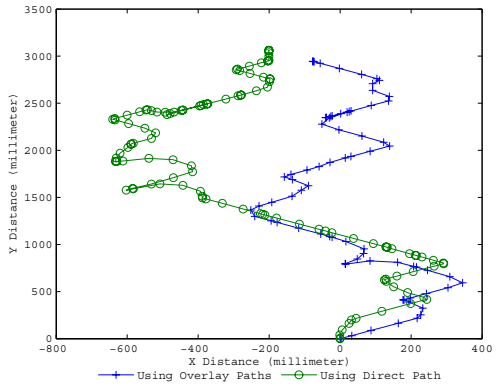

Fig. 9. Robot Base Tracks

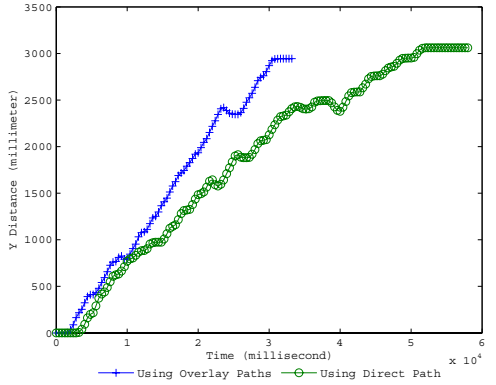

Fig. 10. Y Distance of the Robot Base Tracks vs Time

B. Simulation of the QoS Management Module

In the overlay network based transport service framework, the QoS management module is to allocate the network resource to different supermedia streams (as detailed in section II-C. To reflect the resource variance of the real network, available bandwidth traces were collected between two remote PlanetLab nodes using *Pathload* [22]. The bandwidth trace is further normalized to meet the simulation needs. The simplex-link class in NS2 is modified to enable the bandwidth input.

In the simulation scenario, three supermedia streams $s0$, $s1$, and $s2$ are scheduled to be transmitted. The weights of the three streams are 2, 3, and 5. The priorities of the three streams descend in the order and the floor rates of the streams are 10, 20, 30 KB accordingly. We can see from the simulation results (Figure 11) that the token bucket algorithm is effective in partitioning the available bandwidth according to the QoS specifications of each supermedia stream.

IV. Conclusions

This paper presents a QoS Management framework for supermedia enhanced teleoperation systems. This framework aims to provide an efficient QoS control mechanism to allocate networking resources for different supermedia streams.

TABLE I

TIME MEASUREMENTS FOR EXPERIMENTS CONDUCTED (MILLISECOND)

Experiments	Connection	Near to Obstacle	Finish
1	Direct	35000.2456	64000.4487
2	Direct	32499.728	58300.1089
3	Direct	44700.1128	79800.2612
4	Overlay	14900.2046	33700.0347
5	Overlay	13199.8926	34499.7422
6	Overlay	23900.2673	39299.9763

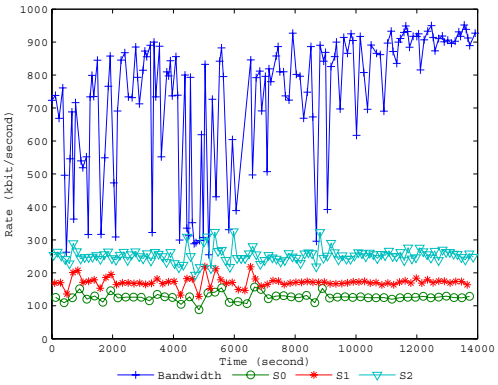

Fig. 11. Effects of the Token Bucket Algorithm

Overlay networks are used to reduce the end to end latency of teleoperation applications and hence improve the operation efficiency and stability.

The architecture of the transport service based on overlay networks is described. The architecture supports dynamically allocate overlay paths based on the task dexterity index generated according to the dexterity of the robotic task. A QoS management model is designed using the token bucket algorithm to ensure the networking resources are allocated efficiently over multiple supermedia streams. The state transitions of the transport protocol are also discussed.

Experiments using multiple overlay nodes on PlanetLab shows that the overlay network is effective in reducing the round trip time and its jitter when the dexterity of the task demands a higher quality of service. Simulations are also done to verify the effectiveness of the bandwidth allocation module of the QoS management framework. The results show that the token bucket algorithm can efficiently partition the bandwidth according to the requirement of each stream.

REFERENCES

[1] Wai Keung Fung, Ning Xi, Wang-tai Lo, BooHeon Song, Yu Sun, and Yun hui Liu, "Task Driven Dynamic QoS based Bandwidth Allocation for Real-time Teleoperation via the Internet," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003.

[2] Mark Handley, Sally Floyd, Jitendra Padhye, and Joerg Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification," *RFC 3448, Proposed Standard*, 2003.

[3] Hugh M. Smith, Matt W. Mutka, and Eric Torng, "Bandwidth Allocation for Layered Multicasted Video," in *ICMCS*, 1999.

[4] Zhiwei Cen, Amit Goradia, Matt Mutka, Ning Xi, Wai-keung Fung, and Yun-hui Liu, "Improving the Operation Efficiency of Supermedia Enhanced Internet Based Teleoperation via an Overlay Network," in *2005 IEEE International Conference on Robotics and Automation*, Barcelona, Spain, 2005.

[5] Zhiwei Cen, Matt Mutka, Danyu Zhu, and Ning Xi, "Supermedia Transport for Teleoperations over Overlay Networks," *Technical Report MSU-CSE-05-01, Department of Computer Science and Engineering, Michigan State University*, 2005.

[6] John W. Byers, Michael Luby, Michael Mitzenmacher, and Ashutosh Rege, "A Digital Fountain Approach to Reliable Distribution of Bulk Data," in *SIGCOMM*, 1998.

[7] Weidong Cui, Ion Stoica, and Randy H. Katz, "Backup Path Allocation Based on a Correlated Link Failure Probability Model in Overlay Networks," in *10th IEEE International Conference on Network Protocols (ICNP'02)*, Paris, France, 2002.

[8] Rebecca Braynard and Amin Vahdat, "Masking Failures Using Anti Entropy and Redundant Independent Paths," in *SOSP Work in Progress Presentation*, 2001.

[9] Akihiro Nakao, Larry Peterson, and Andy Bavier, "A Routing Underlay for Overlay Networks," in *SIGCOMM*, Karlsruhe, Germany, 2003.

[10] David Andersen, Hari Balakrishnan, Frans Kaashoek, and Robert Morris, "Resilient Overlay Networks," in *18th ACM Symposium on Operating Systems Principles*, 2001.

[11] Z. Li and P. Mohapatra, "QRON: QoS-aware Routing in Overlay Networks," *To appear in the special issue on Service Overlay Networks in the IEEE Journal on Selected Areas in Communications*, 2004.

[12] Zhenhai Duan, Zhi-Li Zhang, and Yiwei Thomas Hou, "Service Overlay Networks: SLAs, QoS and Bandwidth Provisioning," *10th IEEE International Conference on Network Protocols (ICNP '02)*, 2002.

[13] Bernhard Suter, T.V. Lakshman, Dimitrios Stiliadis, and Abhijit Choudhury, "Design Considerations for Supporting TCP with Per-flow Queueing," in *INFOCOM*, 1998.

[14] N.G. Duffield, T.V. Lakshman, and D. Stiliadis, "On Adaptive Bandwidth Sharing with Rate Guarantees," in *INFOCOM*, 1998.

[15] F. Zhai, R. Berry, T. N. Pappas, and A. K. Katsaggelos, "A rate-distortion optimized error control scheme for scalable video streaming over the Internet," in *Proc. IEEE ICME*, 2003.

[16] Steven Weber and Gustavo de Veciana, "Network Design for Rate Adaptive Media Stream," in *INFOCOM*, 2003.

[17] Puneet Mehra, Christophe De Vleeschouwer, and Avideh Zakhor, "Receiver-driven bandwidth sharing for TCP," in *IEEE INFOCOM*, San Francisco, USA, 2003.

[18] Pai-Hsiang Hsiao, H.T. Kung, and Koan-Sin Tan, "Active Delay Control for TCP," in *IEEE Globecom '01*, 2001.

[19] A. Ishtiaq, Y. Okabe, and M. Kanazawa, "Modified Congestion Control of SCTP over Wide Area Networks," in *Proceedings of the 2nd International Conference on Information Technology for Application (ICITA 2004)*, 2004.

[20] Rob Brennan and Thomas Curran, "SCTP Congestion Control: Initial Simulation Studies," in *Proceedings of the International Teletraffic Congress (ITC 17, Brazil)*, 2001.

[21] Long Pan, Amit Goradia, and Ning Xi, "Dynamic Multi-objective Optimal Task Distribution for Tele-Operated Mobile Manipulators," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, 2004.

[22] Manish Jain and Constantinos Dovrolis, "Pathload: A Measurement Tool for End-to-end Available Bandwidth," in *Proc. of Passive and Active Measurement Workshop*, Fort Collins, CO, 2002.

[23] Jitendra Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Throughput: A Simple Model and its Empirical Validation," in *Proceedings of ACM Sigcomm*, Vancouver, Canada, 1998.

[24] Sambit Sahu, Philippe Nain, Don Towsley, Christophe Diot, and Victor Firoiu, "On Achievable Service Differentiation with Token Bucket Marking for TCP," in *Proc. ACM SIGMETRICS'00*, Santa Clara, CA, 2000.

[25] PlanetLab, "PlanetLab: An open platform for developing, deploying, and accessing planetary-scale services," *http://www.planet-lab.org/*.