Large Scale Video Conferencing: A Digital Amphitheater

Ladan Gharai Colin Perkins Ron Riley Allison Mankin
USC Information Sciences Institute

Abstract

In this paper we address the problem of large-scale video conferencing, both from a systems and user interface point of view. We present the architecture and current implementation of our video conferencing tool, the Digital Amphitheater, which facilitates large scale video conferencing for hundreds of participants in a natural and cohesive environment. We describe a unique user interface that aims to engender a feeling of presence, using background substitution, eliminating control functions from the screen, and allowing participants to view themselves in an amphitheater setting.

1. Introduction

Video conferencing, once a novel technology, has become a staple of modern business. It is increasing common to conduct small meetings via video conferencing, saving time and economizing on travel budgets.

A variety of conferencing systems are available: commercial products tend to derive from the H.323 standards [18] and support point-to-point conferencing only, or multipoint conferencing based around a single central server. Open source systems such as vic [13] and rat [7] are examples of the alternative light-weight sessions model [8], which rely on IP multicast and relaxed membership controls to scale to very large sessions.

Our experience with the implementation of systems based on the light-weight sessions model has led us to realize that, whilst they provide significant advantages in scaling compared to H.323 based systems, there are many aspects to the problem of video conferencing with hundreds of participants which are left unsolved.

The first and most obvious is the user interface: how to fit a large number of participants onto a single screen, in a cohesive and visually comprehendible manner? For example, assuming a screen size of 1024x1024 pixels, upto 35 QCIF (176x144 pixel) video streams can be displayed in a 5x7 grid. This implies 35 different backgrounds (as each participant is sending from a different environment), which can be visually taxing. In addition, this variation in backgrounds does not engender a feeling of cohesiveness, presence, or of a common meeting place.

Secondly, from a systems point of view, there is the issue of the end-system workload. Although, powerful systems capable of software decompression for hundreds of video streams are becoming common, they are still not the norm. A typical consumer grade desktop can perhaps decompress thirty to fifty video streams. Receiving multiple individual video streams involves a high overhead in terms of interrupts processing, depacketization, frame construction, decompression and context switching between the different processes.

We have attempted to address each of these issues in the design of a large scale video conferencing tool, which we term the Digital Amphitheater. In our design we have drawn from a number of different disciplines. We employ background removal and substitution from image processing, we utilize agent based technology to reduce network and end-system load, and we have applied principles of Human Computer Interaction to our build user interface. Although each of the techniques we have used are well established in their corresponding areas, our contribution has been to pull them together and build a prototype system.

In the following, we describe our vision for the Digital Amphitheater in section 2 and our prototype implementation in section 3. In section 4, we compare our unique user interface to other systems for large scale conferencing in section. We finish with a discussion of related work in section 5 and provide conclusions in section 6.

2. The Digital Amphitheater

Our main motivation in the design of the Digital Amphitheater was to create a digital meeting place, where hundreds of participants can meet as if at one location. This involved two design challenges: how to create a feeling of presence, such that all participants feel they are at the same location; and how to support hundreds of video streams, both from networking and end system perspectives. We first describe the concept of our user interface, then the protocol and end-system architecture needed to realize the system.

2.1. User Interface Concept

The aim of our user interface is to create a digital meeting place, an environment where participants in the meeting can feel that they are interacting with each-other; rather than using a complex teleconferencing system. We envisage an auditorium, with seating for the audience and a panel of speakers, much as one might find in a typical meeting or seminar. To implement this on a flat display, we reflect the audience, so that the participant sees a view from the "stage" showing their presence with the other audience members, but show the speaker and panelists as if viewed from the audience. Figure 1 illustrates the concept, with a mock-up we used in our early design [12].

Figure 1. A Digital Amphitheater.

In this mock-up, the images of the participants have been processed to remove their background. Each participant is seated in an amphitheater seat. The seating follows the rules of perspective, such that seats and participants become smaller as they move towards the back. The use of background substitution and natural seating provides the illusion of presence, and allows a large number of video images to be composited whilst maintaining a visually pleasing aspect.

While the participants are scattered through out the amphitheater, the speaker appears in the middle of the 'front row' amongst other 'panel' members. The speaker occupies a relatively large video frame (possibly with high frame rate) as do other panelists. Both speaker and panelists have their names written in front of them, as they would in an actual panel session.

The only controls visible are for audio, the microphone and a small scroll bar, thereby keeping the 'amphitheater' clear of clutter. The scroll bar, allows one to scroll through the names of attendees in the amphitheater. Once a user selects a name, the attendee is high-lighted, and information about the person is displayed in a pop-up window. Alternatively, information can be obtained by 'clicking' on the person's image.

There is no moderator in place, therefore it is possible for everyone to talk at the same time, although of course the result would be a difficult to understand jumble of sound. Again, our model is based on real-life conferences, where floor time is dictated by social norms. In a more futuristic version of the Digital Amphitheater we foresee the software detecting a raised hand, as a queue for requesting floor time.

2.2. Protocol Architecture

To support a large number of video streams in the digital amphitheater we adopted an agent-based approach, distributing the processing required to build the user interface throughout the network. There are several parts to the system: background substitution at the transmitter, tiling agents within the network, and user interface composition at the receiver.

Each transmitter performs the background substitution algorithm on their own video, replacing the actual background with a synthetic image supplied during session initiation. Each audience member participates by unicasting video to the closest tiling agent. The agent, in turn, tiles together all the video streams it receives, and sends the resulting stream to a multicast group. All participants join this group, receiving and displaying the combined audience video. The panelists and speaker send directly to the multicast group, thus circumventing the tiling agents.

The receivers compose the tiled audience segments, speaker and panelists into a single display. Audio is received directly via a single multicast group, since it is expected that the audio rate will be low (silence is suppressed, so there is typically only a single active audio sender). The process by which the user interface is composed, and tiling is done, is shown in figure 2.

In addition to the agent based distributed processing, control protocols are needed to announce and setup the session, enabling the participants to find the tiling agents and each other. The session can be announced using SAP [5], SIP [6], a web page or even email. The announced session has a single piece of information within it: an anycast address, which should be contacted via SIP to obtain the details needed to join the session.

On sending a SIP request to that anycast address, the routing system will ensure the response comes from the closest member of the anycast group. This will be a SIP server, co-located with a tiling agent, which will respond to this request and return both the multicast group used for the audience, and unicast address of the closest tiling agent.

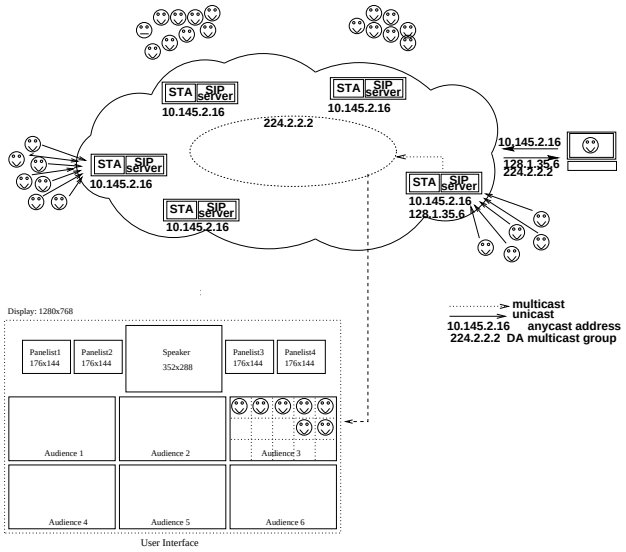


Figure 2. System Architecture

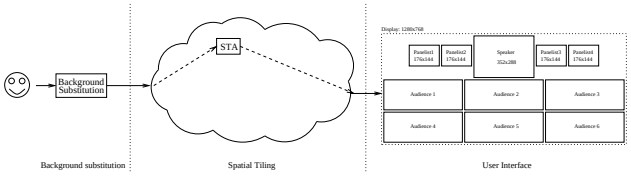


Figure 3. Flow of video through the Digital Amphitheater architecture.

A user can then participate by sending video to either the unicast address or the multicast group, respectively.

This architecture spreads the processing load throughout the network, while maintaining a simple method of joining the session.

3. Implementation

The Digital Amphitheater is implemented in three parts: background substitution at the transmitter, spatial tiling agents within the network, and user interface composition in the receiver. Figure 3 shows the flow of media data through the system, from one of the senders to one of the receivers, and illustrates this partitioning.

In the following sections, we provide more detail on the operation of the background substitution and spatial tiling, and explain how the user interface is composed from the tiled elements.

3.1. Background Substitution

An important motivation in our design of the Digital Amphitheater was providing a feeling of presence, so that all participants feel as if they are in the same location i.e, an amphitheater, a classroom or a beachside resort. Of course, this necessitate removing differing backgrounds of each participants, and substituting it with a background of choice.

3.1.1. Substitution Algorithm

The background substitution process requires an initial background image to use as a baseline for comparison. Once the camera has been positioned and adjusted for use during the meeting, the participant moves out of the field of view of the camera for a few seconds to allow the software to collect several frames of the background. These images are averaged together to provide a low-noise estimate of the background, as shown in the first frame of figure 4.

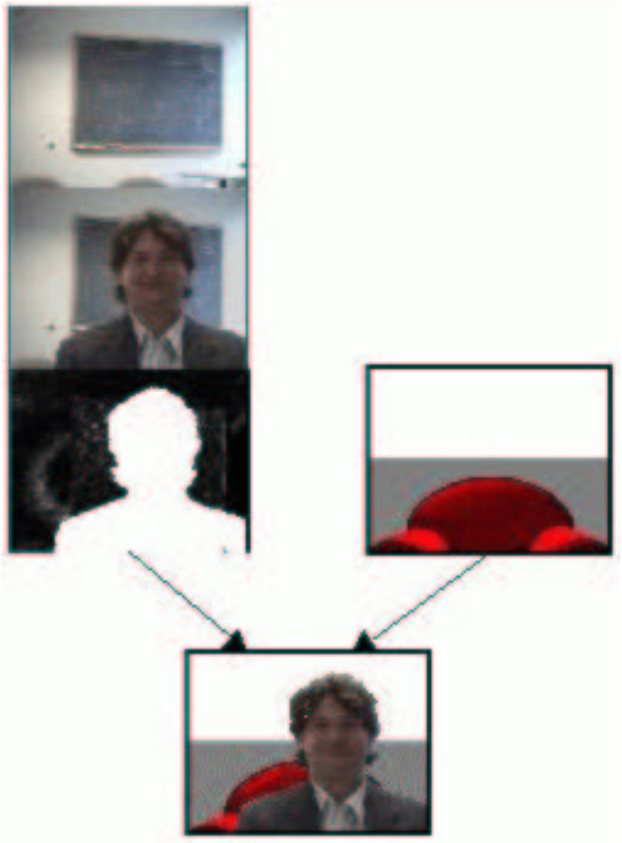


Figure 4. Background Substitution

After this brief training period, the participant returns to his seat, as illustrated in the second frame of figure 2. The

region of the current video image that has changed significantly from the background is is then segmented from the rest of the image, allowing the background to be substituted. The algorithm by which we segment the image is shown in figure 5.

A direct comparison between the current and background frames is made difficult by features common to many commodity video cameras including lighting changes, automated exposure, dynamic white balance, and increased noise. This is the reason for the scaling step in our algorithm: we compare pixels primarily on their color, but allow the apparent intensity to vary in order to compensate for changes in brightness. The resulting distances are thresholded to produce a binary mask labeling the pixels as foreground or background.

It is also possible that natural backgrounds, such as an office, contain small regions that are difficult to distinguish from the foreground. We apply morphological operators [9] to the mask to compensate for small regions of anomalous color match: the mask is eroded by a radius of two pixels to remove most of the isolated regions caused by noise in the current frame; the mask is then dilated by roughly twice the erosion radius to fill in voids (an additional erosion step may be performed, depending on the amount of noise in the image); the mask is finally eroded once more such that the total number of erodes and dilates balance to zero to restore the outer boundary of the foreground

```
clear mask
foreach (pixel) {
  find scaling between stored background and \
     current frame;
  scale current pixel colour according to exposure;
  calculate distance between current pixel and \
     stored background;
  if (distance > threshold) {
    mark as foreground in mask;
  }
}
erode and dilate mask to remove outliers and voids;
foreach (pixel) {
  if (mask[pixel] != foreground) {
    replace pixel with substitute background;
  }
}
```

Figure 5. Outline of the background substitution algorithm

The result is a low-complexity algorithm, with acceptable performance. Performance suffers when the background is subject to large changes in lighting: a more dynamic approach to updating the stored background image is planned for future versions, and should improve performance.

3.1.2. Frame Grabber Calibration

To achieve best performance from the background segmentation algorithm, it is necessary to adjust the brightness and contrast parameters of the frame grabber to match the camera, scene and lighting conditions.

The video sampling process converts analog video into digital samples in one of a range of formats (e.g. 8 bit per-channel RGB). Whilst many cameras have some limited ability to adjust the sampling parameters, it is typical that some bright points overload the convertor, causing saturation in the image. By adjusting the contrast we maximize the color range of the image whilst limiting the number of saturated pixels to a small percentage of the total. An appropriate contrast is chosen starting from a low value, gradually increasing until the 98th percentile of pixel values remain unsaturated with the camera facing a bright area of the scene.

The image brightness should be adjusted such that the "black level" of the analog video maps to zero intensity in the sampled data. To do so, we collect two frames at different brightness settings, with the camera lens covered. By linearly extrapolating the average values of these two images, we compute the brightness that will produce an image with zero intensity black.

At these optimized settings, we estimate the pattern noise of the camera by averaging 128 frames with the lens covered. The pattern noise is stored as so that it may be subtracted from each frame before comparison with the background.

Calibration is performed before starting background substitution, and helps that process. It should be noted that there is some tradeoff to be made: calibration optimizes the image for background segmentation performance, but may adversely affect the perceived color fidelity. For our purpose this tradeoff is acceptable, but other applications may differ.

3.2. Spatial Tiling: N Frames to 1 Frame

To reduce the processing load on the receivers, part of the user interface composition is done within the network, using spatial tiling agents (STAs). The concept of spatial tiling is to tile N frames next to each other, and to modify the meta-data of the tiled frame, such that it represents a single frame. In figure 6 three individual video frames are placed side by side to form a single frame. Each frame is completely represented, however the meta-data, in this case block coordinates, has been adjusted accordingly. Currently we have implemented spatial tiling for two video representations: high bandwidth raw YUV video with conditional replenishment (YUVCR) [4] and H.261 [17] using only intra-frame compression.

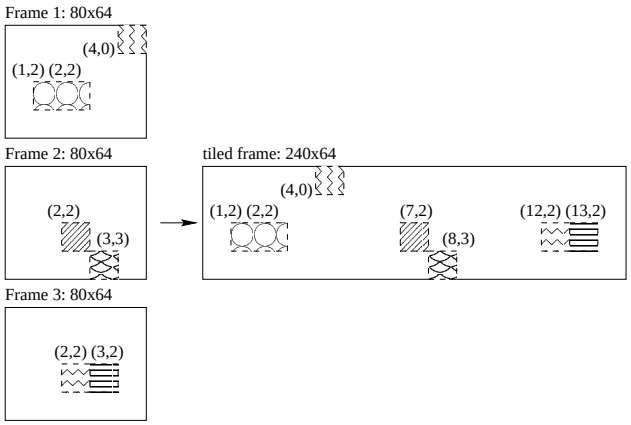


Figure 6. Tiling three frames into a single frame. Both frame size and block coordinates have been adjusted for the tiled frame.

Each audience member unicasts their video to an STA. The STA in turn tiles a number of video streams and sends the result to a multicast group, to form the audience. Therefore, it is important that the STAs do not add significant additional delay to the video stream. In our implementation, the STAs parse and deconstruct the incoming video streams into smaller building blocks whilst maintaining their relevant meta-data: no decompression is done in the STAs.

Although, theorically, it is possible to tile an unlimited number of streams, we have restricted the STAs to 15 video streams. This restriction allows us to use the built in mixer functionality of RTP/RTCP [16], since an RTP packet can carry the contributing source identifiers for up to 15 different sources. The input streams can be tiled in any geometry requested: for 15 streams the STA can generate a single row of 15x1, a square of 4x4 (where the last square will be gray), a 5x3 rectangle, or even a single column.

The STAs are implemented as software processes, running on active service hosts located within the network. They can be considered analogous to the H.323 MCU unit, although provide a very different form of media mixing and arbitration.

More details of the tiling process, and the performance benefits it brings, can be found in [3].

3.3. The User Interface

A key aspect of the digital amphitheater is its innovative user interface. This is implemented based on the *vic* video conferencing application [13] working in conjunction with rat [7] for audio.

The vic user interface has been augmented with an additional mode, which displays the speaker, four panelists, and a number of audience segments (figure 2). Due to the tiling, it is only necessary to display a small number of video streams for the audience: each stream contains a 5x3 block of 80x64 pixel frames. The user interface displays no information about each source by default: a 'tool tip' popup is used to highlight participant names and other SDES [16] information (the mixer functionality included in RTP allows for this to be conveyed along with a tiled video stream).

In present implementation, those participants which appear in the panel and as the speaker are selected command-line parameter, referenced by RTP CNAME. This provides for static selection of panelist and speaker. We plan to add the ability to dynamically chose panelists in a future version.

Audio is provided by a separate instance of the UCL Robust-Audio Tool, rat, running on a local host. Communication and coordination between audio and video – including lip-sync – is achieved via a message bus [14] interface.

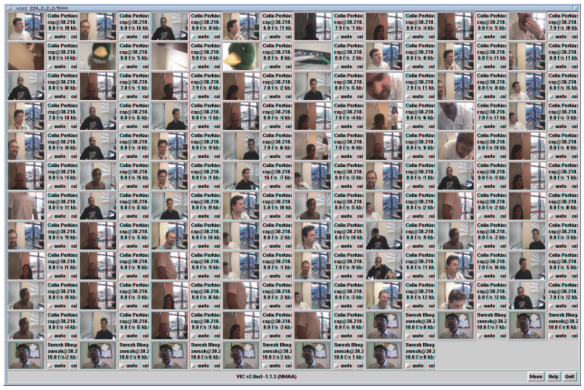
4. Discussion

Figure 7 displays various modes of video conferencing interface, with over 90 participants. Due to limitations in the number of available live participants each figure was generated using pre-recorded streams, hence the replicated participants in the audience.

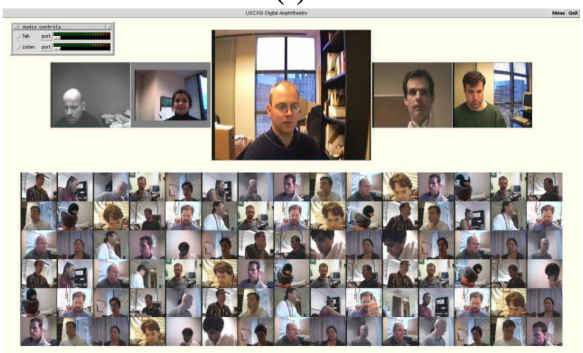
The first mode displayed, in Figure 7(a) is the standard user interface of the conferencing tool *vic* [13], a widely used multicast conferencing program. While *vic* was designed to scale to many participants, using multicast, it is clear that its user interface was not: large amounts of statistical and user information is displayed for each participant, resulting in significant screen clutter.

In Figure 7(b), we show an example of our Digital Amphitheater system running *without* the use of background substitution. The structure of the user interface interface is clearly visible – a tiled audience, with separate speaker and panelists – and hiding the extraneous controls and statistical information clearly results in a less cluttered display. There is still a lot of distracting clutter though, due to the office backgrounds, and the system as a whole does not feel like it provides a uniform environment.

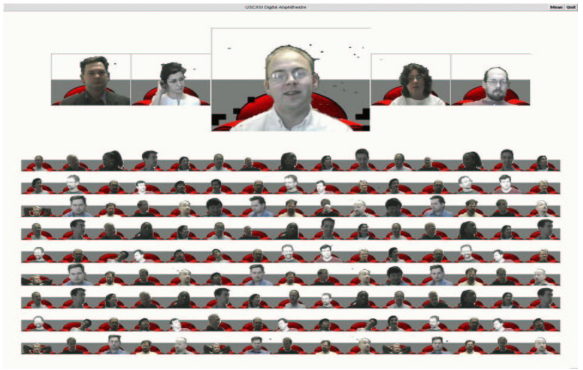
Finally, Figure 7(c) shows the complete Digital Amphitheater user interface, including the background substitution. It is very clear that the substitution of the synthetic background in place of the office clutter in Figure 7(b) results in a much cleaner interface, one that actually feels like it provides a virtual environment. Using this system is a much more pleasant experience: it is easier to concentrate on the participants, since they are clearly visible without the distraction of the varied background clutter, and there is a heightened sense of presence for users.



(a)



(b)



(c)

Figure 7. Progression of the user interface: (a) vic; (b) the digital amphitheater without background substitution; (c) the digital amphitheater with background substitution.

5. Related Work

The digital amphitheater provides a unique conferencing experience, built from a combination of innovative technologies. In creating this experience we built from a number of elements developed in earlier systems, which are noted below.

The concept of a seamless virtual environment has, of course, been widely investigated. Virtual reality systems are the prime motivator, and much of the research with avatars and virtual environments has potential to be of relevance in the future. Unfortunately, such systems are very intensive in their resource consumption, and impractical for large scale meetings at this time.

Simpler – two dimensional – virtual environment and image segmentation schemes are directly related to our work, and there have been a number of these. The work at MIT on "reflection of presence" and the ISIS toolkit [1] is closest to our system, and could provide an alternative implementation.

Also related is the MPEG-4 video coding standard [15], which provides a range of tools for scene segmentation and object-oriented scene composition. Building an interface such as our digital amphitheater would be a relatively straight forward matter if the media used MPEG-4; tiling should be possible also. Use of the MPEG-4 framework would limit our ability to work with objects other than those described within that framework, but that may be less of an issue in future. In addition, the MPEG-4 systems model is not well suited to use over IP: the SIP-based solution we propose is more flexible and scalable (e.g. MPEG-4 doesn't have the concept of an anycast invitation).

The primary benefit of our approach to image segmentation is low complexity, and the ability to work with a range of video coding systems. Our system does not, however, depend on the details of the segmentation scheme, and alternative implementations are possible.

Many H.323 based systems employ an MCU to mix audio and to act as a video switcher: video mixing and tiling is the obvious extension to that, although our agent based approach also offers advantages relating to the automatic placement and location of agents, and our agents provide scaling benefits too. It is clearly critical to the efficient operation of the STAs to optimize their placement within the network. This has received considerable attention in the literature [19, 11, 2], particularly when related to reliable multicast, leading to recent standards work in the IETF [10]. We do not seek to design new tree building mechanisms at present, rather we rely on existing work.

6. Conclusions

We have presented our vision of large scale video conferencing. To support this vision we have designed and built a large-scale video conferencing tool, we term the Digital Amphitheater. In our design we have borrowed from different disciplines: signal processing, human computer interaction, networked agents and video compression. We believe that our implementation has, so far, achieved our initial goals.

In terms of the user interface, our utilization of background substitution helps engender a feeling of presence and gives the participants the impression of meeting in the same location. A comparison of figures 7(b) and 7(c) clearly demonstrate that replacing individual participant backgrounds with a single background, does visually enhance the Digital Amphitheater.

Currently, our implementation is limited to manual session initiation. As part of our future work we will implement the SIP agents, complete with automatic session initiation. Another axes that we intended to expand the architecture along, is to add more codec capabilities to the STAs. Also, given the strategic location of the STAs, we foresee utilizing them for bandwidth and congestion control. With the RTCP feedback that the STAs receive they can detect congestion, where upon they can scale back by reducing video frame rate.

7. Acknowledgements

This work is supported by the Defense Advanced Research Projects Agency Information Technology Office.

References

[1] S. Agamanolis, A. Westner, and Jr. Bove, V. Michael. Reflection of presence: Toward more natural and responsive telecollaboration. In *Proceedings of SPIE Multimedia Networks*, 1997.

[2] P. Francis. Yallcast: Extending the Internet multicast architecture. Technical report, NTT Information Sharing Platform Laboratories, September 1999.

[3] L. Gharai, C. S. Perkins, and A. Mankin. Scaling video conferencing through spatial tiling. In *Proceedings of the 11th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2001)*, Danfords on the Sound, Port Jefferson, New York, June 2001.

[4] M. Handley. YUV-CR codec for vic. Personal correspondance.

[5] M. Handley, C. S. Perkins, and E. Whelan. SAP: Session Announcement Protocol. Internet Engineering Task Force, October 2000. RFC 2974.

[6] M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg. SIP: Session Initiation Protocol. Internet Engineering Task Force, March 1999. RFC 2543.

[7] O. Hodson and C. S. Perkins. Robust-audio tool, version 4. Software available on line. http://www-mice.cs.ucl.ac.uk/multimedia/software/rat/.

[8] V. Jacobson. Multimedia conferencing on the Internet. In SIGCOMM Symposium on Communications Architectures and Protocols, August 1994. Tutorial slides.

[9] A. Jain. Video compression. Prentice Hall, 1989.

[10] M. Kadansky, B. Levine, D. M. Chiu, B. Whetten, G. Taskale, B. Cain, D. Thaler, and W. H. Koh. Reliable multicast transport building block: Tree auto-configuration. Internet Engineering Task Force, November 2000. Work in progress.

[11] B. N. Levine, S. Paul, and J. J. Garcia-Luna-Aceves. Organizing multicast receivers deterministically by packet loss correlation. In *Proc. ACM Multimedia '98*, Bristol, UK, September 1998.

[12] A. Mankin, L. Gharai, R. Riley, M. Perez Maher, and J. Flidr. The design of a digital amphitheater. In *Proc. NOSSDAV 2000*, Chapel Hill, NC, June 2000.

[13] S. McCanne and V. Jacobson. vic: A flexible framework for packet video. In *Proc. ACM Multimedia'95*, San Franscisco, November 1995.

[14] J. Ott, D. Kutscher, and C. S. Perkins. The message bus: A platform for component-based conferencing applications. In *Proceedings of CBG2000: The CSCW2000 workshop on Component based Groupware*, Philadelphia, PA, December 2000.

[15] A. Puri and A. Eleftheriadis. MPEG-4: an object-based multimedia coding standard supporting mobile applications. *Mobile Networks and Applications*, 1998.

[16] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-time Applications. Internet Engineering Task Force, January 1996. RFC 1889.

[17] International Telecommunication Union. Video codec for audiovisual services at Px64 kbits/s. ITU-T recommendation H.261, 1993.

[18] International Telecommunication Union. Packet based multimedia communication systems. ITU-T Recommendation H.323, February 1998.

[19] R. X. Xu, A. C. Myers, H. Zhang, and R. Yavatkar. Resilient multicast support for continuous media applications. In *Proc. NOSSDAV '97*, Washington University in St. Louis, May 1997.