Supermedia Transport for Teleoperations over Overlay Networks

Zhiwei Cen[1], Matt W. Mutka[1], Danyu Zhu[1], and Ning Xi[2]

[1] Dept. of Computer Science and Engineering
Michigan State University, East Lansing, MI 48824
{cenzhiwe,mutka,zhudanyu}@cse.msu.edu
[2] Dept. of Electrical and Computer Engineering
Michigan State University, East Lansing, MI 48824
xin@egr.msu.edu

Abstract. In real-time Internet based teleoperation systems, the operator controls the robot and receives feedback through the Internet. Supermedia refers to robotic control commands, video, audio, haptic feedback, and other sensory information in the control system. Traditional transport services may not be able to meet the timely transmission requirements and dynamic priority changes of supermedia streams. This paper aims to design an efficient transport service for teleoperation applications. Supermedia TRansport for teleoperations over Overlay Networks (STRON) uses multiple disjoint paths besides the IP path and forward error correction encodings to reduce end-to-end latency for supermedia streams. Network routes and encoding redundancy may be adjusted dynamically to meet the supermedia QoS requirements. TFRC is used for congestion control to make STRON traffic friendly to other Internet traffic. NS2 simulations and evaluations using available bandwidth traces from globally distributed computing nodes show that STRON can significantly reduce latency compared with available transport services.

Keywords. Teleoperation, Overlay networks, Forward error correction.

1 Introduction and Related Work

Teleoperation systems allow people to control automatic systems operating at remote sites where they are inaccessible by the operators. Traditional teleoperation systems use dedicated communication channels, which have higher costs and less flexibility. In an Internet based teleoperation system the operator manipulates a control device to issue task commands to the robot through the Internet. The feedback information, including video, audio and haptic information, enables the operator to be informed of the current state of the robot. The haptic or tactile feedback relates to the sense of touch. Haptic information is captured by robot sensors and reproduced at the operator side using special devices. The Internet serves as an action superhighway instead of an information superhighway as seen by traditional networking applications.

We call all the information flowing in a real-time teleoperation system supermedia [1]. Supermedia includes video, audio, haptic, temperature, control commands and other media. Supermedia differs from traditional multimedia in that a larger variety of media are involved, and some media types are extremely sensitive to network delay. High latency may cause the robot to stall in the middle of a task or the operator to lose control of the remote robot. To approach the problem we can either modify the control mechanism of the teleoperation system to accommodate the unpredictable nature of the Internet, or try to improve the quality of service of the Internet to make it close to the dedicated communication channels used by the traditional system. Event based teleoperation systems [1] were presented to address the problem from the control perspective. 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. Extensive research efforts have been dedicated to provide QoS aware transport service over the Internet. Many research efforts aim to improve QoS levels for multimedia applications over the Internet. However these transmission mechanisms do not solve issues in supermedia transmission because a teleoperation system involves several kinds of media types, which have their own dynamic QoS requirements and some of the media types have very strict latency requirements. The purpose of the Supermedia TRansport for teleoperations over Overlay Networks (STRON) approach is to provide a fast transport service to transmit latency sensitive supermedia streams the Internet. STRON takes advantage of multiple disjoint overlay paths (one of them is the IP path) and forward error correction encodings to improve the QoS performance. The networking routes and encoding redundancy may be adjusted dynamically to meet the QoS requirements of the supermedia streams. 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.

An overlay network is composed of a set of IP-layer network paths. The end hosts of the network paths are overlay nodes. Packets may be routed among overlay nodes according to overlay link performance measurements. Since the setup of an overlay network does not require changing the underlying network infrastructure, many overlay networks are used to deploy emerging networking applications. Other overlay networks are used to improve the performance of the Internet by circumventing the inefficiencies of BGP4 in the face of link and router failures. Common overlay networks include RON [3], QRON [4], SON [5]. A Resilient Overlay Network [3] allows distributed applications to detect and recover from link failures or performance degradation, which may be much quicker than solely relying on the routing protocols. The RON nodes monitor the Internet paths among them and decide whether to allow the Internet to route the packets or relay the packets among RON nodes to achieve a better performance. In order to meet the reliable and fast transmission requirement of teleoperation systems, the STRON approach may utilize several overlay network paths supported by the overlay nodes. The use of parallel connections to increase throughput was extensively studied in the high performance computing community. Some research shows that when the network is under utilized, parallel connections can achieve fair utilization among common connections [6]. Statistics shows in large academic networks most network links are usually under utilized.

In order for the STRON system to provide an effective transport service, the overlay paths should be physically as disjoint as possible. Currently there are several research efforts related with disjoint paths selection in overlay networks. Cui uses a probability model to represent the disjoint degree of two overlay paths [7]. The Control Overlay Protocol [8] divides the overlay nodes into regions. Each region has a super node. The super node probes its subordinate nodes to collect their disjoint information. Different super nodes coordinate to provide a set of approximately disjoint paths. A routing underlay approach for overlay networks [9] introduces a routing underlay that sits between the overlay network and the Internet. This paper shows that by detouring into another AS, 1157 out of 1235 direct paths (93.7%) have at least one disjoint secondary path.

STRON uses forward error correction encoding to provide a reliable and fast transmission service. TCP provides the standard reliable transport service but it is not designed to serve the requirement of real time supermedia streams. UDP and its variant transport services, such as TCP Friendly Rate Control (TFRC [2]), Datagram Congestion Control Protocol (DCCP) and Stream Control Transmission Protocol (SCTP), are designed for multimedia transport but each has difficulty to provide reliable transport services by taking advantage of possible performance enhancements of multiple paths.

Other research work also uses overlay networks as a means to improve quality of service for multimedia applications. QRON (QoS-aware Routing in Overlay Networks [4]) is a general unified overlay network framework. OverQoS [10] is an architecture to provide QoS using overlay networks to reduce packet loss rate. OverQoS does not provide a mechanism to reduce end-to-end latency, which is essential in supermedia applications. The method presented by Nguyen et al [11] uses a *traceroute* based heuristic scheme to find redundant paths and transmit FEC encoded data over multiple paths to reduce packet loss. However, the *traceroute* approach is traffic intensive and the heuristic scheme may not always be able to find the optimal path set.

The rest of this paper is organized as follows. In section 2, the architecture of STRON is presented with a detailed design of the optimal path selection module. We discuss the simulation methodology and results in section 3. Section 4 provides the conclusions.

2 System Architecture

2.1 System Overview

The basic idea of STRON may be explained as follows. Each supermedia stream contains a series of messages generated by the teleoperation application in a variable or constant bit rate. The message is again chopped into packets with a certain size (such as the Maximum Transfer Unit or MTU) determined by the networking layer. Given p packets for a certain message that needs to be transmitted, STRON encodes the p packets into αp packets, where α ($\alpha \geq 1$) is called the stretch factor. The encoded data packets are scheduled to be transmitted over multiple overlay paths, one of which is the IP path. As soon as the receiver collects $(1+\epsilon)k$ distinctive encoded data packets, the decoding algorithm can reconstruct the original data packets. Here ϵ is called the reception overhead. The reception overhead is zero for some encoding algorithms and a small number for others. A class of erasure codes that has this property is called a digital fountain code [12]. By using a digital fountain code, the transport protocol can provide a rather reliable transport service without using acknowledgments and retransmissions. The most common digital fountain codes are Reed-Solomon codes and Tornado codes [12]. Simulation using Reed-Solomon codes shows the encoding and decoding time for 100KB data (100 1KB-sized packets) are around 0.12 seconds and 0.054 seconds (the simulation is done on an AMD Duron 1.26GHz CPU computer with 512MB memory).

The system architecture is shown in Figure 1. At the sender side, the supermedia streams are classified according to their roles in the teleoperation system by the Classifying System (CS) according to the QoS characteristics of the supermedia streams, such as Task Dexterity Index (TDI [1]). TDI describes the bandwidth requirement of the robotic task. The Sender-side Overlay Network Agent (SONA) is responsible for transporting supermedia streams. The Disjoint Path Search Module (DPSM) runs a disjoint overlay network path searching algorithm [7–9] through the connected overlay nodes. The paths found by DPSM might not be totally physically disjoint. A disjoint degree shows the correlation of the network performance fluctuation of a pair of overlay paths. The Network Measurement Module (NMM) is usually deployed in overlay nodes as a common service. In some overlay networks like RON, active measurements are launched periodically to probe the networking conditions. Some research (such as *Pathload* [13]) efforts have been dedicated to making accurate measurements of the network. The measurement results (available bandwidth r_i, single trip delay d_i and packet loss rate β_i) of the disjoint path set are fed into the Optimal Path Selection Module (OPSM).

The objectives of OPSM include: (1) find the optimal disjoint path set to be used as active transmission paths; (2) decide the amount of data sent over each active path; (3) decide the stretch factor according to the networking condition of the chosen active paths. The design of the OPSM is discussed in detail in the latter parts of this section.

The Encoding and Transport Module (ETM) encodes the supermedia streams using a stretch factor α and passes the encoded packets to the overlay transport service. The stretch factor α is found by the Optimal Path Selection Module. The encoding algorithm may be any of the digital fountain codes, such as Reed-Solomon codes [14] and Tornado codes [12]. 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 mul-

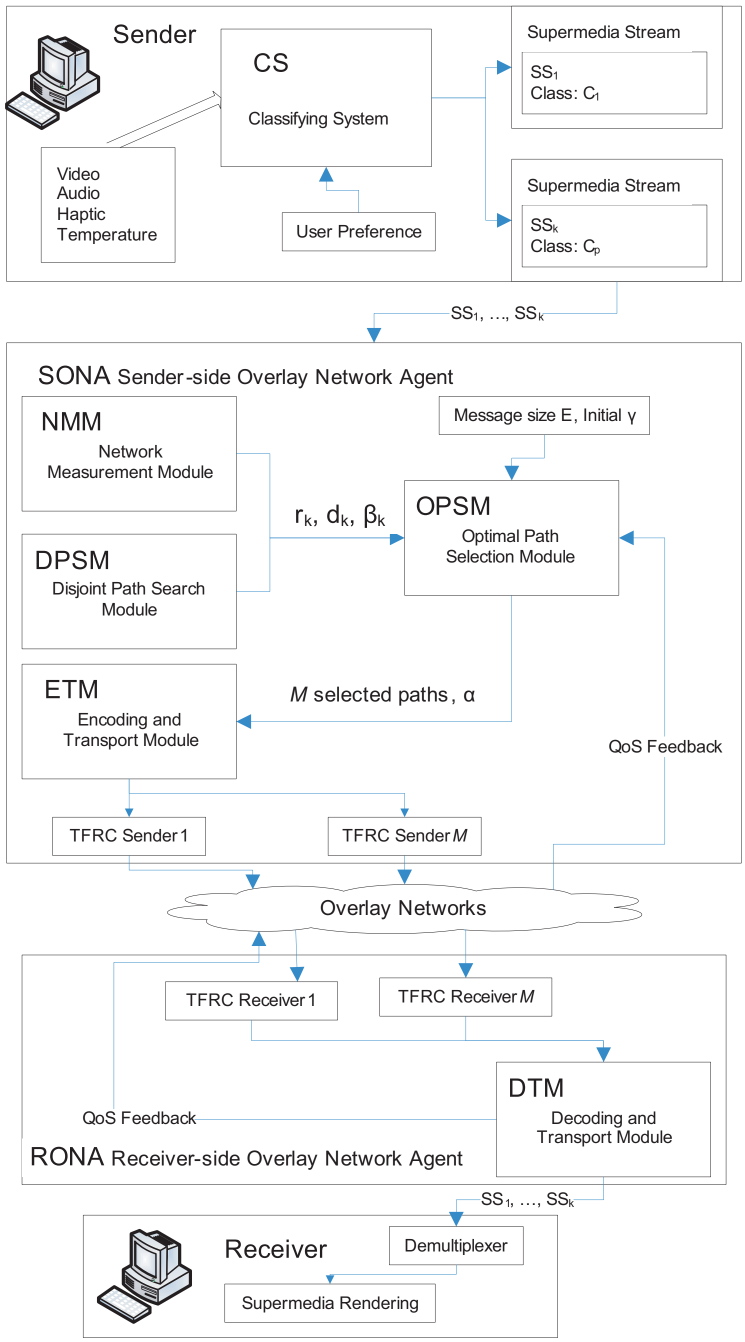

Fig. 1. The Architecture of STRON

timedia applications than TCP while at the same time remains TCP friendly. When receiving a signal from the receiver saying enough packets are received for decoding, the sender will stop sending packets. However during this period there will be still some packets on the wire. These packets are the traffic overhead of the STRON system, of which we will see the details in the simulation.

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 Optimal Path Selection Module and the transport protocol.

2.2 Optimal Path Selection Module (OPSM)

The OPSM receives input from NMM and DPSM specifying the available disjoint overlay paths and the quality parameters of each path. The OPSM consists of two parts: the path selection submodule and the γ adaption submodule.

The Path Selection Submodule We formulate the problem of optimal path selection as follows. Suppose N disjoint (or semi-disjoint) overlay paths were found in DPSM. For each path k ($k \in [1, N]$), the following parameters are given: the single trip delay d_k in terms of seconds, the average throughput r_k in terms of bytes per second, and the packet loss rate β_k. The size of each message, which is the amount of effective data we need to transmit from the sender to the receiver, is E in terms of bytes.

We need M disjoint paths out of the N given paths to minimize the latency between the sender and receiver. For each path i ($i \in [1, M]$) of the M paths, we also need to calculate V_i, which is the amount of data injected into this path by the sender. The system redundancy coefficient γ indicates the redundancy the system has over unexpected packet loss. Assume E_i is the effective data gathered from path i that is used in the data construction process, and we have

$$E_i = \frac{V_i(1-\beta_i)}{\gamma}. \tag{1}$$

Assuming t is the time for the receiver to receive and reconstruct the original data, we have

$$V_i = (t - d_i)r_i \tag{2}$$

According to (1)

$$E_i = \frac{r_i(t-d_i)(1-\beta_i)}{\gamma} \tag{3}$$

We have

$$\sum_{i=1}^{M} E_i = E(1+\epsilon) \tag{4}$$

where ϵ ($\epsilon \in [0,1)$) is the reception overhead of the digital fountain code, or

$$E = \frac{\sum_{i=1}^{M}[(t-d_i)r_i(1-\beta_i)]}{(1+\epsilon)\gamma} \tag{5}$$

Solving t yields

$$t = \frac{E(1+\epsilon)\gamma + \sum_{i=1}^{M} r_i(1-\beta_i)d_i}{\sum_{i=1}^{M} r_i(1-\beta_i)} \tag{6}$$

To solve the problem, we may try set $s = \{(i_1, i_2, \cdots, i_M) | i_p \in [1,N], p \in [1,M],$ and for any $p \neq q, p \in [1,M], q \in [1,M], i_p \neq i_q\}$, which is a combination of set $[1,N]$ to minimize (6), which is

$$t = \frac{E(1+\epsilon)\gamma + \sum_{j\in s} r_j(1-\beta_j)d_j}{\sum_{j\in s} r_j(1-\beta_j)} \tag{7}$$

The most straightforward method is to enumerate all the $\binom{N}{M}$ sets to find the subset s of $[1,N]$ that minimizes (6). When N is small, which is true in most cases, this method works well.

The volume that needs to be sent over a selected disjoint path i, which is V_i, may be found by using (2). The stretch factor α of the digital fountain encoding may be calculated as

$$\alpha = \frac{\sum_{i=1}^{M} V_i}{E} \tag{8}$$

A *transport plan* consists of a series of active overlay network paths, QoS parameters of these paths, the redundancy factor γ, stretch factor α and the number of packets (or volume of data) to be sent over each active overlay path.

The γ Adaptation Submodule The redundancy coefficient γ is a user-defined parameter specifying the amount of redundancy to be used. The value γ influences the efficiency and reliability of the system and should be adjusted dynamically when the quality of the overlay paths changes. With larger γ, the transport service is more reliable. However, larger γ requires more encoding and decoding overhead. Figure 2 shows the relationship of γ and the successful rate.

In the first case, both overlay paths were using an MMPP error model (detailed description of the simulation methodology may be found in Section 3) with an expected packet loss rate of 4%. We see that $\gamma = 1.2$ ensures the transport to be 100% reliable under this network condition. In the second case, both overlay paths used an MMPP error model with an expected packet loss rate of 1%. A smaller γ was able to ensure that the transport is 100% reliable.

From Figure 2 we can see it is difficult for the user to specify the redundancy coefficient, which should be adjusted dynamically. We need an automatic

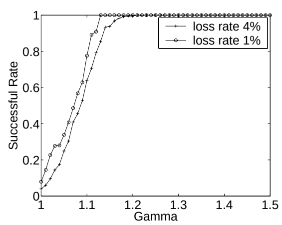

Fig. 2. Effects of Gamma over Successful Rate

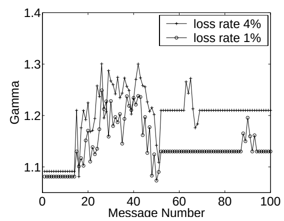

Fig. 3. Automatic Gamma Adaptation

```
bool gamma_changed = false;
if ((uf_rate<uf_floor||
target_suc_rate-actual_suc_rate
      >diff_allowable)&&
    gamma<gamma_roof) {
    increase gamma;
    gamma_changed=true;
} else if (uf_rate>uf_roof&&
    gamma>gamma_floor){
    decrease gamma;
    gamma_changed=true;
} if(gamma_changed)
    run path selection algorithm;
```

Fig. 4. γ Adaptation Algorithm

γ updating algorithm. Figure 4 shows the gamma adaptation algorithm. The variables `actual_succ_rate` and uf_rate are feedbacks received from the receiver by the sender. The target successful rate of this supermedia stream is denoted as `target_succ_rate`. The variables `gamma_roof` and `gamma_floor` are the minimal and maximal value of γ. The variables `diff_allowable`, `uf_floor` and `uf_roof` will be explained in the following paragraphs.

The variable `actual_succ_rate` is the ratio of the messages successfully delivered by the system up to the present. A message is regarded as delivered successfully if enough packets are received by the receiver before a timeout occurs and the original message is successful decoded by the decoder. The variable uf_rate means "under-full rate." The under-full rate is calculated by the receiver after a message is successfully decoded, otherwise the under-full rate is set to -1. The definition for uf_rate is

$$uf\_rate = \frac{(AllPkts - PktsSent) \cdot \frac{PktsReceived}{PktsSent}}{AllPkts} \tag{9}$$

where $AllPkts$ is the total number of packets after the message is encoded and may be calculated as

$$AllPkts = \lceil \alpha \cdot EffectivePkts \rceil . \tag{10}$$

α is the stretch factor and $EffectivePkts$ is the original message packets.

Upon decoding, $PktsReceived$ should equal $EffectivePkts$. $PktsSent$ is the number of packets sent by the sender up to now. The receiver obtains this information from the last packet received. The variable uf_rate shows the percentage of the packets that would have been useless if they were received successfully by the receiver. Thus, uf_rate is a parameter that shows how much redundancy was "wasted" during previous transmissions. To keep the system stable, uf_rate is updated in the following way:

$$uf\_rate_{new} = uf\_rate_{old} \cdot \alpha_{uf} + uf\_rate_{last\_time} \cdot (1 - \alpha_{uf}) \tag{11}$$

where α_{uf} is a stabling parameter. If the actual successful rate of the supermedia stream is below the target successful rate by a difference more than `diff_allowable` or uf_rate is below the floor threshold and γ has not reached the roof value, γ is increased. Otherwise if the under-full rate is over the roof value (`uf_roof`) and γ is over the floor value, γ is decreased. The system adjusts the quality of service for different classes of supermedia by setting appropriate γ adaptation parameters for them. If the quality of the overlay paths is stable, the value of γ will converge and the system reaches a stable state. Figure 3 shows the working process of the automatic γ adaptation system. After a certain number of transmissions, γ tends to converge to a stable value.

3 Simulation

To evaluate the effectiveness and efficiency of the system, the ns2 network simulator was used to construct a simulation environment for the overlay network transport protocol. An Optimal Path Selection Module was implemented in the application layer of the simulator. An implementation of Reed-Solomon codes [15] was used to estimate the encoding and decoding overhead of the algorithm. We did not use Tornado codes in the simulation due to its proprietary nature. In order to simulate the behavior of the system under lossy network conditions, it is important to choose good packet loss models. A Poisson arrival process assumes the packet loss event to be independent and the loss event inter-arrival time conforms to an exponential distribution. Some measurements [16] show that the packet loss events of the Internet may be independent and fit an exponential distribution. To depict packet loss events more accurately, the Poisson process is modified into a Markov Modulated Poisson Process (MMPP). The MMPP is a doubly stochastic Poisson process whose arrival rate is given by an m-state irreducible continuous time Markov chain.

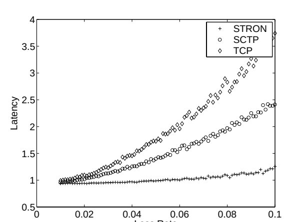

Fig. 5. Latency vs Loss Rate: STRON, TCP and SCTP (with 1 path)

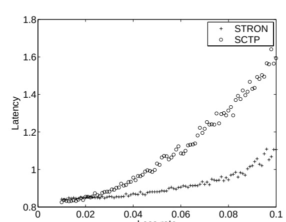

Fig. 6. Latency vs Loss Rate: STRON and SCTP (with 2 Paths)

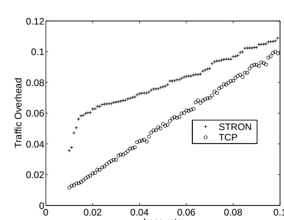

Fig. 7. Traffic Overhead of STRON and TCP

3.1 Simulation with Variable Packet Loss Rates

In the following simulations, two candidate disjoint overlay network paths were established. The slower path $path0$ has an available bandwidth of 1MB and a single trip delay of 100ms. The faster path *path*1 has an available bandwidth of 10MB and a single trip delay of 100ms. Each message contains 50 packets, each of which is 1000 bytes. In the following two subsections, we compared the performance of STRON with TCP and SCTP under variable packet loss rates. Since TCP does not support multiple transmission paths, only *path*1 is used in the simulation when TCP is involved in the comparison.

Unless explicitly specified, we used the exponential packet loss model in this simulation since the exponential model is more suitable to generate packet loss events according to a given packet loss rate. In the following simulation, the packet loss rate was increased from 0.01 to 0.1. We see from Figure 5 that while the packet loss rate increases, STRON performs better than TCP and SCTP. For example, when the loss rate is 0.05, STRON takes 0.98 seconds to transmit a 50KB message, while TCP takes 1.73 seconds and SCTP takes 1.40 seconds. An experiment with the Reed-Solomon codes shows with stretch factor 1.2, the encoding and decoding of 50KB data takes 0.014 and 0.002 seconds under the experimenting computer (an AMD Duron 1.26GHz CPU with 512MB memory). Thus overall STRON takes 0.996 seconds, with an improvement of 42% compared with TCP and 29% compared with SCTP.

Unlike TCP, SCTP is able to support multiple transmission paths. In Figure 6 two paths were used for the transmission. The path *path*0 has an MMPP packet loss model with expected packet loss rate 0.01. The path *path*2 has an exponential packet loss model with the packet loss rate increasing from 0.01 to 0.1. Although when the packet loss rate is low, SCTP performs a little better than STRON, STRON may take better advantage of the overlay paths as the loss rate increases. Figure 7 shows the traffic overhead of STRON and TCP while transmitting the same number of messages. For STRON, the traffic overhead is the redundant traffic divided by the amount of effective traffic. The redundant traffic is calculated as the actual traffic on the wire minus the effective traffic. For TCP, the traffic overhead is the amount of retransmitted traffic divided by the net traffic. From the figure we can see that STRON only has a slightly higher traffic overhead than STRON and the overhead of STRON tends to increase slower compared with TCP when the loss rate increases.

3.2 Simulation Using Available Bandwidth Traces

We simulated the behavior of STRON using bandwidth traces collected under PlanetLab. PlanetLab [17] is an overlay network used to design, evaluate and deploy geographically distributed network services. The available bandwidth traces were measured using *Pathload* [13] among three nodes that are located in Michigan State University (node MSU), Japan (node JPN) and Australia (node AUS), respectively. Given the heavy usage of many PlanetLab nodes, we considered the load to obtain accurate traces. Furthermore, *Pathload* has important features to deal with context switches and interrupt coalescence at receiver nodes. Even if many PlanetLab nodes are heavily loaded most of the time, the simulation using the traces show that STRON performs the same or better in a less stressed networking environment.

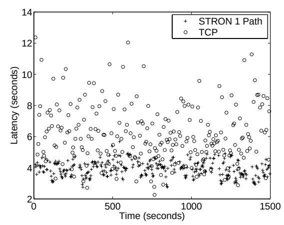

Fig. 8. Comparison of STRON and TCP Using BW 1

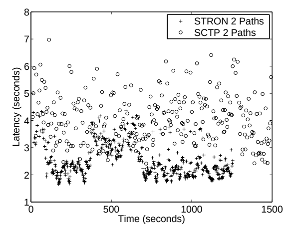

Fig. 9. Comparison of STRON and SCTP Using BW 1 and BW 2

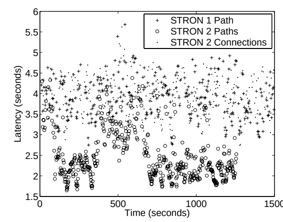

Fig. 10. STRON under Different Conditions

The measurements were done among the three nodes at the same time. One *Pathload* probe started at the beginning of every minute. $BW1$ was the available bandwidth between node MSU and AUS. $BW2$ was the minimum of the available bandwidth of the path between MSU and JPN, and the path between JPN and AUS. $BW2$ can be regarded as the available bandwidth series of an overlay path between MSU and AUS that traverses JPN. Results of *traceroute* show that the paths of MSU-AUS and MSU-JPN-AUS are quite disjoint.

We modified the simplex-link object in ns2 to enable the bandwidth input. A series of supermedia messages were sent from node MSU to AUS. Each message has 250 original packets of size 1000 bytes. An MMPP packet loss model with expected packet loss rate 1% was assumed on all paths. Figure 8 shows the latency variance when the messages were sent over the direct connection using the available bandwidth series $BW1$. STRON achieves better performance and has less variance compared with TCP. Figure 9 shows the situation when both the direct path and overlay path are used by STRON and SCTP. In the simulation of Figure 10, we established two TFRC connections under the direct path (with available bandwidth $BW1$). The results show that two TFRC connections under one path do not necessarily decrease the latency compared with one TFRC connection under one path. However, a second overlay path can improve the performance of the system under the same packet loss conditions.

4 Conclusions

Designing a transport service for an Internet based teleoperation system is challenging because of diverse media forms in the application and the timely transmission requirement of supermedia streams. The end-to-end latency of the communication channel is crucial for the teleoperator to send commands to the robot and receive haptic feedback from the robot, both of which are important for the operator to maintain smooth control of the robot. The STRON system is built to reduce the end-to-end latency and improve other QoS parameters for teleoperation systems.

Multiple overlay network paths and forward error correction encoding are used to improve reliability and efficiency. Automatic redundancy adjustment can update the redundancy level to ensure the right amount of redundancy is used for reliable transmission. An optimal path selection module can adjust the transmission paths according to dynamic network conditions.

The results of the simulation demonstrates that the supermedia transport system performs well under different network conditions without introducing too much traffic overhead. In the case of network paths with heavy packet loss rates, it has a significant improvement over TCP and SCTP.

References

1. Fung, W.K., Xi, N., Lo, W.t., Song, B., Sun, Y., Liu, Y.h.: 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. Handley, M., Floyd, S., Padhye, J., Widmer, J.: TCP Friendly Rate Control (TFRC): Protocol Specification. RFC 3448, Proposed Standard (2003)
3. Andersen, D., Balakrishnan, H., Kaashoek, F., Morris, R.: Resilient Overlay Networks. In: 18th ACM Symposium on Operating Systems Principles. (2001)
4. Li, Z., Mohapatra, P.: 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)
5. Duan, Z., Zhang, Z.L., Hou, Y.T.: Service Overlay Networks: SLAs, QoS and Bandwidth Provisioning. 10th IEEE International Conference on Network Protocols (ICNP '02) (2002)
6. Hacker, T.J., Noble, B.D., Athey, B.D.: The Effects of Systemic Packet Loss on Aggregate TCP Flows. Conference on High Performance Networking and Computing (2002)
7. Cui, W., Stoica, I., Katz, R.H.: 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. Braynard, R., Vahdat, A.: Masking Failures Using Anti Entropy and Redundant Independent Paths. In: SOSP Work in Progress Presentation. (2001)
9. Nakao, A., Peterson, L., Bavier, A.: A Routing Underlay for Overlay Networks. In: SIGCOMM, Karlsruhe, Germany (2003)
10. Subramanian, L., Stoica, I., Balakrishnan, H., R.H.Katz: OverQoS: Offering Internet QoS using Overlays. Proc. HotNet-I Workshop, October 2002 (2002)
11. Nguyen, T., Zakhor, A.: Path Diversity with Forward Error Correction (PDF) System for Packet Switched Networks. In: IEEE INFOCOM. (2003)
12. Byers, J.W., Luby, M., Mitzenmacher, M., Rege, A.: A Digital Fountain Approach to Reliable Distribution of Bulk Data. In: SIGCOMM. (1998)
13. Jain, M., Dovrolis, C.: Pathload: A Measurement Tool for End-to-end Available Bandwidth. In: Proc. of Passive and Active Measurement Workshop, Fort Collins, CO (2002)
14. Rizzo, L.: Effective Erasure Codes for Reliable Computer Communication Protocols. Computer Communication Review (1997)
15. Blomer, J., Kalfane, M., Karp, R., Karpinski, M., Luby, M., Zuckerman, D.: An xor-based erasure-resilient coding scheme. Technical report, International Computer Science Institute, Berkeley, California (1995)
16. Yajnik, M., Moon, S., Kurose, J., Towsley, D.: Measurement and Modeling of the Temporal Dependence in Packet Loss. Proceedings of IEEE INFOCOM'99 (1999)
17. PlanetLab. (http://www.planet-lab.org/)