

# Distributed Meetings

## A meeting capture and broadcasting system

Masterseminar  
Spicher Christian, University of Freiburg  
[christian.spicher@unifr.ch](mailto:christian.spicher@unifr.ch)

März 2005

### 1. Introduction

During this Seminar in the DIVA (Document image Voice Analysis) Group of University of Freiburg, some studies over multi media solutions to archive and browse the content of meetings were analysed. One of them were two papers from the Microsoft research group, in which a prototype of software tool called “DM: Distributed Meetings” was presented. The Microsoft solution to the need of a meeting browser meets their requirements quite well: the hardware costs should be quite moderate – for what reason they decided to use standard mass ware – and the resulting browser should be easy and intuitive to use.

## 2. Overview

The whole concept focuses strongly on the whiteboard as the most important graphical media during the session. As to keep up with the idea of having cheap hardware, a special electronic whiteboard was declined. A further reason to simply use a normal whiteboard was the unhandy usage of special pens that have to be used in combination with an electronic whiteboard.

Instead, the concept is to record the content of the whiteboard visually and post process and recognize the content of the whiteboard. Figure 1 gives a bird view of a schematic design of the conference room.

- 1) normal everyday whiteboard
- 2) high resolution whiteboard camera on the opposite wall
- 3) a ring camera covering a 360° field of view recording the faces of the participants
- 4) an overview camera covering the whole room
- 5) a “kiosk” called electronic IO board with a card reader
- 6) the meeting room server interconnects all components

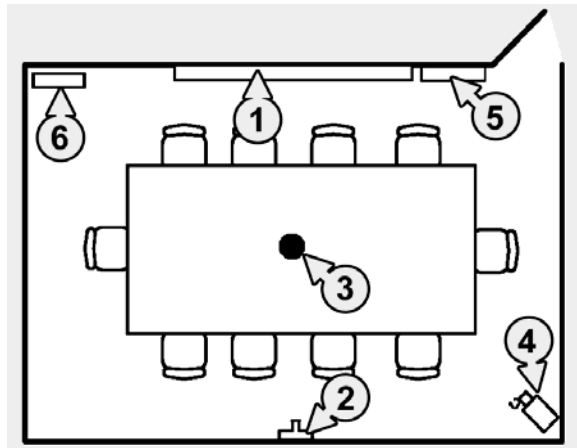


fig. 1 given a schematic overview of the design

The DM System allows to participate a session live by accessing the streamed content live via internet. These remote participants (live clients) can even interact with those ad hoc via a normal telephone line.

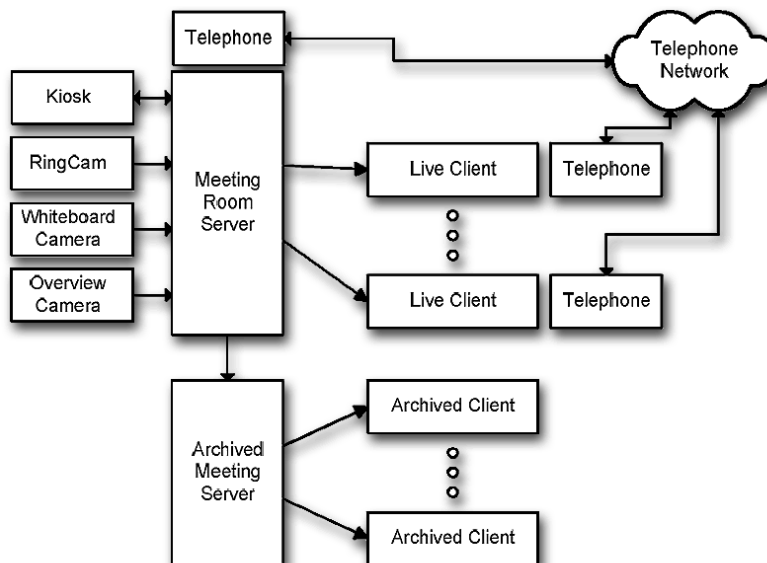


Fig. 2: The Components of the DM system

When the meeting is over, the recorded data is analysed and stored on the “archived meeting server”.

### 3. The Browser

The multi media browser that serves to view any stored session is designed very similar to broadly used movie viewers, like Microsofts Mediaplayer. Similar to this one, the DM browser has a large time line at the bottom that allows quick navigation through time (5). Figure 3 shows a screenshot of the DM browser.



fig. 3: the GUI of the browser

But the most important part is the whiteboard: it gets the largest part of the browser and it is in the center of it too (1). This manifests the importance of the whiteboard content. But the navigation through time is also possible by another component: the key frame history (2). Every time, right before the content of the whiteboard is erased, a snapshot with its content is stored as a so called “key frame”. They are visualised in chronological order as thumb nails in a vertical bar, and clicking one of these nails causes the browser to jump at the start where this content was being drawn on the whiteboard.

The portrait monitor (3) shows the person currently speaking. This is a magnified part of the 360° view of the ring camera (6). But there are also buttons to navigate through the sessions content similar to any (virtual) media player (4).

## 4. Hardware

**Whiteboard capturing device:** The MS research team decided to use a high-resolution still camera to capture the content of the whiteboard. They chose a consumer 4 megapixel Canon G2 that creates a snapshot every five seconds. It is connected to the meeting room server via USB and transfers the image data as (M)JPEG.

**Ring camera:** To achieve a 360° capturing functionality, test with a single sensor omnicam with a hyperbolic mirror were done and it showed that this system suffers from poor resolution. Thus an adapted hardware solution consisting of cheap consumer components were successfully integrated into the system.

The ring camera consists of five inexpensive 1394 cameras. The resulting images are corrected for radial distortion and stitched together with a static image remapping table, all in software. The data is transferred via 1394 bus (Firewire).

The blue head with the cameras is mounted on a massive, stable foot to improve the perspective on the participants. But it is low enough to not cross the direct views of the persons one to each other.

Additionally, 8 microphones are integrated in the foot which allow recording the speech and locating the angle of the speaker. The position close to the table surface minimized sound reflections.



*fig. 4: the engineered ring camera*

**Overview camera:** The clever solution for a reasonable overview to the conference room is quite trivial: another cheap consumer camera with 640x480 resolution at 15 FPS recording speed covering 90° horizontal field of view. It is connected to the meeting room server via 1394 bus as well.

**Meeting Room Server & Archived Meeting Server:** both are moderately sophisticated PCs, namely two dual CPU 2.2GHz Pentium 4 workstations with a RAID to store the mass of data.

**Kiosk:** The “Kiosk” called control board basically consists of a few buttons to control the DM system and a keycard reader to automatically identify the participants.

## 5. Sound Source Localization

The eight audio streams from the ring camera are needed to apply sound source localization. As the sound of the current speaker does not hit the each microphone at exactly the same time – the distance of about 0.2 m from the first microphone to the opposed one takes about another microsecond – the source can be determined using triangulation. For this purpose, only 3 microphones would be sufficient, but the other ones are handy to reduce imprecision.

The sound is also filtered to reduce noise, for example from fans, from the server etc. Another disguising effect are sound reverberations. A technique called “beam forming” increases the sound quality. Beam forming means that

## 6. Virtual Director

The “Virtual Director” is that program who decides who is visible in the speaker view on the top left of the browser. As mentioned, it shows a magnification of the ring camera image. As any real director, it has to show the current speaker and or multiple persons, if more than one is speaking. This task sounds difficult, and the papers of the MS team do not reveal a lot of technical details. The virtual director simply uses the results from SSL and combines it with a visual multi-person tracker to localize the current speaker and focuses in that direction. Therefore, the assumption that the virtual director is not yet working as perfectly as desired is obvious.

## 7. Whiteboard recognition

As mentioned, the whiteboard recognition bases on visual capture and recognition techniques only, especially were undesired:

- Special drawing and erasing tools
- A “keyframe” button that needs to be pushed before erasures
- High-sophisticated, costly hardware

So, the whole concept consists of capturing the whiteboard using a still camera and handle the whiteboard content basing on pixels.

To archive and recognize the content of the whiteboard, the image is split up in small cells that are independently processed. The most difficult problem is posed by the fact that parts of the whiteboard may be obstructed, some parts of the whiteboard may even be invisible to the camera for multiple frames. So, each cell is classified at every image as either (handmade) **stroke**, **background** or **foreground** object.

The classification is performed in seven steps:

### Step 1: Rectifying

First, the image is cropped to the content of the whiteboard only, then the polygon is bi-linear warped using bi-cubic interpolation, applying constant parameters. (Fig 7.1) As the whiteboard camera doesn't move, the parameters have to be calibrated only once.

Of course, it could be automatized applying advanced image recognition algorithms, but the MS team decided to keep it simple in this study.

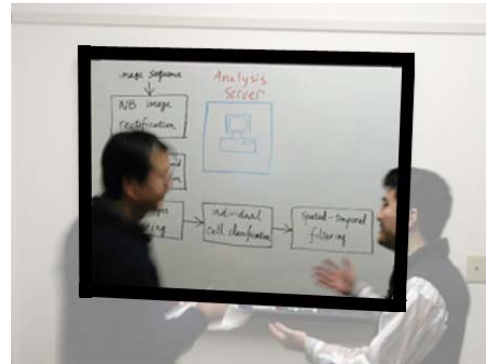


Fig. 7.1: the original camera shot

### Step 2: Extract BG color

After the image was corrected, it is split up in cells. (Fig. 7.2) The size of the cells is near the size of letters written. It shouldn't be too small to not lose the advantage of splitting up into cells, and not too lose precision.

Then, the background color has to be calculated as accurate as possible, to improve the final image quality, after the color balancing.

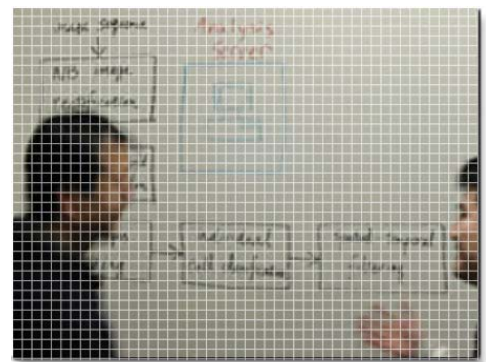


Fig. 7.2.1: the cropped image is split up into cells

### Step 2: Strategy A:

Assuming that the whiteboard cells are always the brightest, use the whitest found color in a cell. Holes are simply filled with the same color as the nearest non-holed neighbour. Massive problems show up if someone in the foreground wears white clothes or uses, like in our example, a white paper. Some parts are too bright, a better strategy needs to be found. (Fig. 7.2.2)



Fig. 7.2.2: Strategy A is not very successful

### Step 2: Strategy B:

A color histogram of each cell over the time is computed (Fig. 7.2.3, red). Now, the peak (bright) cells are most likely background cells of the whiteboard. However, as strategy A shows, some peaks are too white. Therefore, these “outliers” have to be detected.

A simple least-median-square results in a clean curve (Fig. 7.2.3, blue) that approximates the background color quite well. (Fig. 7.2.4)

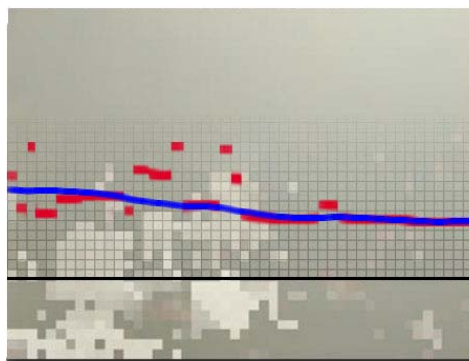


Fig. 7.2.3: Strategy B applies a histogram, here with an imaginary 2D example for visualisation purpose.



Fig. 7.2.4: Strategy B's result is satisfying

### Step 3: Clustering:

The content of each cell changes over time, from shot to shot. If the content of the same cell is equal to the following and preceeding, they are grouped together in so called “clusters”. Clusters will be needed in later steps. (Fig. 7.3)

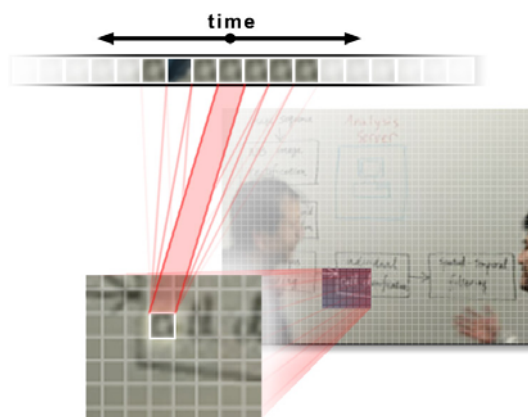


Fig. 7.3: Clusters groups the cell together over time

#### Step 4: Classifying the cells:

Every cell of every whiteboard shot has now to be classified either as:

- Whiteboard (WB) / background
- Strokes
- Foreground objects

The attributes considered are straight forward:

- a whiteboard cell is greyish, with the RGB values approximately the same.
- A stroke cell is mostly white or grey with one or two primary colors mixed in.
- A foreground object has none of the above

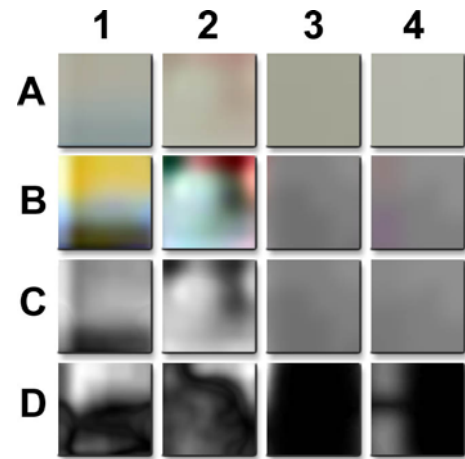


Fig 7.4.1: examples of cell categorisation

Fig. 7.4.1 shows four examples, 1 and 2 with blue and red strokes cells, 3 and 4 with background cells. Row A are the original cell contents. Row B are the same but they are normalized. Row C are the same as B but in greyscale. Row D finally shows the difference between B and C. This last row visualises the clear difference between the two strokes that are quite bright and the greyish background cells that end up as very black. Thus, all that needs to be specified is a threshold value to differentiate these two cases.

Now, the cells have to be compared to the previously calculated whiteboard background color  $I_w$ . Be  $\bar{I}$  the cells current mean color,  $\sigma$  the currents cells standard deviation and  $\sigma_w$  the deviation of the whiteboard. Then the scheme on fig. 7.4.2 is applied to categorize the cells.

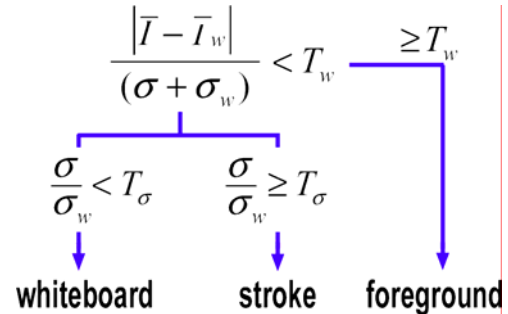


Fig. 7.4.2: the categorization tree

#### Step 5: Filtering the cells:

Isolated “foreground” cells are reclassified as strokes because obscuring objects are always quite large.

In a last step, all cells that have been categorized as “stroke” and that are neighbours to foreground object cells are re-assigned as foreground objects cells too. This step was inserted due to the experience that the foreground objects brought “dirt” to the next stroke cells. Fig. 7.5 shows the final result.

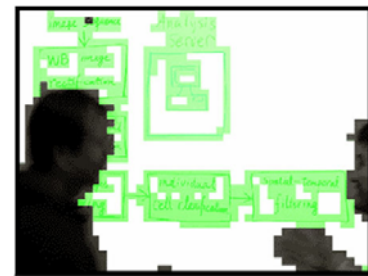
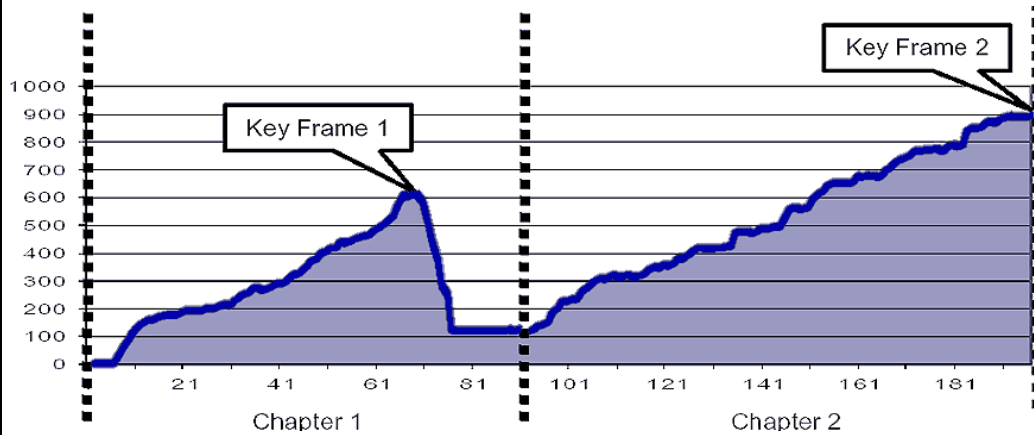


Fig. 7.5: the final result after categorization

### Step 6: Extracting key frames:

A key frame is the summary of the whiteboard session. Before a larger part of the whiteboard is erased, such a snapshot should be stored right before the erasure. Of course, these key frames should contain all the whiteboard content before the erasure, and preferably no foreground objects. The decision when to make that snap shot is very simple and straight forward, looking at a graph with the numbers of categorized stroke cells of all shots, *fig 7.6*.



*Fig. 7.6: A curve showing the amount of cells categorized as “strokes” in the y-axis, along with time on the x-axis. The dashed lines signify the “chapters” of the session.*

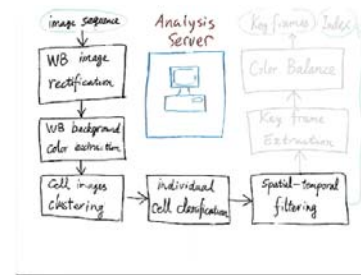
The drawing process on the whiteboard manifests itself in a slowly increasing curve, an erasing process as rapidly falling curve and the key frames are taken on the local maxima, on the “tips of the mountains”.

If the shots of the ideal moment (right before the erasure starts) has occluded parts in it, they are replaced by the nearest, older content in that cell, that is not part of a foreground object. This way, the full key frame can be puzzled together.

### Step 7: Key frame color balancing:

Now, all cells of the key frame are normalized and glued together to the final result (*Fig 7.7*). Normalization improves the image quality hard.

The calculated background from Step 2 now comes in: every cell is brightened up in a way so that the background part of it all become bright white.



*Fig. 7.7: the final result*

## 8. Playback:

By grouping the pieces of each cell into clusters, the exact moment when a stroke is drawn in that cell is easy to calculate: its when a cell turns from “background” to “stroke”. As mentioned, the DM browser allows to jump to a precise point in time by clicking at a key frame thumb nail on the top right. The whiteboard view in the center of the browser not only shows the stroke cells that have been made up to the current playback time, but it can also display the strokes that are being drawn in the future (but of course in the same “chapter”, i.e. until the whiteboard is being erased).

These future strokes – so called “ghost” strokes – are brighter than the actually existent strokes. An alpha value that is customisable between 0.0 (invisible) and 1.0 (opaque) causes the future “ghost” strokes to become more or less visible.

## 9. Limitations:

One weakness of the proposed whiteboard recording result in missing parts of one or more key frames, if an object or a person is obscuring it over a long time. On the other hand, this “drawback” is shared with any natural whiteboard too: if a person stands still on the same part all the time, no one is able to read the content behind him!

Another, more severe drawback lies in the unsophisticated state of the whiteboard recording and recognition system. It assumes constant lightning of the room. If light conditions vary at the meeting, problems occur as the background color of the whiteboard is not constant. This problem could be encountered by inventing advanced logic or simply by gluing a poster with three points of calibrated color on it on top of the whiteboard, close enough that the whiteboard camera records the points too. Then, the image could be preprocessed by correcting the image based on the color displacement on the three points.

## References

- [1] MSR: Meeting recording <http://research.microsoft.com/~rcutler/DM/dm.htm>
- [2] MSR: A Whiteboard and Audio Meeting Capture System <http://research.microsoft.com/users/zhang/WhiteboardArchiving/>
- [3] MSR: Li-wei He's Home Page <http://research.microsoft.com/~lhe/>
- [4] MSR: Distributed meetings: a meeting capture and broadcasting system. Cutler, R., Rui, Y., Gupta, A., Cadiz, J., Tashev, I., He, L.w., Colburn, A., Zhang, Z., Liu, Z., Silverberg, S. (2002). [http://www.research.microsoft.com/~rcutler/pub/dm\\_mm02.pdf](http://www.research.microsoft.com/~rcutler/pub/dm_mm02.pdf)
- [5] MSR: Why take notes? Use the whiteboard capture, Li-wei He, Zicheng Liu, Zhengyou Zhang <http://research.microsoft.com/users/lhe/papers/icassp03.wbcap.pdf>

*MSR: Microsoft Research*