Improved Transport Service for Remote Sensing and Control over Wireless Networks*

Zhiwei Cen[1], Matt W. Mutka[1], Danyu Zhu[1] and Ning Xi[2]
[1]Dept. of Computer Science and Engineering
[2]Dept. of Electrical and Computer Engineering
Michigan State University
East Lansing, MI 48824, USA

Abstract— **In a bilateral teleoperated system, the signal transmissions between the operator and the slave manipulators have different QoS requirements in comparison to traditional network traffic. Running teleoperated systems over wireless networks poses more challenges in comparison to wired networks. The media streams involved differentiate themselves from other media types in that they require both reliable and smooth delivery. Reliable delivery requires the transport service to have TCP style semantics. By being smooth, the transport service should be able to deliver the control and sensing data with bounded and reduced latency and its variation. For example, we have conducted numerous teleoperated experiments using our system. We have found in some of our applications that if the end-to-end latency variance becomes larger than 0.3 second, the operator has difficulty maintaining smooth control of the slave manipulator. However, our simulations show that using TCP, the end-to-end latency variance can be as much as 2.5 seconds in an ad hoc wireless network. This paper proposes an improved Transport service for Remote Sensing and Control (TRSC). The service reduces the end-to-end latency and latency variance (jitter) for real-time reliable media in mobile ad hoc networks by using forward error correction encoding and multiple network paths. Simulation using NS2 shows the approach performs well under different wireless scenarios.**

I. Introduction

Mobile Ad hoc Networks (MANETS) are becoming a supporting framework for applications that enable people to acquire and disseminate information freely. Remote control and sensing are another class of applications that may find their way to wireless networks. A typical application that uses remote control and sensing is a bilateral teleoperation system [1]. In such a system, a human operator controls a mobile manipulator (robot) to perform certain tasks and receives force and other feedback from the remote environment. The manipulator may be located at a remote location. An IP based network is an ideal candidate for the communication channel between the operator and the robot due to its flexibility and interoperability. IP based MANETs further make a teleoperation system more resilient and adaptive to different environments, such as battle fields or emergency rescue scenarios. In the scope of the military, the ability of a soldier to control the fighting devices remotely and sense the remote environment is one of the important objectives of Future Combat Systems (FCS). In other areas, such as intelligent surveillance or navigation, hazardous materials handling, remote control and sensing through wireless network is also a critical component of the system.

A typical MANET based teleoperation system may be illustrated as in Figure 1. A multi-functional robot is controlled by a human operator through the ad hoc wireless network. The operator issues control commands to the robot and receives feedback information from the robot. The feedback information includes audio, video, haptic and temperature, etc. The working environment of the robot may be deployed with a number of sensors, which may be video cameras, recorders, infrared sensors, laser sensors, temperature sensors, etc. In a mobile surveillance network, mobile computer nodes equipped with multiple functional sensors may recognize, characterize and track certain objects. Information of the target objects in the form of image, binary data and even audio and video need to be exchanged. The mobile nodes also need to coordinate the tracking movements in order for the target to be covered by at least one of the nodes. The information exchange is also time sensitive depending on the movement features of the target.

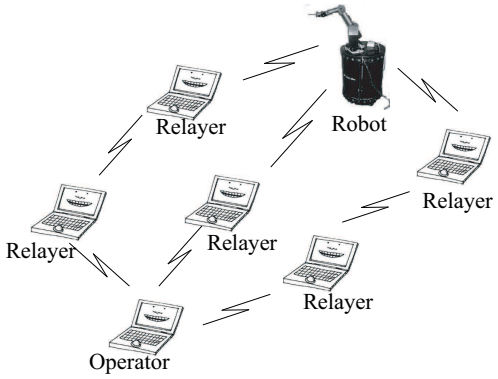

Fig. 1. An Teleoperation System Over Mobile Ad hoc Networks

The media streams involved in these applications may be classified into two groups. The first group, which includes video and audio, features traditional multimedia streams. The second group, which includes control commands and binary feedback (force, temperature etc.), is described as *Real-time Reliable Media* or RRM.

RRM differs from traditional non-real-time reliable data. The typical non-real-time reliable data are FTP or HTTP traffic supported by TCP. TCP serves the non-real-time reliable transport

*This research was supported in part by NSF Grant No. 0334035.

service well with a conservative congestion control scheme. However, the AIMD (Additive Increase and Multiplicative Decrease) nature of TCP causes undesirable jitter for RRM applications in the face of congestion. The segment based acknowledgment scheme may also increase end-to-end latency, which may impact the performance of RRM applications. Under wireless environments, the impact will become worse.

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. However, each has difficulty providing reliable transport services, which is crucial to RRM. RTP and RTCP provide support for real time application controls but they lack the congestion control and error recovery mechanisms that are essential for RRM applications.

We conducted a series of teleoperation experiments using wired networks. These experiments gave us insight of the QoS requirements of the RRM streams. In our experiments, we controlled a mobile robot based upon Nomad XR400 and Puma 560 controllers using the Phantom desktop haptic device. Using our teleoperated system, if the end-to-end latency standard deviation becomes larger than 0.3 second, the operator would have difficulty maintaining smooth control of the slave manipulator. However, in an ad hoc network, the latency variance may be much larger under TCP. Using NS2, we simulated a MANET of 50 mobile nodes equipped with Orinoco 802.11b network cards with data rate of 11 Mbit/sec and a transmission range of 60 m, which are scattered across a $400\ m \times 400\ m$ rectangular area. A modified version of the Random Waypoint Model [3] was used to determine the initial position and the moving patterns of these mobile nodes, with one second pause time and $4\ m/s$ as the maximal moving speed. A series of $1000\ byte$ packets were sent from a selected sender to a certain receiver to test the end-to-end latency variance. The result is that the end-to-end latency standard deviation may be up to 2.5 seconds in the simulated scenario. This is far beyond the acceptable range of a teleoperation system.

This paper aims to build an improved Transport service for Remote Sensing and Control (TRSC) over wireless networks. Multiple disjoint paths between the sender and receiver are used to improve reliability and reduce end-to-end latency. Forward error correction encodings are applied to certain media streams to further increase the resilience over packet loss. TFRC is used as the transport protocol, which provides a smoother congestion control and at the same time remains friendly to other TCP traffic. TRSC is inspired by our previous work for wired networks developed as Supermedia Transport for Teleoperations over Overlay Networks (STRON [4]). STRON was designed to improve the QoS for teleoperation applications based on wired wide area networks using overlay network paths. The target applications of TRSC are remote sensing and control applications based on MANETs. Wireless networks have peculiarities that wired networks do not have, which include link performance, node movements, and unstable topology. These wireless specific characteristics pose significant challenges for a reliable and efficient transport service. TRSC addresses these problems by introducing a resilient multi-path routing protocol and forward error correction encodings.

The rest of this paper is organized as follows. Section II covers related work. In section III, the architecture of the system is presented. We discuss the simulation methodology and results in section IV. Section V provides the conclusions.

II. Related Work

Much research has been dedicated to solve Quality of Service issues for MANETs. For example, soft QoS [5] is proposed to permit a grace period in which the QoS specifications might not be satisfied after a connection is established. In dynamic RSVP (dRSVP [6]), resource reservations represent ranges of network quality. Applications adapt to a QoS level within the requested range. Earlier MANET routing protocols, such as Ad hoc On-demand Distance Vector (AODV [7]) and Dynamic Source Routing (DSR [8]) do not support QoS. CEDAR, a Core-Extraction Distributed Ad hoc Routing algorithm [9] uses a set of ad hoc nodes, called the *core*, to establish a QoS aware route from the source to the destination. Information regarding the availability of bandwidth propagates among core nodes using a link state protocol. There are extensive research in improving MANETs QoS using multiple paths routing schemes. In QAWBA (QoS Aware Wireless Bandwidth Aggregation) [10], multiple paths on an ad hoc network are used to improve the performance of cellular link based applications. However, none of these approaches works well for RRM applications since they did not attempt to reduce latency or latency jitter.

Two notable multiple paths routing protocols over MANETs are AODVM [11] and Split Multi-path Routing (SMR [12]). AODVM is a multiple paths routing protocol based on Ad hoc On-Demand Distance Vector (AODV) Routing protocol. In AODV, duplicate RREQ packets are discarded by the intermediate nodes while only the first received RREQ message is processed. In AODVM, however, all received RREQ messages are recorded in the RREQ table at each intermediate node. The destination node sends an RREP for all the received RREQ packets. In AODVM, only the destination node can initiate sending the RREP message. This is different from AODV, where intermediate nodes can also send the RREP message. Any intermediate node that is forwarding the RREP message should keep hearing to see if one of its neighbors is sending (or forwarding) RREP also. If the node overhears one of its neighbors broadcasting an RREP message, it deletes that neighbor from its RREQ table. This ensures one node will not participate in more than one route and the routes are node-disjoint. However, the limitation of AODVM is that it cannot ensure a disjoint path is found if one exists. For example, in Figure 2, assuming we are routing from node A to node I, the algorithm cannot ensure path $(AEBFI)$ and $(AGCHI)$ to be found. In some cases, the algorithm may only be able to find a single path $(ABCDI)$.

Split Multi-path Routing (SMR [12]) is a DSR (Dynamic Source Routing) based multiple paths routing approach. Unlike DSR, intermediate nodes in SMR do not keep a route cache and do not reply to RREQ messages. This ensures the destination node receives all the possible routing messages so that it can select more paths. Another difference with DSR is that duplicate RREQ messages are not discarded by intermediate

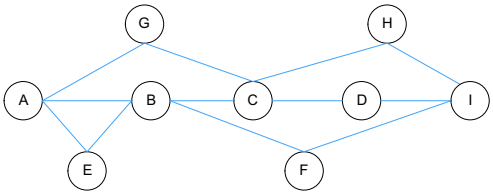

Fig. 2. An Example for AODVM

nodes. Instead, intermediate nodes forward RREQ messages that are received through different incoming links and whose hop counts are not larger than previously received RREQ messages. The first RREQ received by the receiver denotes the primary path with the shortest delay. The destination then waits to receive more RREQ messages. The path that is maximally disjoint with the primary path is selected as the secondary path.

FEC is traditionally used in the link layer to avoid unnecessary retransmissions. There are also some research about using FEC to improve the performance for general wireless transport protocol. TCP with adaptive forward error correction (TCP-AFEC) [13] uses FEC to improve the performance of the TCP connection between the mobile host and the base station. FEC is also used in end-to-end TCP connections to reduce the number of retransmissions [14]. FEC was also used by McKinley, et al. [15] to build a reliable web resource multicasting service over WLANs.

Other research aims to build a general real time transport protocol using multi-flows, but FEC is not integrated into the transport protocol [16]. Some research also deals with using multiple paths for specific video encoding schemes [17]. The work of Ma, et al. [18] uses hop-by-hop FEC check in ad hoc networks to improve the performance of multi-hop connections, however this approach imposes a large processing burden on intermediate ad hoc nodes, which is not power-optimal for mobile computing devices.

III. System Architecture

The improved transport service for RRM applications takes advantage of multiple disjoint ad hoc routes and FEC encoding to improve the quality of service. Current ad hoc routing protocols will be extended to support finding disjoint routes among the ad hoc nodes. Digital fountain [19] encoding is used to reduce the necessities of retransmissions of current transport protocols. The encoding redundancy level is adjusted automatically according to the network condition and the QoS requirement of the application. The transport protocol is built over TFRC to ensure its friendliness with other networking applications. A measurement module provides the networking parameters needed for the application.

The system consists of the following modules (Figure 3).

- Multi-path Routing Module (MRM). Responsible for finding multiple paths between the source and the destination. The paths are preferably node disjoint.
- Network Measurement Module (NMM). Responsible for measuring and reporting the QoS parameters of the paths.
- Active Path Selection Module (APSM). Responsible for selecting the optimal path set that will be used in the transmission. APSM also decides the amount of data sent over each active path and the stretch factor according to the networking condition of the chosen active paths.
- Encoding and Transport Module (ETM). Encode the transport messages and transmit the messages using TFRC connections.
- Decoding and QoS Feedback Module (DQFM). It resides at the receiver and is responsible for decoding and QoS feedback.

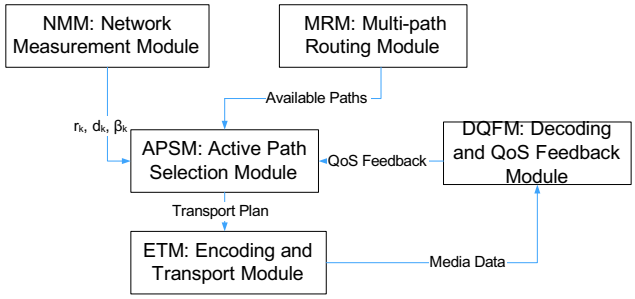

Fig. 3. Architecture of the Reliable Transport Service over Ad hoc Networks

The transport layer message for each media stream is chopped into packets with a certain size (such as the Maximum Transfer Unit or MTU) determined by the networking layer. Assuming the message for a certain stream contains p packets, ETM encodes the p packets into αp packets, where α $(\alpha \geq 1)$ is referred as the stretch factor. The encoded data packets are scheduled to be transmitted over multiple paths. As soon as the receiver collects $(1+\epsilon)p$ distinctive encoded data packets, the decoding algorithm can reconstruct the original message. Here ϵ is referred as 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 [19]. 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 [19]. Simulation using Reed-Solomon codes shows the encoding and decoding process does not introduce significant overhead under common computing environments (Section IV). Moreover, the Tornado codes can decrease the encoding and decoding time by over 100 times [19]. Tornado codes are not used in our simulation due to its proprietary nature.

A. Multi-path Routing Module (MRM)

In order to provide a reliable and efficient transport service for the RRM applications, multiple paths may be used in the ad hoc network. Compared with single path routes, multiple paths may provide fault tolerance when one routing path experiences performance degradation. By splitting the traffic over different paths, QoS parameters such as throughput, delay and performance jitter may be improved. Moreover, to ensure fault tolerance and reduce media contention, the multiple paths between the sender and receiver are desired to be disjoint. Paths can be disjoint in two senses: node-disjoint and link-disjoint.

Node-disjoint paths are more desirable in that they can be more fault tolerant and contention free.

The issue of finding multiple disjoint paths in ad hoc network can be challenging. Here we extend the SMR protocol to use it to choose multiple disjoint paths in the RRM application transport framework.

As in DSR, the sender initializes the path finding process by sending RREQ messages to its neighbors. Besides the original fields of the DSR RREQ message (the address of the initiator, the address of the target, a sequence number, and a route record), the RREQ message in MRM also contains path measurement parameters, as described in Section III-B. Unlike DSR, intermediate nodes do not keep a route cache and do not reply to RREQ messages. This ensures the destination node receives all the possible routing messages. Also duplicate RREQ messages are not unconditionally discarded by intermediate nodes. Instead, intermediate nodes forward RREQ messages that are received through different incoming links and whose hop counts are not larger than previously received RREQ messages. Since the source route is appended in each of the RREQ message, the destination thus is able to choose a group of disjoint paths according to the quality of service requirement of the application.

B. Network Measurement Module (NMM)

After MRM has found a group of available disjoint paths, it is important to decide if some paths perform better than others. Thus, we can allocate the data streams over different paths according to their performance. The QoS parameters for each path include available bandwidth r_i, packet loss rate β_i and one-way trip delay d_i.

The available bandwidth of an ad hoc link can be measured using the method presented in [10].

$$r_i = R_{capacity} - \sum_{J \in N(I)} R_{consumed}(J) \tag{1}$$

According to (1), the available bandwidth of one node I could be computed as the ad hoc network capacity minus the total consumed traffic in I's neighbor $N(I)$. The available bandwidth of a path could be computed as the minimal of the available bandwidth of the nodes the path traversed.

As for the packet loss rate, we assume that most of the packet loss events are caused by the physical layer transmission, other than packet dropping by the relay nodes. This assumption is reasonable since the relay nodes are only responsible for forwarding for a small number of nodes in a moderate sized ad hoc network. We assume that each node is capable of Signal to Interference plus Noise Ratio (SINR) measurement. If independent Rayleigh fading from symbol to symbol within a packet is assumed, we can map the SINR to a symbol error rate through some modulation analysis. For example, if the 2^m-ary symbols are transmitted with 2^m-ary noncoherently decoded orthogonal frequency shift keying, the symbol error rate is [20], [13]:

$$SER = \sum_{l=1}^{2^m-1} \frac{(-1)^{l+1}\binom{2^m-1}{l}}{1+l+lSINR} \tag{2}$$

And the frame error rate (FER), or the packet loss rate is given by:

$$FER = 1 - \sum_{i=0}^{(N-K)/2} \binom{N}{i} SER^i (1-SER)^{(N-i)} \tag{3}$$

If we assume block fading from packet to packet, similar calculations can be done accordingly. Details can be seen in [20].

If we assume the packet loss events are independent at the hops of one path, we can have the packet loss rate of a path with k hops given by:

$$\beta_i = 1 - \prod_{i=1}^{k} (1 - FER_i) \tag{4}$$

In most cases, the one-way trip delay d_i of path i may be taken as half of the round trip delay of the path. The measurement of the round trip delay of a path is straightforward. One possible way might be to piggyback the measurement information to the routing messages. Other measurement information may be passed back to the sender through the feedback mechanism, which will be discussed in the following sections.

C. Active Path Selection Module (APSM)

The active path selection module selects some of the disjoint paths found in the MRM according to the QoS parameters of these paths found in the NMM. The selected paths serve as the active transmission paths for the RRM application.

The problem of optimal path selection may be formulated as follows. Suppose N paths were found in MRM. For each path k ($k \in [1, N]$), the following parameters are given: the one-way trip delay d_k in terms of seconds, the average throughput r_k in terms of bytes/second, and the packet loss rate β_k. Each media stream is chopped into messages as a variable or constant bit rate. The size of the 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 to choose M 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 path, 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 shows how much redundancy the system has over unexpected packet loss. The process to determine γ will be discussed the following sections. Assume E_i is the effective data gathered from path i that is used in the data construction process, and we have

$$E_i = \frac{V_i(1-\beta_i)}{\gamma}. \tag{5}$$

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

$$t = \frac{V_i}{r_i} + d_i \tag{6}$$

and

$$V_i = (t - d_i) r_i \tag{7}$$

According to (5)

$$E_i = \frac{r_i(t-d_i)(1-\beta_i)}{\gamma} \tag{8}$$

We have

$$\sum_{i=1}^{M} E_i = E(1+\epsilon) \tag{9}$$

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

$$E = \frac{\sum_{i=1}^{M}[(t-d_i)r_i(1-\beta_i)]}{(1+\epsilon)\gamma} \tag{10}$$

Solving t yields

$$t = \frac{E(1+\epsilon)\gamma + \sum_{i=1}^{M} r_i(1-\beta_i)d_i}{\sum_{i=1}^{M} r_i(1-\beta_i)} \tag{11}$$

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 (11), which is

$$t = \frac{E(1+\epsilon)\gamma + \sum_{j\in s} r_j(1-\beta_j)d_j}{\sum_{j\in s} r_j(1-\beta_j)} \tag{12}$$

The most straightforward method is to enumerate all the $\binom{N}{M}$ sets to find the subset s of $[1,N]$ that minimizes (11). When N is small, which is true in most cases, this method works well. When N is large, more efficient algorithms are needed.

The volume that needs to be sent over a selected disjoint path i, which is V_i, may be found by using (7). The stretch factor α of the digital fountain encoding may be calculated as

$$\alpha = \frac{\sum_{i=1}^{M} V_i}{E} \tag{13}$$

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

We have the following example to illustrate how the path selection process works. Assume the message size is 100 packets and the packet size is 1 KB. Thus $E = 100\ KB$. We assume the redundancy coefficient γ is 2 and the reception overhead ϵ is 0.05. Assuming we have two paths to choose (shown in Table I), we have the following possible strategies.

TABLE I
TWO AVAILABLE PATHS

Path	r_k (KB/sec)	d_k (sec)	β_k
1	600	0.1	0.01
2	1000	0.2	0.005

1) Choosing path 1 only, we have $t = \frac{E(1+\epsilon)\gamma + r_1(1-\beta_1)d_1}{r_1(1-\beta_1)} = 0.4535$.
2) Choosing path 2 only, we have $t = \frac{E(1+\epsilon)\gamma + r_2(1-\beta_2)d_2}{r_2(1-\beta_2)} = 0.4111$.
3) Choosing both paths, we have $t = \frac{E(1+\epsilon)\gamma + r_1(1-\beta_1)d_1 + r_2(1-\beta_2)d_2}{r_1(1-\beta_1) + r_2(1-\beta_2)} = 0.2948$.

It is clear that strategy 3 is the best one. Given this strategy, we can calculate the V_i for each path according to (7): $V_1 = (t-d_1)r_1 \approx 117KB$ and $V_2 = (t-d_2)r_2 \approx 95KB$. The stretch factor α is $\frac{V_1+V_2}{E} = 2.1164$.

The redundancy coefficient γ is a user defined parameter specifying the amount of redundancy to be used in the system. The value of γ influences the efficiency and reliability of the system and should be adjusted dynamically when the quality of the overlay paths changes. The larger γ is, the more reliable the transport service is. However, a larger γ requires more encoding and decoding overhead. In the system implementation, γ is initialized to an empirically determined value and adjusted through the QoS feedback process (Section III-E).

D. Encoding and Transport Module (ETM)

The encoding module encodes the supermedia streams that need to be transmitted using forward error correction to ensure the quality of service. The encoded packets are allocated to the selected path and transmitted using an end-to-end transport service. One transport service candidate is TFRC since it is designed to transmit multimedia data while still remaining friendly to existing TCP traffic.

Figure 4 illustrates the mechanisms of transporting the media streams over multiple paths. Source data were encoded with stretch factor 2 and sent over two disjoint paths. Six packets were sent over path 1, which has a higher bandwidth and lower loss rate. Four packets were sent over path 2, whose bandwidth is low and the loss rate is high. As soon as the receiver collects 5 data packets, which are enough for the decoder to work, it signals the sender to stop sending packets and decodes the information. Because the system eliminates the requirement of most retransmissions when packet loss occurs, it has great advantage over traditional transport protocols.

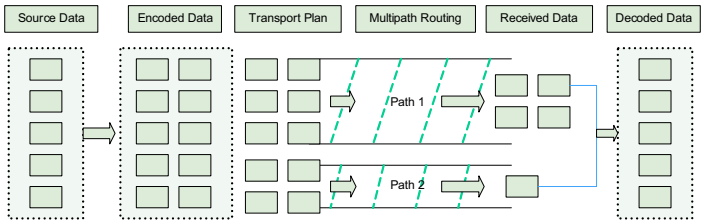

Fig. 4. An Example of Transporting Messages over Multiple Paths

The state transitions of the sender side transport protocol are shown in Figure 5. The active path selection module produces a transport plan using the network measurement information and the feedback. According to this transport plan, TFRC connections are initialized and the messages are encoded using the given redundancy parameters. As the packets are being sent, the sender starts a timer. If the timer timeouts before any feedback is received, the sender 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 active path selection algorithm, and δ is an adjustable parameter.

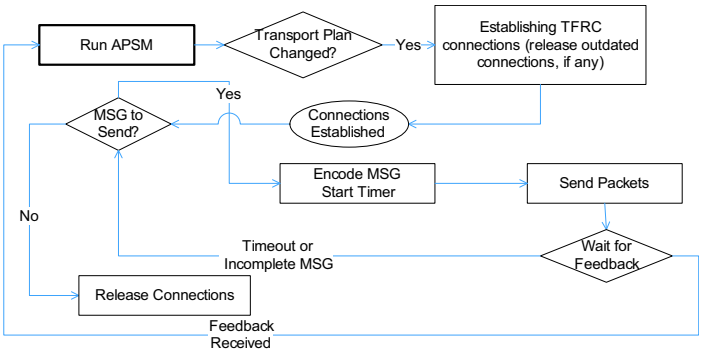

Fig. 5. The Transport Protocol Design

```
bool gamma_changed = false;
if ( (uf_rate < uf_floor ||
      target_succ_rate - actual_succ_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. 6. γ Adaptation Algorithm

The receiver side 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), the receiver decodes the message and sends feedback to the sender. If the timer timeouts before enough packets were received, a negative feedback is sent back to the sender.

E. Decoding and QoS Feedback Module (DQFM)

The Decoding and QoS Feedback Module collects the data packets from different paths and decodes them. QoS feedbacks that show the effectiveness of the redundancy encoding are sent back to the sender. A timer at the receiver side also starts when the first packet of a message is received. If the timer timeouts and the receiver has not collected enough packets, a negative feedback is sent back to the sender, otherwise a QoS feedback packet is sent back. The QoS feedback includes QoS parameter adjustments made based on the new transmission, including the redundancy coefficient γ. 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 one-way 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 path (the 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 paths.

Figure 6 shows the gamma adaptation algorithm. The target successful rate of this media class is denoted as `target_succ_rate`. The variables `gamma_roof` and `gamma_floor` are the minimal and maximal value of γ for this media class. The variables `diff_allowable`, `uf_floor` and `uf_roof` are other QoS parameters for this media class and 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 without making any retransmission attempts. 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

$$uf\_rate = \frac{(AllPkts - PktsSent) \cdot \frac{PktsReceived}{PktsSent}}{AllPkts} \quad (14)$$

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

$$AllPkts = \lceil \alpha \cdot EffectivePkts \rceil. \quad (15)$$

α is the stretch factor and $EffectivePkts$ is the original data packets of the message. Upon decoding, $PktsReceived$ should equal $EffectivePkts$. $PktsSent$ is the number of packets sent by the sender up to now.

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:

$$uf\_rate_{new} = uf\_rate_{old} \cdot \alpha_{uf} + uf\_rate_{last\_time} \cdot (1 - \alpha_{uf}) \quad (16)$$

where α_{uf} is a stabling parameter.

The γ adaptation algorithm works as the following. If the actual successful rate of the 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 media streams by setting appropriate γ adaptation parameters for them. This increases the flexibility of the system to accommodate for the QoS requirement of different real-time media streams.

IV. Simulation Methodology and Results

To evaluate the effectiveness and efficiency of the system, the NS2 network simulator was used to construct a simulation environment for the real-time media transport protocol.

An ad hoc routing protocol was extended to provide two paths between the media sender and receiver. An Active Path Selection Module was implemented in the application layer of the simulator. An implementation of Reed-Solomon codes [21] 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. The TFRC module in NS2 was modified to support having a payload. One TFRC connection was established over each chosen path in the transport plan. The receiver side of the application collected successfully transmitted packets and carried out a simulated decoding process. In the simulation, the available bandwidth r_i was set to equal to the total link bandwidth since the real-time application was assumed to be the only application running on the network. For the sake of simplicity, the packet loss and the one-way trip delay were set to fixed values without losing generality of the algorithm.

In the mobile network scenario, the mobile nodes were scattered into a $400m \times 400m$ rectangular area. A modified version of Random Waypoint Model [3] was used to determine the initial position and the moving patterns of these mobile nodes. The old Random Waypoint Model was found to fail to provide a steady state in that the average nodal speed consistently decreases over time. In the modified Random Waypoint Model, the minimal speed of the mobile nodes was set to a nonzero value (0.1 meter/second in the simulation) so that a steady state could be reached. Each node moved to a random spot within the area, rested for a certain period and then headed for another random spot, resulting in continuous changes in the topology of the underlying network. Unless otherwise stated, the pause times of the mobile nodes were set to be one second.

Each mobile node was simulated to work with the Orinoco 802.11b network card, which has a data rate of 11 $Mbit/sec$ and a transmission range of 60 m. A series of 50 messages with a size of 1000 bytes were sent from one mobile node to another during the simulation period, which was set to be 100 seconds.

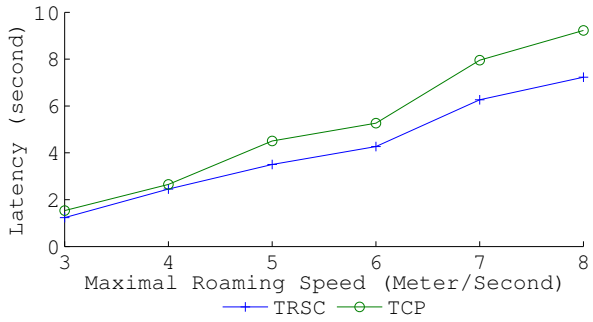

Fig. 7. Comparison of the Latency of TRSC and TCP under Different Maximal Speeds

Two types of experiments were carried out in the simulation platform. In both types of simulations, we measured the end-to-end latency of sending the specified number of messages from the sender to the receiver, as well as the goodput ratio of both of the TRSC and TCP cases. Many simulation runs were done and the results are averages of the simulations runs. For TRSC, the goodput ratio is the effective traffic divided by the total amount of traffic sent out by the sender, which is encoded using FEC. For TCP, the goodput ratio is the amount of net traffic (which is the same as the effective traffic of TRSC under the same simulation settings) divided by the total TCP traffic sent out by the sender. The total TCP traffic consists of the net traffic and the traffic retransmitted under TCP semantics. When calculating the latency of the TRSC, we took into account of the encoding time and decoding time of the packets. 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).

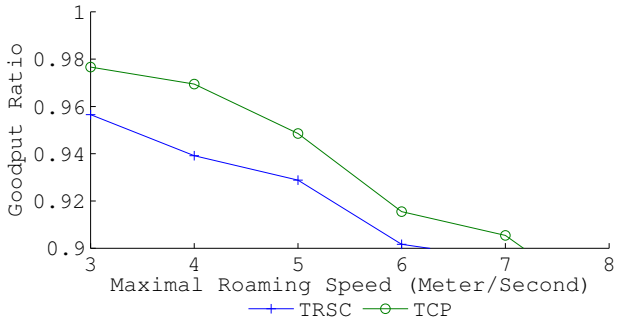

Fig. 8. Comparison of Goodput Ratio of TRSC and TCP under Different Maximal Speeds

First, we fixed the number of nodes in the simulation and changed the maximal roaming speed of the Random Waypoint Model. In the simulations, the number of nodes were chosen to be 50 and the maximal roaming speed changed from 3 m/s to 8 m/s. The end-to-end latency of sending the given number of messages using TRSC and TCP is shown in Figure 7. Of all the speed modes, the average latency for TRSC and TCP were 4.157 s and 5.19 s. TRSC had an improvement of 19.9% compared with TCP. The goodput ratio of TRSC and TCP were shown in Figure 8. The averages of the goodput ratio were 91.5% and 93.2%, respectively. We can see that TRSC reduced the latency by 19.9% but only had a 1.7% goodput ratio decrease.

Second, the maximal roaming speed was fixed to 3 m/s and the number of nodes in the simulation was changed from 50 to 100. The end-to-end latency of sending the given number of messages using TRSC and TCP is shown in Figure 9. Since the sender and receiver were the only traffic generators in the network, having more nodes increases the chance of finding a viable and good quality path between the sender and the receiver. Of all the cases with different number of nodes, the average latency for TRSC and TCP are 0.874s and 1.3 s. TRSC has an improvement of 32.7% compared with TCP. Compared with the case when only the maximal speed changes, TRSC performs even better when the number of nodes is large, in

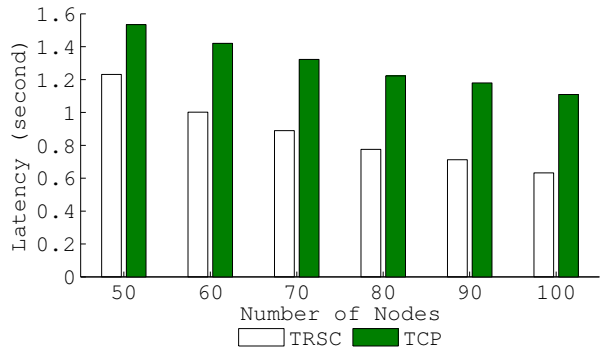

Fig. 9. Comparison of the Latency of TRSC and TCP with Different Number of Nodes

which situation TRSC can take advantage of more good quality paths. The goodput ratio of TRSC and TCP are shown in Figure 10. The averages of the goodput ratio were 96.2% and 97.9%, respectively. We see that TRSC reduces the latency by 32.7% and has a 1.74% goodput ratio decrease compared with TCP.

Due to the limitation of the computing facility, we did not do the simulation with more mobile nodes, but we can infer that TRSC works better in the scenarios where more mobile nodes are involved. However we also did some simulations for TRSC and TCP when fewer nodes are involved. It shows that as long as the number of mobile nodes is larger than 10, TRSC has significant advantages over TCP in reducing latency and latency jitter. In MANETs deployed for remote sensing and control applications, the number of nodes are often large. In those circumstances TCP often performs poor in terms of latency and latency variance due to broken routes and frequent retransmissions. TRSC, on the other hand, can reduce the latency and latency variance through the usage of multiple paths routing and forward error correction encodings.

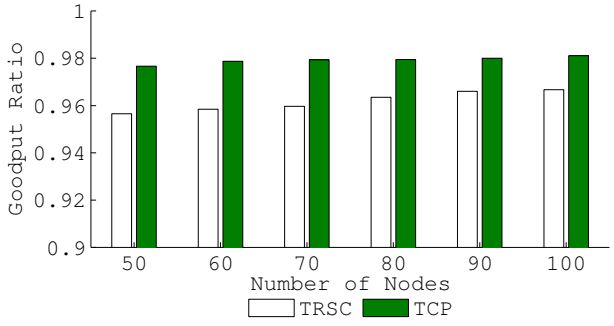

Fig. 10. Comparison of Goodput Ratio of TRSC and TCP with Different Number of Nodes

V. Conclusions

Real-time wireless applications such as teleoperation and mobile surveillance systems, require some media streams to be reliably transmitted between end hosts with the latency to be low and have little jitter. Traditional transport services either provide reliable content based transport services (TCP) or unreliable real time transport services (UDP and its variants). However none works well with Real-time reliable media (RRM). This paper proposed an improved real-time reliable transport service (TRSC) over mobile ad hoc networks. The proposed approach utilized multiple disjoint paths and forward error correction to reduce the end-to-end latency and the jitter of the latency. Simulation results show that in comparison to TCP, TRSC performs well in reducing end-to-end latency and jitter with only marginal goodput ratio decrease.

References

[1] I. Elhajj, N. Xi, W. K. Fung, Y. h. Liu, T. Kaga, and T. Fukuda, "Supermedia in internet-based telerobotic operations," in *MMNS 2001*, 2001.

[2] M. Handley, S. Floyd, J. Padhye, and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification," *RFC 3448, Proposed Standard*, 2003.

[3] J. Yoon, M. Liu, and B. Noble, "Random Waypoint Considered Harmful," in *IEEE INFOCOM 2003*, 2003.

[4] Z. Cen, M. W. Mutka, D. Zhu, and N. Xi, "Supermedia Transport for Teleoperations over Overlay Networks," in *Networking 2005*, University of Waterloo, Waterloo Ontario Canada, 2005.

[5] A. Veres, A. T. Campbell, M. Barry, and L. H. Sun, "Supporting Service Differentation in Wireless Packet Networks Using Distributed Control," *IEEE Journal of Selected Areas in Communications*, 2001.

[6] M. Mirhakkak, N. Schult, and D. Thomson, "Dynamic quality-of-service for mobile ad hoc networks," in *International Conference on Mobile Computing and Networking*, Boston, Massachusetts, 2000, pp. 137–138.

[7] C. E. Perkins, E. M. Belding-Royer, and S. R. Das, "Ad-hoc on-demand distance vector routing," *IETF Draft, draft-ietf-manet-aodv-13.txt, Work in Progress*, 2003.

[8] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad-Hoc Wireless Networks," *Mobile Computing*, pp. 153–181, 1996.

[9] P. Sinha, R. Sivakumar, and V. Bharghavan, "CEDAR: a Core Extraction Distributed Ad hoc Routing Algorithm," in *IEEE INFOCOMM '99*, 1999.

[10] D. Zhu, M. W. Mutka, and Z. Cen, "QoS Aware Wireless Bandwidth Aggregation (QAWBA) by Integrating Cellular and Ad-Hoc Networks," in *First International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QSHINE'04)*, Dallas, Texas, 2004, pp. 156–163.

[11] Z. Ye, S. V. Krishnamurthy, and S. K. Tripathi, "A Framework for Reliable Routing in Mobile Ad Hoc Networks," in *INFOCOM*, 2003.

[12] S.-J. Lee and M. Gerla, "Split Multipath Routing with Maximally Disjoint Paths in Ad Hoc Networks," *IEEE International Conference on Communications*, vol. 10, 2001.

[13] B. Liu, D. L. Goeckel, and D. Towsley, "TCP-Cognizant Adaptive Forward Error Correction in Wireless Networks," in *IEEE INFOCOM*, 2002.

[14] L. Baldantoni, H. Lundqvist, and G. Karlsson, "Adaptive End-to-End FEC for Improving TCP Performance over Wireless Link," in *ICC 2004*, 2004.

[15] P. K. McKinley, C. Tang, and A. P. Mani, "A Study of Adaptive Forward Error Correction for Wireless Collaborative Computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 9, pp. 936–947, 2002.

[16] S. Mao, D. Bushmitch, S. Narayanan, and S. S. Panwar, "MRTP: A Multi-Flow Realtime Transport Protocol for Ad Hoc Networks," in *Proceedings of the Fall IEEE Vehicular Technology Conference*, Orlando, Florida, 2003.

[17] J. Zhou, H.-R. Shao, C. Shen, and M.-T. Sun, "Multi-path Transport of FGS Video," *13th International Workshop on Packet Video*, 2003.

[18] R. Ma and J. Ilow, "Reliable Multipath Routing with Fixed Delays in MANET Using Regenerating Nodes," in *28th Annual IEEE International Conference on Local Computer Networks*, Bonn/Konigswinter, Germany, 2003.

[19] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A Digital Fountain Approach to Reliable Distribution of Bulk Data," in *SIGCOMM*, 1998.

[20] J. Proakis, "Digital Communications: Third Edition," *McGraw-Hill, New York*, 1995.

[21] J. Blomer, M. Kalfane, R. Karp, M. Karpinski, M. Luby, and D. Zuckerman, "An xor-based erasure-resilient coding scheme," *Technical report, International Computer Science Institute, Berkeley, California*, 1995.