REAL-TIME INTERACTION OVER THE INTERNET

Christophe Salzmann and Denis Gillet

Swiss Federal Institute of Technology Lausanne - EPFL

Abstract: This paper describes the challenges to meet in order to enable effective real-time interaction over the Internet for the remote prototyping of automatic control solutions. The type of real-time interaction required for such a prototyping is the ability for an operator to adjust remotely the settings of a controlled system and to observe immediately the effects of this action on its dynamics. Solutions are proposed to keep a seamless degree of interactivity despite the Internet bandwidth variations. These solutions rely on an end-to-end adaptation scheme of the transmitted content. This scheme mimics the TCP communication protocol without showing the same drawbacks.

Keywords: Remote control, Interaction, Real-time communication, Communication protocols, Cascade control.

1. INTRODUCTION

Real-time interaction over the Internet (IoI) for the remote prototyping of automatic control solutions is a new kind of Internet service. Typical applications are remote experimentation for educational or research purposes, as well as remote diagnosis of industrial processes. While most of the classical Internet applications retrieve stored multimedia content with loose time constraints, in IoI the information representing the actual behavior of the physical system remotely operated should be provided to the user in a minimal delay. Real-time interaction over the Internet is not limited to automatic control applications. The solutions presented in this paper apply to all cases requiring a remote access to a physical entity, in particular tele-surveillance or remote experimentation in fundamental sciences, such as physics, chemistry or biology.

Among the applications of real-time interaction over the Internet, remote experimentation carried out on mechatronic systems is one of the most challenging. This is due to the fact that such systems show time constants of the same order of magnitude as the typical delay observed in Internet communication. A representative example is the remote control of an inverted pendulum (Gillet *et al.*, 2000).

The key issue in implementing real-time interaction over the Internet with physical systems is to enable the perception at distance of their dynamics. This lies in the need for two integrated features. The first feature includes graphical or haptic representations of the current states. The second is the synchronization of these representations with the actual evolution of the real system for animation purpose. Examples of predominant representations are live video images, virtual reality, augmented reality and real-time measurement signals. While providing such representations is trivial using Internet technologies, ensuring that these representations are adequately synchronized requires dedicated solutions. As mentioned earlier, another issue is the need for the operators to get feedback on actions carried out as quickly as possible. This additional constraint also requires dedicated solutions to avoid erroneous actions due to misperception of the actual system behavior.

The previously mentioned representations are introduced at the client application level. They are integrated as graphical user interface (GUI) components and typically arrange according to a cockpit metaphor. The GUI provided for the remote control of an inverted pendulum is shown in Figure 1. In addition to display areas, the GUI includes buttons and sliders for interactive tuning.

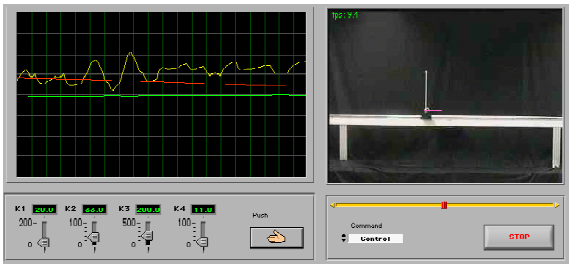

Fig 1. Cockpit-like GUI for real-time interaction over the Internet with an inverted pendulum.

To understand why dedicated solutions are necessary to enable real-time interaction over the Internet, it is important to recall that the Internet is a non-deterministic best effort network. In other words neither bandwidth nor latency can be guaranteed. The routers along the transmission path simply do their utmost to deliver the data to the receiver as fast as possible despite the variation of the available bandwidth due to the network load. It is the application responsibility to ensure that each connection sharing the network infrastructure uses a fair amount of the overhaul bandwidth. The TCP protocol (RFC793) has been designed as intermediary layer between the applications and the network infrastructure to guarantee the delivery of the sent information packets and to standardize the policy for adapting to the available bandwidth.

Real-time applications such as video streaming use buffers –located either at the sender or at the receiver side– to smooth the Internet bandwidth variation and to display images at a constant rate to the user (Feng Rexford, 1997). Since these buffers delay the delivery of information, this scheme cannot be considered to sustained interaction. Video conferencing is another real-time application that seems more similar to IoI but it carries differences: the priorities for the video and the audio are inverted. Sound is preponderant to image for video conferencing while this is not the case for IoI. There might even be not sound at all. Another difference is the scalability of the used bandwidth. IoI bandwidth usage ranges from a few bytes per second to Kilobytes per second. The lowest values cannot be considered for video conferencing due to sound quality constraints. Finally, most of the existing video conferencing software does not guarantee the principle of fair bandwidth usage. Therefore, the use of standard video conferencing software is inappropriate for efficient IoI.

Various solutions can be considered to efficiently implement real-time interaction over the Internet. A straightforward solution is to use a communication channel that can guarantee a given quality of service (QoS), such as a given bandwidth and latency, via reservation or by other means (Aktan *et al.*, 1996) This can be done by placing additional intelligence at the routers level. While this solution might be a promising one, it not only requires a widely accepted agreement among manufacturers and providers regarding new communication protocols, but also ask for expensive software upgrades for most of the already deployed routers. These solutions will not be explored in this paper.

IoI has to rely on an implementation scheme which is compatible with the current and future Internet protocols and infrastructure. We propose a software only solution implemented at the application level, at both the client and server side (Fig 2). This so called End-to-End (Floyd and Fall 1999) solution is developed to minimize the transmission delay while ensuring a seamless degree of interactivity for a varying bandwidth. This scalability is achieved using a multi-layer approach.

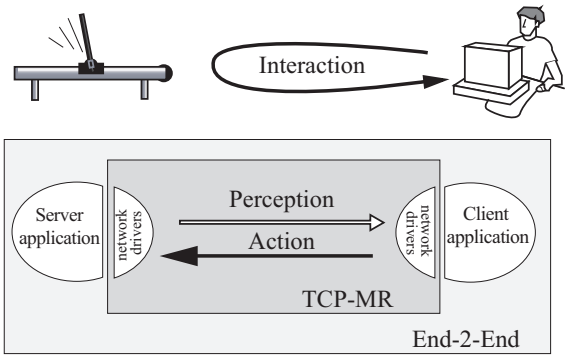

Fig 2. The multiple-layer implementation scheme to sustain interaction over Internet.

The inner layer implements a dedicated network protocol called TCP Most Recent (TCP-MR) that guarantees a signature similar to TCP necessary to ensure a fair bandwidth usage, while optimizing the usage of communication channel for interaction. In most cases the network link is the bottleneck, but the computer capabilities can also impair the quality of service. The outer layer takes into account the overhaul capabilities of the transmission channel and the networked devices to satisfy the requirement for an efficient interaction. The transmission channel typically includes the links and the routers, while the networked devices can be a PDA or a computer with its network interface and its video card.

The path between the client and the server is a chain of cascaded subsystems. Furthermore, the End-to-End

solution proposed to sustain real-time interaction over Internet relies on feedback mechanisms. It is therefore natural to apply an automatic control approach for both analysis and design.

Section 2 of this paper first analyses the TCP protocol to underline its congestion control mechanism and shows why it is not well-suited for real-time interaction. The approach to improve this mechanism for IoI is detailed in Section 3. The End-to-End control mechanism required to provide an optimal interaction is presented in Section 4.

2. TCP CONTROL MECHANISMS

Nowadays TCP is the predominant protocol used for the Internet transmission such as web page transfer (http) or file transfer (ftp). TCP provides a reliable transmission meaning that all packets sent by the server will be delivered to the client (Steven, 1994), this implying that lost packets will be retransmitted. A packet contains raw data of various sizes that can either be dynamically or statically set. Today's packet size is typically 1500 Kbytes or less, this is due to the Ethernet cell size.

TCP has a built-in mechanism designed to probe and to adapt the sending rate of the packets to the available bandwidth (Jacobson, 1988). Provided that most of the Internet traffic relies on TCP (> 90%), the bandwidth will be fairly shared among these TCP connections using the network infrastructure. However, applications can use other protocols such as UDP (RFC768) which do not have a bandwidth adaptation mechanism. If UDP-based applications do not implement their own adaptation mechanism or implement a scheme more aggressive than TCP, the fairness among TCP and UDP connections is not respected and the aggressive source gets a bandwidth share bigger than what it should have. While this behavior is tolerated in today's network policy, routers might discard transmission exhibiting such behavior in the future.

The main TCP feature is guaranteed packets delivery. TCP relies on receiver's acknowledgments to guarantee that a packet has successfully reached the destination. The acknowledgment contains a number representing the last valid number of bytes received. If this acknowledgment is not received, TCP will re-send the packet after a given retransmission timeout (RTO). The receiving of duplicated acknowledgments (DUPACK) is another indication that a packet has been lost. DUPACK occurs because TCP always acknowledges the last valid received data. When a packet is missing and a packet with newer data arrives, TCP acknowledges the last valid data until it receives the retransmitted missing data. Since the absence of acknowledgment or duplicated acknowledgments can be the sign of communication channel congestion, TCP not only retransmits the packet but also decreases its transmission rate. At the opposite, when packets are acknowledged, TCP increases its transmission rate. TCP uses the additive increase multiplicative decrease (AIMD) mechanism to adapt its sending rate. If no packets are lost the number of packets per round is increased by one. If there is a packet loss, the number of packets per round is divided by two. A round consists of the transmission of one or more packets and the receiving of their corresponding acknowledgment(s). A new round begins when the acknowledgment for the previous round is received by the server. The shortest duration of a round is the round trip time (RTT) and the processing time to compute the acknowledgment.

There is a third mechanism called slow start that limits the TCP sending rate at the beginning of the transmission or when a time out occurs during the transmission. Various TCP incarnations exist, TCP-Reno being the most commonly used. New-Reno, Vegas, SACK try to improve the TCP mechanisms described above but the global AIMD scheme remains (Brakmo and Peterson, 1995; Fall and Floyd, 1996).

The above mechanisms make TCP a robust and reliable protocol, well-suited to transfer bulk data. However, the same mechanisms make TCP inappropriate for real-time interaction. The main problem is the delay introduced by the retransmission of lost packets. By the time the re-sent packet reaches the receiver, the real-time information contained in the packet is outdated. Therefore it is more adequate for interaction applications to ignore about the lost data and send up-to-date information. Moreover the retransmitted information will be transferred at half the previous sending rate thus adding latency to the transmission.

A control protocol implanting the "Most Recent" principle to avoid re-sending outdated information useless for interaction is introduced in the next section.

3. TCP MOST RECENT PROTOCOL

To ensure equitable bandwidth share among transmissions using the same Internet infrastructure, communications are required to be TCP friendly (Chiu and Jain, 1989). This means that the bandwidth share a TCP-friendly communications can achieve in steady state has to be the same as the one achieved by TCP in the same conditions.

TCP Most Recent (TCP-MR) principle is to mimic the TCP protocol from a packet handling point of view regarding fairness, while filling the packets with the most recent available data. TCP-MR is implemented at the application level using UDP. This approach permits the estimation of the available bandwidth neces-

sary to adapt the packets content for suiting the IoI requirements. This information is not available when using “regular” TCP. The tradeoff in deploying TCP-MR is to accept possible losses in order to gain interactivity. Such a tradeoff is acceptable if the adaptation is carried out at the application level, given that the semantic nature of the transmitted information is known at that level. This means that the data to be potentially discarded can be chosen according to their priority level.

From an automatic control point of view, the TCP-MR emulation of TCP can be seen as the inner loop of a cascade structure, the content adaptation being its outer loop. This second control loop permits to take into account the application specificity and permits to better adapt to the end-to-end environment, not only to the communication channel (see next Section).

Figure 3 presents the transmission of real-time data, for example video frames, using TCP vs. TCP-MR. The upper part of the figure shows the “standard” TCP transmission while the lower part shows the TCP-MR transmission. For this example frames are generated by the server at a constant rate of 1 frame per round with a constant frame size of 16 packets. The server sending rate is slightly above the available bandwidth. The white squares represent the frames generated by the server, the line below represents the TCP received frames (plain black squares), while the TCP-MR received frames are represented by the partially filled squares in the middle of the figure.

Each column corresponds to one round. The height of the column represents the congestion window size. The packet indexes are stamped on the top of each column. A lost packet is marked with a cross. A retransmitted packet has a darker background. The *ssthresh* line represents the threshold at which congestion avoidance mechanism starts.

The TCP mechanisms regulate the number of packets per round. At the beginning the available bandwidth is not large enough to transmit one frame per round. Therefore the frame 1 is received after two rounds. When a loss occurs (pk 40, 5th round), the missing packet is retransmitted in the next round. The number of packets per round being set to half the previous one. Notice that the frame would have been completed in the previous round if the loss had occurred after the

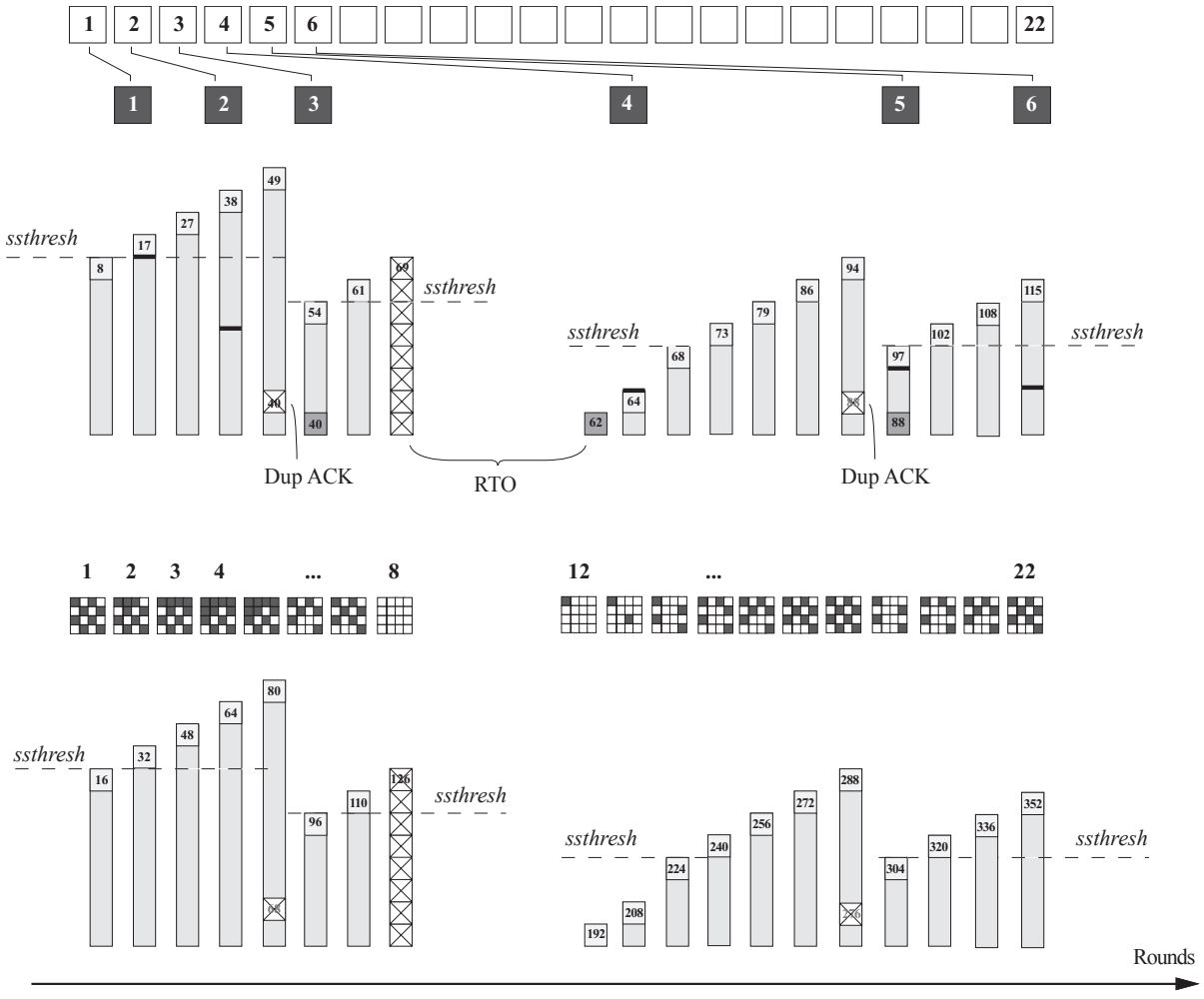

Fig 3. Comparison of frame transmission using TCP and TCP MR.

packet 48. Severe loss (round 8) detected by a retransmission timeout (RTO) triggers a slow start mechanism. The sending rate is set to one packet and it increases exponentially until the number of packets reaches half of the previous sending rate.

The TCP-MR mechanism has the same signature as the TCP mechanism but the packet content is different. The benefit is that the received frame rate is the same as the sending frame rate. Consequently the interactivity is greatly improved since each packet carries data to guarantee pacing with the actual evolution of the real system. The drawback is that the original frame content is not integrally transmitted.

TCP integrates a mechanism that refrains the sender from sending new data while the receiver has not consumed the transmitted one. This mechanism does not hold in TCP-MR since the client application will consume the incoming data as fast as possible. If for some reasons the incoming data rate is too fast, the client will inform the sender to lower its sending rate via the end-to-end control mechanism described in the next Section.

4. END-TO-END CONTROL

While the TCP-MR layer guarantees a TCP friendly signature, the End-to-End control aims at providing an optimal interaction for a given bandwidth. The End-to-End control takes into account the overhaul transmission path from the information capture to the information rendering on the client GUI. The information undergoes various transformations when flowing from the server to the client. The information is first digitized, then encoded, mixed and packetized before transiting via the communication link. At the receiver side, the data follow the inverse transformations.

4.1 Server-side operations

Depending on the nature of the information source at the server side, the encoding takes various forms. Video images can be encoded using hierarchical encoding which permits easy information discarding when fast bandwidth adaptation is required (Saha S., 2001). Measurements acquired via DAQ board can be encoded using RLE (Run Length Encoding). The encoders compression ratios change over time to follow the bandwidth variations.

The TCP bandwidth rapid variations might be too fast for the encoders to adapt. Therefore the optimizer needs to take additional actions to adapt to the abruptly changing bandwidth and to avoid adding latency to the communication (due to the bandwidth diminution). For example dropping video image high frequency coefficients, decimating the measured data or changing the stream combination lowers the bandwidth usage prior to passing the data to the TCP-MR layer. If enough bandwidth is available, data can be protected with a Forward Encoding Code (FEC) which permits the reconstruction of missing data (Bolot and Vega-García, 1998).

There is a tradeoff between the time taken by the encoder and the transmission time which is proportional to the compressed data size. Therefore the product of the compression rate and the compressed size should be carefully chosen. The same tradeoff applies when decoding the information at the client side. This can become more complex with asymmetrical encoding or when the client computer (i.e. PDA) capabilities cannot handle highly compressed data as fast as the encoding process produces them.

To minimize the effect of data loss during the transmission, each packet holds a combination of the various stream's information. This combination also permits a better rendering of the dynamic since each packet contains part of the real-time information.

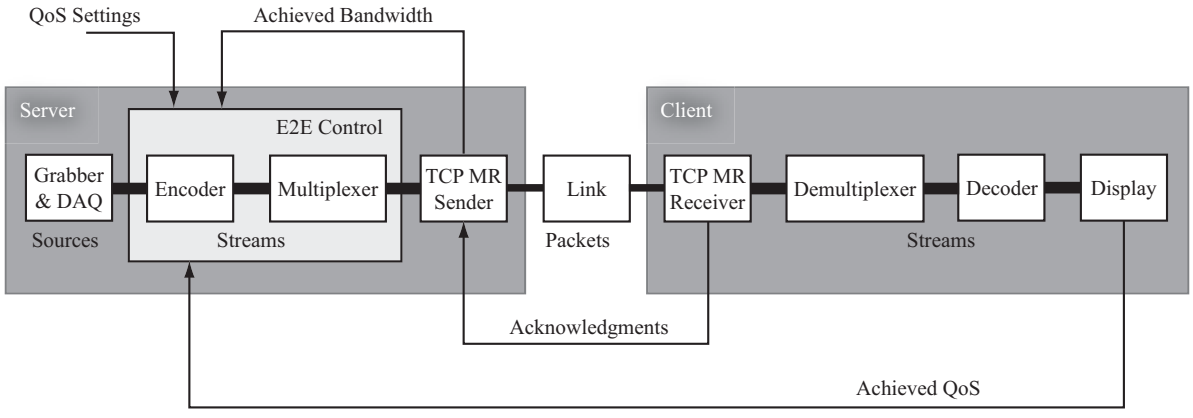

Fig 4. The cascade structure implemented to sustain real-time interaction over the Internet.

The E2E control relies on information provided by three sources: the QoS settings (reference), the achieved QoS (measurement) and the achieved bandwidth (observed state) (Fig. 4). The QoS settings specify the importance of the interaction components relatively to each other's. The latency should be kept to the minimum, therefore the user can only chooses between display rate, information rendering quality and stream priority. The achieved QoS measurement takes into account the end-to-end communication composed by the end-to-end latency, the rate at which the information is provided to the user and the quantity of received information. The achieved bandwidth is derived by the TCP-MR layer resulting from the AIMD adaptation scheme. This computation only takes into account the communication channel characteristics.

The encoding parameters and the multiplexing mechanism are adapted according to the achieved QoS and to the available bandwidth in order to optimize the interaction performances. This is currently implemented using logical rules. However, the cascade structure presented in this paper enables further implementations of classical linear or nonlinear control schemes.

4.2 Client-side operations

The client application processes and displays the received information to the user. The information processing takes various steps. First the data contained in the packet have to be demultiplexed into the various streams. Then these streams have to be decoded, this decoding stage may require missing data reconstruction. Depending on the received data the original information might have to be reconstructed (simulation), extrapolated (i.e. bigger pixel) or recovered (Carle and Biersack, 1997). Finally the information are displayed to the user interface at the appropriate time and pace.

5. CONCLUSIONS

We presented a novel scheme to provide real-time interaction over the Internet. A cascade approach is proposed to first implement a TCP friendly scheme that is required for a fair use of the available bandwidth. As opposed to "regular" TCP the proposed scheme does not retransmit outdated information but instead it retransmits the most recent one (TCP-MR). The outer loop which takes into account the End-to-End performances enables efficient real-time interaction since the semantic nature of the transmitted information is know at that level.

The enlightened cascade structure is well suited for further applications of the automatic control methodology to communication systems.

REFERENCES

Aktan B., Bohus C.A., Crowl L.A. and M.H. Shor (1996). Distant learning Applied to Control Engineering *IEEE Transaction on Education*, **39**, pp. 320-326.

Bolot J-C. and A.Vega-García (1998). The case for FEC-based error control for packet audio in the Internet, to appear in *ACM Multimedia Systems*.

Brakmo L.S. and L. L. Peterson (1995). TCP Vegas: End to End Congestion Avoidance on a Global Internet. *IEEE journal on selected Areas in Communications*, **13-8**, pp. 1465 - 1480.

Carle G. and Biersack E.W. (1997). Survey of Error Recovery Techniques For IP-based Audio-Visual Multicast Applications, *IEEE Network*, pp. 1124-1136.

Chiu D. and R. Jain (1989). Analysis of the Increase/ Decrease Algorithms for Congestion Avoidance in Computer Networks. *Journal of Computer Networks and ISDN*, **17,1**, pp. 1-14.

Floyd S. and K. R. Fall (1999). Promoting the use of end-to-end congestion control in the Internet. *EEE/ACM Transactions on Networking*, **7, 4**, pp. 458-472.

Feng W.and J. Rexford (1997). A Comparison of Bandwidth Smoothing Techniques for the Transmission of Prerecorded Compressed Video, *Proceedings of the IEEE INFOCOM conference*, pp. 58-66.

Fall K. and S. Floyd (1996). Simulation-based comparison of tahoe, reno and sack tcp. *Computer Communication Review*, **26**, pp. 5 - 21.

Gillet D., C. Salzmann and P. Huguenin (2001). Distributed Architecture for Teleoperation over the Internet. *Lecture Notes in Control and Information Sciences 258: Nonlinear Control in the year 2000*, Springer-Verlag, 399-407, London.

Jacobson, V. (1988). Congestion avoidance and control. *Proceedings of SIGCOMM Symposium on Communications Architectures and Protocols*, pp. 314 - 329.

RFC793 (1981). Transmission Control Protocol ftp:// ftp.isi.edu/in-notes/rfc793.txt.

RFC768 (1980). User Datagram Protocol ftp:// ftp.isi.edu/in-notes/rfc768.txt.

Steven, R. W. (1994). *TCP/IP Illustrated, Volume 1: The Protocols*. Addison-Wesley.

Saha S. (2001). Image Compression - from DCT to Wavelets: A Review, *ACM Crossroad Electronic Magazine*, http://www.engr.ucdavis.edu/ ~ssaha/ sahaimgcoding.html.