Interaction Control Protocols for Distributed Multi-user Multi-camera Environments

Gareth W DANIEL and Min CHEN
Department of Computer Science, University of Wales Swansea
Swansea, SA2 8PP, UK

ABSTRACT

Video-centred communication (e.g., video conferencing, multimedia online learning, traffic monitoring, and surveillance) is becoming a customary activity in our lives. The management of interactions in such an environment is a complicated HCI issue. In this paper, we present our study on a collection of interaction control protocols for distributed multi-user multi-camera environments. These protocols facilitate different approaches to managing a user's entitlement for controlling a particular camera. We describe a web-based system that allows multiple users to manipulate multiple cameras in varying remote locations. The system was developed using the Java framework, and all protocols discussed have been incorporated into the system. Experiments were designed and conducted to evaluate the effectiveness of these protocols, and to enable the identification of various human factors in a distributed multi-user and multi-camera environment. This work provides an insight into the complexity associated with the interaction management in video-centred communication. It can also serve as a conceptual and experimental framework for further research in this area.

Keywords
Video-centred communications, distributed systems, camera control, interaction protocols, human-computer interaction, interaction management, interactive systems and models.

1. INTRODUCTION

Even in the early days of the Internet, with the ARPANET project, we see the inherent desire for interactive communication. With the rapid development of network infrastructures and streaming media technologies, *video-centred communication* is becoming a customary activity in our life. Some of us frequently hold multi-site videoconferences, whilst others sometimes stream lectures to remote classes.

Many of us have had at least a glimpse of a big-brother house on the TV or the Internet, whilst most of us are constantly captured by traffic or surveillance cameras. There is an ever-increasing scope for a distributed interactive environment involving *multiple users* and *multiple cameras*. The interaction in such an environment is usually complex and difficult to manage. The studies on the relevant interaction control protocols are particularly scarce.

One interesting feature of video-centred communication is that it can offer only a limited view of what is being filmed, which raises the desire of a viewer to gain control of the camera. Such a technical feature and its associated human factors are uncommon in text-, image-, or audio-centred communications. In other words, video-centred communication presents us a collection of HCI issues to be explored. An understanding of these issues would help answer many questions in designing software systems and user interfaces for video-centred communication. For example,

- in designing a remote meeting and collaboration environment, shall we restrict camera control locally, or make each camera controllable by all sites?
- in designing a web-based traffic monitoring system, what would happen if we allow web users to have simultaneous control of the cameras involved?
- in designing an area-surveillance system, how can we moderate the control of the cameras among different monitoring stations?

Such questions encompass a range of general as well as application-specific issues. In particular, the following three interrelated aspects of interactions represent the fundamental human factors that underpin a scientific answer to such questions:

1) Human-Computer Interaction (HCI) — how the users may interact with the cameras.
2) Human-Human Interaction (HHI) — how the users may coordinate the camera control among themselves.
3) Human-Computer-Human Interaction (HCHI) — how the system facilitates, moderates and manages user actions for manipulating the cameras.

This paper will focus on the HCHI aspect, whilst considering HCI and HHI whenever appropriate. In particular, we will present our study on different interaction protocols for managing the communications associated with camera controls. In our study, we have considered the abstract notions of interaction protocols, which address the generalised needs of many applications. We have demonstrated the technical feasibility of implementing these protocols through the development of an environment where a set of advanced technologies are integrated together. We have also conducted experiments that emulate a multi-user, multi-camera surveillance environment where such interaction protocols may be deployed. Our experimental results have demonstrated the relative merits of the protocols studied, and offered an insight into how these protocols may affect the management of interactive activities and the effectiveness of HCHI.

The rest of the paper is organised as follows. In Section 2, we will provide a brief review of related research on the major developments in video-centred communication technologies, models of session management and methods for remote interaction control. This will be followed by Section 3, where we will briefly describe our motivation and the potential applications of this work. In Section 4, we will present a web-based multi-user, multi-camera environment that facilitates video-centred communications using different interaction protocols. This will be followed by Section 5, where we will introduce a set of notions and describe a collection of interaction protocols in a relatively abstract form. We will describe the design and set up of our experiments in Section 6, and report and discuss our experimental results in Section 7. Finally, we will provide our concluding remarks in Section 8.

2. RELATED WORK

This work touches on many areas of human computer interaction. In this section, we provide a brief of related work in two main aspects of video centred communications, namely *technologies* and *interaction management*.

Technologies

Recently, research into software environments for video-conferencing addressed a number of design issues, including application control [15], floor control and question management [15, 23], meeting history management [11], video recording [19], life size displays [9], multiple camera views [30, 10], automated camera management [22], omni-directional cameras [26]. There are several special purpose video conferencing systems, among which TRIUMF, an integrated system for remote interviews, has captured many advanced technical concepts [20].

Traditionally, surveillance systems [28, 29] are related to video-conferencing systems in terms of camera technologies. Nowadays, there is significant convergence between two types of systems, in terms of technical specification, product design and support for interactions. It is desirable to explore the conceptual convergence between these systems from a HCI perspective.

With the aid of software development environments, such as Java™ Media APIs and NetMeeting-SDK, the difficulties in developing advanced systems for video-centred communications have been alleviated. One example is JASMINE [27], which utilises the Java framework to provide collaborative instrumental control across the Internet involving different platforms. Some features in the Java framework, such as dynamic session entry and exit, offer great flexibility for organising interactive activities, hence demanding control protocols for managing such activities.

Interaction Management

Interaction management in video-centred communications has largely been focused on *floor management* and *session management* for meetings [15, 23]. Recently, the consideration of this issue has been extended to collaborative environments in general [14, 27, 8, 24]. Applications of floor management, that is, medium-access or application-access control, further extend to Internet-based teleoperation [13], control of robot motion [12], real-time Internet multimedia applications [7], collaborative spacecraft design [21]. Over the coming years, as handheld devices become more widespread and wireless network technologies more prevalent, it is inevitable that there will be a rapid increase in activities that require interaction management [5].

An elementary, and perhaps still the most effective, means for managing interactive actives in a meaningful and orderly manner is to establish a protocol, which is a predefined agreement between parties involved in such activities. Many interaction models and protocols have been proposed for distributed learning environments [18, 2], multi-server, multi-client data browsing [17], management of sharing [25], controlling vision systems [6] and collaborative agents [3].

However, most of these protocols do not accommodate the needs for one or more users to interact with a remote instrument, such as a camera, directly. Such a need is more difficult to accommodate as (i) the users involved may not engage in collaboration, and may be unwilling to cooperate with each other, (ii) the ability to control the instrument usually has a more profound impact on the objective of a user.

3. MOTIVATION

This work was motivated by the demand for complex multi-user multi-camera surveillance systems, and the needs for sophisticated interaction management in such systems. In particular we are inspired by the recent discussions on sensing systems [1], and encouraged by potential applications in areas such as:

- *Computer-assisted experiment control*, e.g., for explosive experiments, radioactive environments, or location-constrained equipment.
- *Surveillance*, e.g., property, street and traffic monitoring, and police collaborative searches.
- *Entertainment*, e.g., watching a big brother house, and pay-to-view cameras in large concerts and sport events.
- *Tourism*, e.g., exploration of major cities and natural wonders, or even checking the queue length and weather condition of a tourist attraction.
- *Training*, e.g., live surgical procedures, and mine removal.
- *Other applications*, e.g., online property viewing, and cameras in a day-care nursery for parents.

Existing software that facilitates multi-user multi-camera interaction is generally restricted to a simple *equal round-robin* (*ERR*) protocol [4]. Although the ERR protocol is easy to implement and understand, its effectiveness is far from satisfactory in a collaborative environment. In particular, it does not facilitate "supply according to demand" in any way, nor provide scalability for a large number of users.

Har-Peled *et al* [16] recently proposed an algorithm for deriving camera positions using combined viewing constraints specified by a group of users. This effectively reduces the interaction management to a democratic voting process, but at the cost of introducing some undesirable side-effects. For example, a weighted average of various camera parameters may not meet most users' requirements; and it is difficult for users to predict the effects of their cooperative actions.

The lack of comparative studies to develop and identify effective and efficient mechanisms for interaction management

in multi-user multi-camera environments, also prompted the authors to take a comprehensive approach to this work, which involved the building of a multi-user and multi-camera environment, the development of a collection of interaction protocols and an experiment based on a real-life scenario.

4. SYSTEM OVERVIEW

In this section, we will describe a web-based multi-user and multi-camera environment, which represents a particular class of video-centred communications. We will delve into the hardware and software that facilitate interactions relevant to camera control.

Design Objectives

Despite the extensive use of video conferencing and surveillance-like applications, most video-centred communication environments involve only a single camera that is usually controlled by the service provider at the *broadcasting* end rather than by the client at the *viewing* end. Even in systems with user-controllable cameras, such as WebView Livescope, which is commercially distributed software from Canon™ [4], the control is usually limited to a very basic fixed time round-robin protocol.

Built upon our previous work on special-purpose video conferencing [20] and Internet-based collaborative environments [14], we have constructed a general-purpose system that allows us to examine various aspects of interactions from a broader perspective. Although the system may itself be used as a web-surveillance system, it was designed primarily for experimenting with different interaction protocols. Our design objectives include:

- *web-based* — this feature facilitates the use of general purpose web-browsers, and offers the potential of dis-seminating this work in a wide range of applications;
- *multi-user and multi-camera* — this feature provides a generalised platform for video-centred communications where complex HCI, HHI and HCHI scenarios can be formulated and studied;
- *intuitive to use* — this is not only a generally desirable feature, but also particularly important to our experiments for minimising the possible distortion attributed to diverse technical abilities among the subjects;
- *minimal software overhead* — this helps minimise the time delay between moving a camera and noting the change, thus enhancing the virtual presence.

Hardware

At the server end of the system, the main hardware components, which facilitate the remote camera control and video transmission, are:

- 1 web server (2.2Ghz, 80Gb HDD),
- 4 EVI-D31/B Sony video cameras (Figure 1),
- 1 Robot MV87 colour quad box (Figure 1),
- 1 Matrox Marvel G400-TV video capture card.

Because of the need for high interactive capabilities, Sony D31 cameras were deployed, which have some important features that are not found on most webcams. For example, ASCII byte

Figure 1: Sony EVI-D31/B camera & MV87 Colour Quad.

signals can be sent directly to a Sony D31 from a computer to which the camera is connected. With such signals, instructions, such as zooming, tilting and panning, can be sent to operate the camera remotely.

With a standard configuration, only one specific camera view can be captured by a video capture card. In order to facilitate multiple cameras in our environment, we utilised a colour quad box, with which video captured by up to four cameras can be transmitted back to a server. This configuration allows a user to select one of the four views, or all four views together. Again, an important feature of this quad box is that it can be controlled from a computer by sending it ASCII commands through an RS232 serial port. This implementation can be extended to include more cameras using multiple quad boxes or a more complex video mixing device.

When several users are receiving real-time video streams from the web server, there is a high consumption of processing power. The processor is not only capturing eight frames to the hard disk each second, but also continually supplying video images to remote users. A server with a 2.2 GHz processor and 512 Mb RAM was hence chosen, along with a high-speed 80 GB hard disk. Figure 2, illustrates the overall hardware architecture of the system. We note two distinct, yet integrally linked modules of the web server. These modules allow us to make a clear distinction between a *viewer* and *controller* of the cameras. All users will initially connect to the HTTP server and retrieve a real-time video stream from the server. The user can then make a further request for camera control, in which case they will maintain a connection to the Java server, as well as the HTTP server.

Software

There are four major pieces of software that were written to support communications and interactions.

Image Capture: This program was written using Borland's Delphi and has the primary purpose of archiving images to the hard disk for future retrieval. The program interacts with the capture card, which in turn is connected to the colour quad box and hence each camera.

The capture resolution is specified as 352 x 288 pixels, while the capture rate is set to 8 frames per second. Each image, initially captured in the bitmap format, is converted to the JPEG format with 20% compression before a name is assigned to the image. In total one million individual images are stored at any one time, allowing the user special functionality that will be described later. The program streams out the most recently captured images, according to some system values stored in the computers registry. Figure 3 shows the capture program in use, capturing a quad view from all cameras.

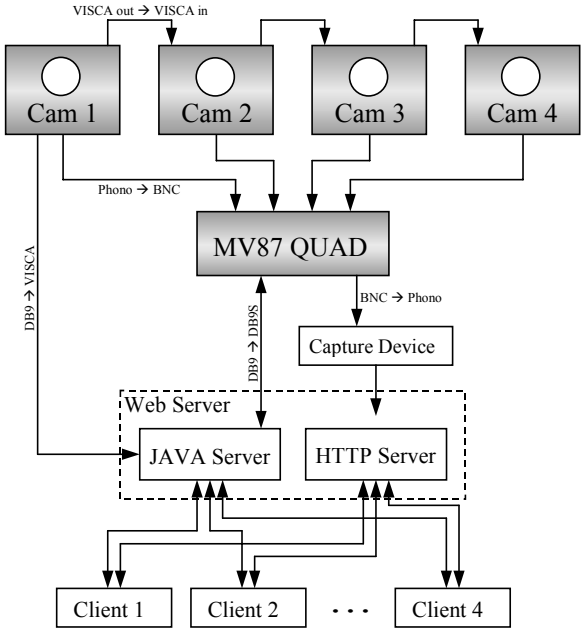

Figure 2: System Hardware Architecture

Camera Control Server: This program acts as a *hub of interaction* where most of the interaction management takes place. It relays control instructions from remote clients to any of the cameras and the MV87 quad box, hence allowing remote interactive control of these devices. The program was written entirely in Java together with various communication APIs. It maintains a two way RMI (Remote Method Invocation) session, allowing bi-directional communications to be conducted. The majority of protocol implementations can be found within this program.

Camera Control Client: This program was also written in Java, but as an applet, and is downloaded from the URL associated to the *Camera Control Server*. As Java applets are supported by most Internet web browsers, this enables a wide access across the Internet. Supported by the RMI communication API, the program provides users with a collection of camera control facilities, including switching between cameras, moving and zooming cameras, recording live or past video footage, moving cameras to a preset position, brightening the scene, and automatic sequencing between cameras. The user can also view past video streams in real-time, by specifying either a specific time to view, or an amount of time to go back by. When the cameras are not being used, motion detection is enabled, logging any movement detected by the cameras, for future analysis. Figure 4 shows a snapshot of the *Camera Control Client*.

Automated Movie Creator: During the interactive control session, of which the user is a part, the user can select to record at any time. When the user perceives that the observed video footage may be of future interest they can simply select the record button. Similarly, pressing this button again will end recording.

Figure 3: Image Capture Program

Figure 4: Camera Control Client and one of its pop-up windows, "Advanced Viewing Options".

The user can then decide whether they would like the chosen time segment to be converted into a video file and sent to them via email. When the user requests a movie file, this program written in Java, finds the required JPEG images and converts them into one QuickTime movie file running at the originally captured, eight frames per second. This movie file is then automatically sent via email to the email address specified by the user.

Other Minor Software: At this point it is also worth mentioning some smaller pieces of software that were written to support the system. One of these is a web page, written in JavaScript and HTML, for delivering video across the web (Figure 5). A similar web page was written to support the above-mentioned *Camera Control Client*. Some intricate JavaScript code controls smooth image delivery, together with a feature allowing the user to view past footage as though it were live. Several CGI-Scripts are run on the web-server. One of these extracts the most recent image captured from the cameras and delivers it to the user. Another script is used to obtain the IP address of each remote user for their unique identification. Another CGI-Script used for the entry of experimental results, logs data relating to the time, user, camera and object that was identified.

Figure 5: Video Delivery Website.

5. PROTOCOL DESIGN

In this section, we will describe a set of protocols proposed for managing interactions in our study. The experimental investigation of these protocols will be described in the next section EXPERIMENT. To facilitate consistent definition of the protocols, we introduce some generic notations for modelling a distributed multi-user, multi-camera environment in a relatively abstract manner.

Generic Notations

Consider a distributed environment centred on a server S, which may be divided into many functional servers in a real implementation. As shown in Figure 6, there are $\boldsymbol{k}$ *camera devices* $c_1, c_2, \ldots, c_k$, $\mathbf{n}$ *active users,* $u_1, u_2, \ldots, u_n$ (who are involved in camera control) and $\mathbf{m}$ *viewers,* $v_1, v_2, \ldots, v_m$ (who are not involved in camera control). Communications between different entities are represented as generic messages in the form of $M[x \rightarrow y]$. For example, $M[u_i \rightarrow S]$ represents a message from user u_i to the server S. Messages may be processed, by appropriate *filters* prior to the transmission or upon their reception. In the following discussions, we will consider mainly messages relevant to camera control. We also assume the followings:

- There are duplex connections between S to all other entities, and the order of messages sent along a particular connection is preserved. However, there is no assumption of a constant transmission speed in any form, and messages arriving from different connections are not guaranteed to have the same order as they were sent.
- Messages arriving at S from different connections are placed into a single queue, and S processes all messages sequentially. In some way, this software mechanism is similar to time-division multiplexing in networking.

A user's access to camera control can be considered as a mapping between $\{c_1, \ldots, c_k\}$ and $\{u_1, \ldots, u_n\}$. There are four basic strategies for the management of such a mapping:

1) *1U1C (one user one camera)* — A user may control no more than one camera, and each camera may be controlled by no more than one user.
2) *1UmC (one user many cameras)* — A user may control any number of cameras, and each camera may be controlled by no more than one user.
3) *mU1C (many users one camera)* — A user may control no more than one camera, and each camera may be controlled by any number of users.
4) *mUmC (many users many cameras)* — A user may control any number of cameras, and each camera may be controlled by any number of users.

These four approaches are illustrated in Figure 6. In this paper we will concentrate on protocols in the mUmC categories for multi-user multi-camera environments. The notion of *camera* here can easily be generalised to an *instrument*, making the protocols applicable to many other interactive environments.

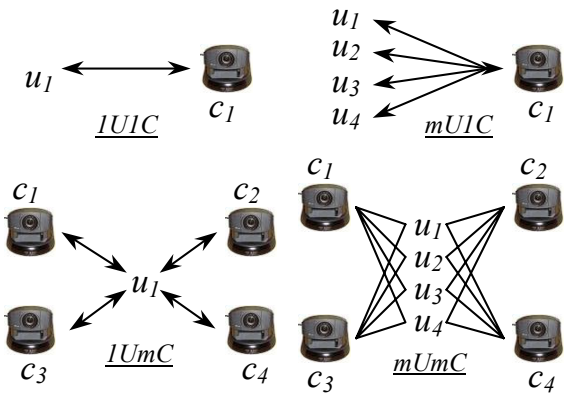

Figure 6: Four approaches to user-camera mapping.

An *interaction control protocol* defines the conventions to which interactive activities in a specific context must conform. It enables the users and instruments involved to have a meaningful interaction, and may in addition monitor and regulate interactive activities according to specific rules. In this study, we have considered four interaction control protocols in the mUmC category. The primary aim of these protocols is to facilitate effective use of cameras, and fairness in accessing camera control.

Simple Contention (SC)

In this protocol, every user u_i connected to the server S has the same access priority, at any time, in gaining the control of any of the cameras. All control messages are forwarded to the designated cameras. As the execution of each camera control

command takes time, it is possible for a later command to overwrite a previous command that is half way through its execution.

For instance, user u_1 may decide to move a camera c_j to the left with control message $M_a[u_1 \rightarrow S]$ while user u_2 would like to move the same camera to the right with message $M_b[u_2 \rightarrow S]$. The server S simply forwards the messages to the camera according to the order when they arrive at S. Suppose that M_a arrives at c_j first and activates the rotation motor of the camera. When M_b arrives at c_j a few milliseconds later, it will instruct the motor to discontinue the left motion, and start moving right.

When there are only a few users, this protocol can be quite effective, due to the fact there would be less contention for camera control. It is believed that this protocol may hold most benefit for users working in collaboration, rather than with complete strangers who often do not share a common objective, nor prepared to compromise. When there is a serious competition for camera control, the situation could become a bit destructive. There is nothing to say who should win this simple contention battle, as all users have the same access priority. Perhaps the one with a faster connection to S, a more powerful computer or a more persistent personality may eventually gain an advantage.

Contention with Access-Hold and Timeout (CAHT)
In this protocol, a user u_i gains control of a camera c_j through contention. Once the server S authorises the camera control to u_i, it will block the requests from all other users by filtering out their messages. In other words, u_i is allowed to hold on to the access (i.e., *access-hold*), provided that u_i is continually making use of the camera c_j. Once the remote u_i stops issuing instructions to c_j for a specified amount of time T_{out} (i.e., *timeout*), the server S will make the access available for the next round of contention. The main advantage of this protocol is the efficient use of the interaction medium by keeping the medium busy for productive interactions only. However its main disadvantage lies in the inherent unfairness of its makeup, which could be critical to some applications. One user could potentially hog the whole system for a very long period, leaving others waiting in vain, not knowing when they will next gain control.

Equal Round-Robin (ERR)
This protocol intends to address the fairness issue by allocating each user a specific time span, T_{span}, during which the user has complete control over all cameras. After a user u_i activates the *Camera Control Client*, the IP address of u_i is added to the end of a queue maintained by the server S. Users in the queue gain the control for a predefined duration on a first-come-first-served basis. While u_i is waiting in the queue, the *Camera Control Client* displays a countdown timer indicating when u_i will next gain control. Likewise, after u_i gains the control, a similar countdown timer shows the time left before losing control. When u_i relinquishes the control, the corresponding IP address is again placed at the end of the queue.

Although the protocol seems to be very fair, it lacks in flexibility and may introduce inefficiency in utilising the interaction medium. A user who is given the camera control may not make use of this opportunity at all, whilst others who do not have the control may really want to.

Weighted Round-Robin with Timeout (WRRT)
This is a variation of the *equal round-robin* protocol, with the introduction of two additional features, *weight* and *timeout*. The server maintains an account for each user u_i with a pre-determined weight w_i, indicating an allowance for a share of the interaction medium. Pre-registered users will log onto the server with appropriate usernames and passwords, whilst others may log on using the guest account, which gives the lowest weight. Different weight values correspond to different sizes of time span. The allocation of the weight can be a function of a range of attributes associated to users, such as user status. There is also an administrator account giving unrestricted control.

The protocols efficiency can be enhanced by introducing the timeout feature. Similarly to *Contention with Access-Hold and Timeout*, if a user does not interact with the camera for a period T_{out} after gaining control, the user will automatically lose the control, and the server S hands the control to the next user in the queue. When any user joins or leaves the system, every other user becomes aware of this fact. In the knowledge that other users may want control, the current controller can allow a timeout to occur.

This protocol is ideal for maintaining efficient use of the interaction medium, whilst removing the unproductive competition in the *Contention* protocol and inefficiency in *Equal Round-Robin*. It is particularly effective in managing interactions involving a less coherent group of users.

Remarks on Protocols
No doubt some protocols may be suited in certain situations, but completely inappropriate in others. Some situations may be suited to collaborative control, whereas it would be highly competitive in the case of others. We summarise the merits of each protocol in Table 1. The attributes considered are:

- Efficiency in using the interaction medium,
- Fairness in gaining access to camera control,
- User's ability to anticipate the outcome of an action,
- User's awareness of the system states, such as, who has the control and for how long.

Table 1: The relative merits of interaction protocols

	SC	CAHT	ERR	WRRT
Efficiency	①✓ ②×	✓✓	×	✓✓
Fairness	✓	××	✓✓	✓
Anticipation	××	✓	✓✓	✓✓
Awareness	××	③✓ ④×	✓✓	✓✓

① contention among a small group of users, ② a large group of users, ③ awareness about the user him/herself, ④ about other users

6. EXPERIMENT

In order to evaluate the efficiency and effectiveness of the above-mentioned protocols, it was imperative that they were tested in real-life situations. We designed our experiments based on the needs of a multi-user multi-camera security surveillance system. One can easily imagine the application of such a system in a practical circumstance. In this section, we

report our findings through one experiment for the successful identification of automobiles.

Experiment Set-up
The main entities involved in the experiment are:

- *Cameras:* There were four cameras, $c_1,\ldots, c_4$, which were set to view several different roads and car parks in a campus. All cameras can be panned, tilted and zoomed.
- *Users:* There were four users, $u_1,\ldots, u_4$, who were not allowed to communicate verbally with one another during the experiments. No viewer was involved.
- *Automobiles:* There were four target cars $a_1,\ldots, a_4$ to be identified, which were driven around the campus blending with other traffic flows.
- *Computing and Networking:* The server S is a P4 2.2GHz Computer with 512MB RAM and an 80GB HDD, connected to a 100Mbps Ethernet with TCP/IP. The computers used by all users are of the same configuration, that is an AMD Athlon 800Mhz, with 512MB RAM.
- *Protocols:* Three protocols were tested, which are *Simple Contention* (SC), *Equal Round Robin* (ERR) and *Weighted Round Robin with Timeout* (WRRT). With WRRT, all users were given the same weight. In effect, it is an *Equal Round Robin with Timeout* protocol.

Experiment Process
At the commencement of the experiment, drivers gathered in a location out of view of any camera. Drivers were given a precise route they should follow, along with the time they should depart into the scene. Each test lasted for about fifteen minutes, during which, four target cars made their way through the scene on their predetermined routes.

Figure 7 shows a map of the university campus, where the experiment was conducted. The blackened area in the figure represents the viewable road area, where target cars could potentially be identified. Cameras $c_1, \ldots, c_4$ were located in the tower building shown by a striped roof in figure 7. As an example, Figure 8 gives four driving routes that were followed by cars during the first session, where the *Simple Contention* protocol was tested.

All routes were prearranged, but they were arbitrarily selected for each protocol test. Although for each driver, the route to follow and the time to start was predetermined at the beginning of each session, these facts were unknown to the camera users, who hence had an impression of unpredictability about the events.

Goal and Result Collection
All users were made aware of the colour, model, driver and registration plate of each of the four target cars. The goal for each user was to make positive identifications of theses cars during each session. The final score of each user is the number of positive identifications subtracted by the number of negative identifications. There was a certain degree of competitiveness among the users, though success often needed a certain amount of cooperation, especially with the *Simple Contention* protocol. Figure 9 shows an image from the recording where car a_1 was caught on camera c_1 during the first session, which in fact led to positive identification by users u_1 and u_3.

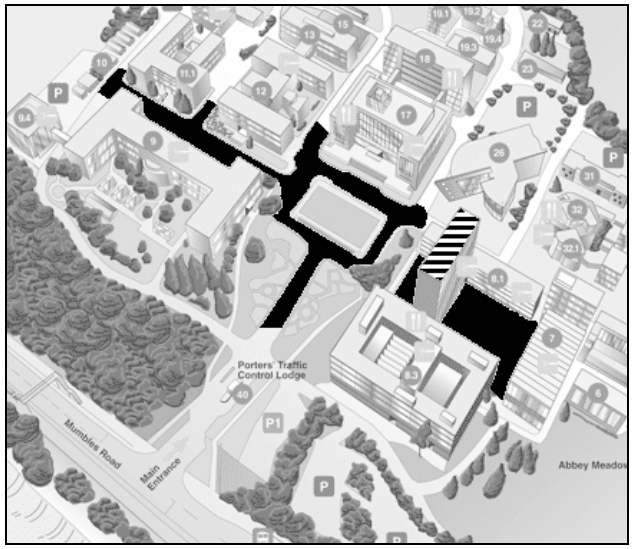

Figure 7: Road area viewable by cameras $c_1,\ldots, c_4$.

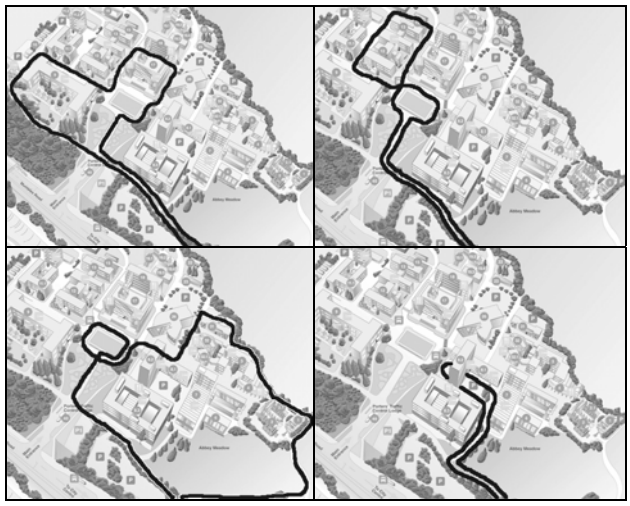

Figure 8: Four driving routes for session 1: SC

Figure 9: Automobile a_1 spotted in camera c_1.

A special web page, as shown in Figure 10, was set up for the experiment. It displays the features of each car, and facilitates quick and easy registration of car identification. Every time, a user considers that a target car has been located, he/she presses an appropriate button on the web page to register his/her identification. Each button is associated with a target car and a camera. This minimises the interaction needed for results collection. The server *S* then records the given response, the IP address of the user and the exact time of identification. In addition, the entire video sequence was recorded together with timing information stamped onto each image.

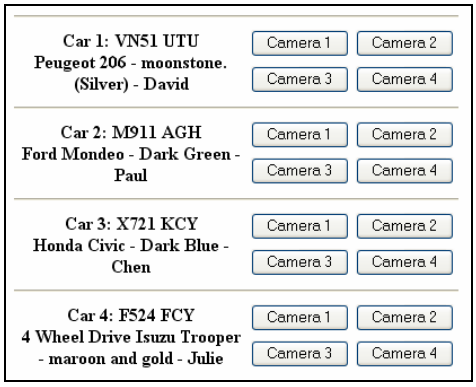

Figure 10: Result Collection Webpage

7. EVALUATION

Results

The video sequence recorded during the experiment was carefully analysed to ascertain (i) the times when each car entered and left in the viewable road area, and (ii) the times when a car was identified on a camera, indicated by users. The first set of information enabled us to measure the effectiveness of each protocol by examining the delay between the time when a car could potentially be identified and the recorded time of actual positive identification. The second set of information enabled us to differentiate positive identification from false identification.

The results were processed manually in two steps:

1) For each registered identification, we established if it was a positive or negative sighting. For repeated identification (either positive or false) of the same car registered by the same user, we consider only the first registration.
2) For each positive identification, we established the amount of *delay*, from the entry of the target car into the viewable road area, until it was positively spotted.

Tables 2, 3 and 4 list all recorded positive identifications during the three sessions respectively and the measured delays. The data is summarized in Table 5. For each protocol, the table lists the total numbers of positive and negative sightings, together with the average, minimal and maximal time needed for positive identifications. Figure 11 shows a chart that depicts the data in Table 5 in a comparative manner.

Observations

From experimental results, we have made some interesting observations regarding the interaction protocols in the *mUmC* category.

Table 2: Positive identifications with the *Simple Contention* (SC) protocol.

Car	Potentially Viewable	Positive Identification	User	Delay (*seconds*)
a_3	10:53:49	10:54:34	u_1	45
a_1	10:49:00	10:49:35	u_1	35
		10:49:53	u_3	53
a_2	10:47:20	10:48:50	u_1	90

Table 3: Positive identifications with the *Equal Round Robin* (ERR) protocol.

Car	Potentially Viewable	Positive Identification	User	Delay (*seconds*)
a_4	11:15:02	11:15:23	u_1	21
a_1	11:17:28	11:17:34	u_1	6
		11:17:37	u_3	9
a_3	11:16:27	11:16:35	u_1	8
		11:16:39	u_4	12
	11:17:40	11:18:03	u_1	23
		11:18:04	u_4	24
	11:19:19	11:19:41	u_4	22
		11:19:44	u_3	25
		11:20:18	u_1	59
		11:20:28	u_3	69
a_2	11:11:10	11:11:41	u_4	31

- Due to conflicting interests of the users, the *Simple Contention* (SC) protocol appeared to be quite inefficient and ineffective, resulting in a string of negative identifications, and longer delays for positive identifications.
- The *Round Robin* based protocols, where a user has complete control for a period of time, are more effective in facilitating identifications. All users (including those without the control) have benefited from the relatively unwavering camera control by one user over a small period.
- ERR, without a timeout mechanism, has shown to be slightly inefficient. Our recording shows that the timeout mechanism did in fact allow productive users to make more effective use of the cameras. On average, WRRT has the shortest delay from the time a target car entered the scene to a positive identification.
- Dark coloured cars are more likely to be wrongly identified than brighter cars.

In general, our test results successfully reflected most of the analytical predictions given in Table 1.

Table 4: Positive identifications with the *Weighted Round Robin with Timeout* (WRRT) protocol.

Car	Potentially Viewable	Positive Identification	User	Delay (*seconds*)
a_2	11:32:15	11:32:37	u_4	22
		11:32:38	u_3	23
	11:33:34	11:34:02	u_4	28
a_3	11:29:01	11:29:26	u_1	25
		11:29:32	u_3	31
		11:29:34	u_4	33
a_1	11:34:06	11:34:11	u_3	5
		11:34:13	u_1	7
		11:34:15	u_4	9
	11:35:31	11:35:35	u_1	4
		11:35:41	u_3	10
		11:35:44	u_4	13
	11:37:29	11:37:57	u_1	28
		11:38:00	u_3	31
		11:38:03	u_4	34
a_4	11:36:26	11:36:39	u_1	13
		11:36:44	u_3	18
		11:36:53	u_4	27
		11:36:53	u_2	27

Table 5: Time (*sec*) to identify cars for each protocol.

Protocol	Avg. time	Min. time	Max. time	Positive sightings	Negative sightings
SC	55.8	35	90	4	12
ERR	25.8	6	69	12	6
WRRT	20.4	4	34	19	6

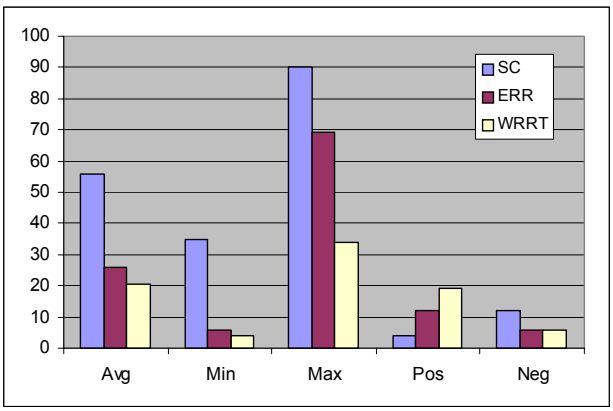

Figure 11: Bar chart depicting Table 5.

8. CONCLUSIONS

We have considered video-centred communications in a multi-user multi-camera environment. We have presented four interaction protocols to support human-computer-human interactions in such an environment. A web-based system was built, which allowed us to evaluate these protocols in the real world. Our experiments have demonstrated the importance of allowing each user to have an unrestricted period of control, and efficiency of the timeout mechanism.

Our work echoes some important issues that were raised in a recent study on making sense of sensing systems [1], and highlights the necessity and feasibility of designing suitable protocols for managing interactions in distributed systems.

We will continue our study to involve user groups of different sizes in order to establish the effectiveness and efficiency of each protocol in relation to the number of users. We will look into other protocols such as reservation-based ones. We also intend to investigate the protocol performance in circumstances when direct collaborative interactions are allowed.

ACKNOWLEDGMENTS

We are grateful to Alfie Abdul-Rahman, Justin Biddle, David Clark, Owain Couch, Julie Pellard, Paul Roberts and Ann Smith, who participated in our experiments. Special thanks to Justin for his enthusiasm and help, and to Prof. Mike Webster for his advice in the early stage of this work.

REFERENCES

[1] Bellotti, V., Back, M., Edwards, W.K., Grinter, R., Henderson, A., Lopes, C. **Making sense of sensing systems: five questions for designers and researchers**. Proc. the SIGCHI Conference on Human factors in computing systems, Minneapolis, Minnesota, USA, 415-422, 2002.

[2] Burger, C., Rothermel, K., Mecklenburg, R. **Interactive protocol simulation applets for distance education**. Interactive Distributed Multimedia Systems and Telecommunication Services 1483, Oslo, Norway, 29-40, 1998.

[3] Bussmann, S., Jennings, N.R., Wooldridge, M. **Re-use of Interaction Protocols for Agent-Based Control Applications**, Agent-Oriented Software Engineering III: Third International Workshop (LNCS 2585), Bologna, Italy, 73-87, July, 2002.

[4] Canon, **Webview Livescope: Image Webcasting System**. http://www.canondv.com/livescope.

[5] Crocker, S.D. **Internet-Based Collaboration in 2010**, IEEE Internet Computing, 4, 53-54, 2000.

[6] Demazeau, Y., Boissier, O., Koning, J.L. **Using Interaction Protocols to Control Vision Systems**, Proc. of 1994 IEEE International Conf. on Systems, Man and Cybernetics, San Antonio, Texas, 1616-1621, 1994.

[7] Dommel, H.-P., Garcia-Luna-Aceves, J.J. **A novel group coordination protocol for collaborative multimedia systems**, IEEE International Conf. on Systems, Man, and Cybernetics, 2, 1225-1230, 1998.

[8] Ewards, K. **Session management for collaborative applications**. Proc. the Conference on Computer Supported Cooperative Work, Chapel Hill, North Carolina, United States, 323-330, 1994.

[9] Fish, R.S., Kraut, R.E., Chalfonte, B.L. **The video window system in informal communications**. Proc. CSCW, 1-11, 1990.

[10] Gaver, W. W., et al. **One is not enough: multiple views in a media space**, Proc. HCI & INTERCHI'93, Amsterdam, p335-341, 1993.

[11] Ginsberg, A., Ahuja, S. **Automating envisionment of virtual meeting room histories**. Proc. 3rd ACM International Multimedia Conference, San Francisco, CA, 65-75, November, 1995.

[12] Goldberg K., Chen, B. **Collaborative Control of Robot Motion: Robustness to Error**, IEEE/RSJ International Conf. on Robots and Systems, Maui, HI, 2001.

[13] Goldberg K., Chen B., Bui, S., Farzin B., Heitler J., Poon D., Solomon R., Smith G. **Collaborative Teleoperation via the Internet**, Proc. IEEE Int. Conf. on Robotics and Automation, San Francisco, CA,2019-2024, 2000.

[14] Haji-Ismail, A.S., Chen, M, Grant P.W., Kiddell, M. **JACIE – an authoring language for rapid prototyping net-centric, multimedia and collaborative applications**. Annuals of Software Engineering, 12, 47-75, 2001.

[15] Handley, M., Wakeman, I., Crowcroft, J. **The Conference Control Channel Protocol (CCCP): A scalable base for building conference applications**. In Proc. ACM SIGCOMM'95, New York, August, 1995.

[16] Har-Peled, S., Koltun, V., Song, D., Goldberg, K. **Efficient Algorithms for Shared Camera Control**, Proc. of the nineteenth conference on Computational geometry, Diego, California, 68-77, 2003.

[17] Helbig, T., Schreyer, O. **Protocol for browsing in continuous data for cooperative multi-server and multi-client applications**. Interactive Distributed Multimedia Systems and Telecommunication Services 1483, Oslo, Norway, 231-236, 1998.

[18] Hilt, V., Geyer, W. **A model for collaborative services in distributed learning environments**. Interactive Distributed Systems and Telecommunication Services 1309, Darmstadt, Germany, 364-375, 1997.

[19] Holfelder, W. **MBone VCR — Video Conference Recording on the MBone**. Proc. 3rd ACM International Multimedia Conference, San Francisco, CA, 237-242, November, 1995.

[20] Kiddell, M, Chen, M., Oborne, D.J., Slater F.W., McCulloch, M. **TRIUMF – a system for remote multimedia interviewing**. Proc. IEEE International Conference on Multimedia Computing and Systems (ICMCS), Florence, Italy, Volume II, 715-719, 1999.

[21] Lamarra, N., Dunphy, J. **Web-based operation of interactive sharable environment for collaborative spacecraft design**, IEEE Proc. of Aerospace Conference, 2, 43-49, 1999.

[22] Liu, Q., Rui, Y., Gupta, A., Cadiz, J.J. **Automating camera management for lecture room environments**. Proc. SIGCHI Conference on Human Factors in Computing Systems, Seattle, Washington, USA, 442-449, 2001.

[23] Malpani R., Rowe, L.A. **Floor control for large-scale MBone seminars**. Proc. 5th ACM International Multimedia Conference, Seattle, Washington, 155-163, November, 1997.

[24] Martin, D., Rouncefield M., Sommerville, I. **Applying patterns of cooperative interaction to work (re)design: E-government and planning**. Proc. of the SIGCHI Conference on Human Factors in Computing Systems, Minneapolis, Minnesota, USA, 235-242, 2002.

[25] Patterson, J., Hill, R., Rohall, S. **Rendezvous: An architecture for synchronous multi-user applications**. Proc. CSCW, 317-328, 1990.

[26] Rui, Y., Gupta, A. and Cadiz, J.J. **Viewing meeting captured by an omni-directional camera**. Proc. SIGCHI Conference on Human factors in Computing Systems, Seattle, Washington, USA, 450-457, 2001.

[27] Saddik, A., Shirmohammadi, S., Georganas, N., Steinmetz, R. **JASMINE: Java application sharing in multiuser interactive environments**. Proc. Interactive Distributed Multimedia Systems and Telecommunication Services, Enschede, The Netherlands, 214-226, 2000.

[28] Tsukada, A., Hata, T., Matsuda, F., Sato, K., Ozaki, M. **Video data management in media controller: A distributed multimedia surveillance system**. Interactive Distributed Systems and Telecommunication Services 1309, Darmstadt, Germany, 231-240, 1997.

[29] Wolf, K., Froitzheim, K., Weber, M. **Interactive video and remote control via the world wide web**. Proc. Interactive Distributed Multimedia Systems and Services, Berlin, Germany, Springer, 91-104, 1996.

[30] Yamaash, K., Cooperstock, J. R., Narine, T., Buxton, W. **Beating the limitations of camera-monitor mediated telepresence with extra eyes**. Proc. ACM/SIGCHI CHI' 96, Vancouver, British Columbia, 50-57, 1996.