The Magic Table: Computer-Vision Based Augmentation of a Whiteboard for Creative Meetings

François Bérard
CLIPS-IMAG, University of Grenoble
francois.berard@imag.fr

Abstract

This paper presents the "Magic Table": an augmented whiteboard surface for supporting creative meetings. The Magic Table uses computer vision for scanning and spatially organizing texts and drawings on the surface. Digitization of the physical ink is done by a-posteriori capture of the strokes. The digital information is organized through the manipulation of tokens (small plastic disks). The interaction consist of fast and easy to learn gestures that support multiple simultaneous users and two-handed control.

After motivating our approach with respect to the more common pen trajectory capture approach, we detail the interaction offered by the Magic Table and report on our initial observations of users interacting with the Table. Finally, we present the implementation of the two main components of the system: the color model based token tracker and the scanner based on mosaicking techniques.

1. Introduction

1.1. Motivations

The use of whiteboards, on which people draw with dry-erase pens, is widespread throughout the majority of modern organizations. Their success is attributable to two main qualities: ease of use and rapidity. We are taught to draw with a pen at a very early age and so everyone knows how to write, draw, and erase on any whiteboard. Furthermore, using the whiteboard is fast and fluid: there is no activation time (if only removing the cap of a pen). The limiting factor in writing and drawing speed is human performance, not the whiteboard performance (i.e. whiteboard pens adapt to any human writing speed). Switching between the whiteboard "functions" (draw, erase, change color) only requires a quick gesture. Because of these essential qualities, the whiteboard has become the universal tool for supporting creative meeting in many domains (science, engineering, business, education, etc.).

Despite its successes, it is easy to see that the whiteboard also has some severe limitations. These are mainly due to the *physical* nature of the whiteboard, as opposed to the *digital* nature of the more recent tools that support creative work. Once all of the writing space on a whiteboard has been filled, one has no choice but to erase some of the drawings if the work is to be carried on. Often, it is not easy to choose what to erase because every part is important. Sometime, the information is spatially reorganized: some parts are erased from one location and redrawn, often smaller, in another location. This manual reorganization is lengthy, cumbersome, and prone to copy errors.

There has been a great amount of work in designing "electronic whiteboards" that overcome these limitations. Commercial whiteboards, such as the LiveBoard [2], the SoftBoard [http://www.websterboards.com/], the SmartBoard [http://www.smarttech.com/] or the Mimio [http://www.mimio.com/] make use of special hardware to capture the trajectory and color of the pens. The problem is that the capture technology tends to get in the way of the most compelling user requirements of speed and immediacy, as described above. For example, all systems require that users exert a stronger pressure on the pen compared to the unconstrained writing. The technology is often limited to a single user at a time, which can be a source of frustration for a tool meant to support fluid collaborative work. The resolution of capture also carries some limitations: the shape of the stroke is not captured (width, calligraphic effects), and insufficient temporal resolution doesn't allow for very fast writing. Moreover, as Stafford Fraser points out [6], using specially equipped tool for trajectory capture doesn't support common practice on a whiteboard such as correcting the shape of a letter with the tip of the finger, using post-it notes to easily reposition them, or using a paper towel when the regular eraser is lost.

Because of their limitations, electronic whiteboards have failed to get into mainstream use in most places where a whiteboard is used. Their spread is limited to the support of formal presentations, such as a lecture or a business presentation. It could be argued that the cost of the equipment is the limiting factor, but our observation is that, for creative ("brainstorm") meetings, the electronic services of the whiteboards are not activated even where they are available.

The electronic whiteboards, by adding functionality to an ordinary whiteboard, fail to conserve its simplicity and rapidity. It appears that, with current technology (optical pen for the LiveBoard, laser scanning for the SoftBoard, touch-sensitive surface for the SmartBoard, or ultrasound signal tracking for the Mimio), capturing the trajectory of the tools always incurs a degradation of the ergonomics and speed of writing. This is the main motivation for working along another path toward the electronic augmentation of a whiteboard: the a-posteriori visual capture of strokes.

Because it does not seem achievable to capture the pen trajectory in a nonintrusive way, we choose to let the user write with unmodified pens and surfaces, then subsequently capture the stroke with a camera. This way we insure the conservation of the fundamental qualities of writing on a whiteboard. At this point, it should be noted that this approach does not yield the same information as the capture of the pen trajectory. In particular the dynamic information is lost (pauses in the writing, temporal sequence of strokes, etc.). The dynamic information is essential for the recognition process used in character recognition or beautification of drawings [3]. It is also possible to analyze the trajectory in real-time to detect command gestures, such as "cut" and "paste". Flatland [4] for example makes extensive use of gesture recognition to offer a rich set of electronic functionality to its users. We choose to sacrifice the dynamic information because we hypothesize that, in the context of creative meetings, conserving the ergonomics of the writing is of higher priority from the end-user perspective.

1.2. Related work

The idea of using physical tools while trying to augment them with electronic capabilities without degrading their natural properties first appeared with the Digital Desk prototype from Wellner [9]. But the Digital Desk was mostly a "proof of concept" simulation and focused on the desktop environment rather than a shared whiteboard.

Stafford-Fraser's BrightBoard [6] is a fully functional system aimed at real-world usage. It allows the capture of strokes written on a whiteboard. Strokes can be either text or drawings supporting the meeting, or symbols used to send commands to the system. For example, a person can draw four angles of a rectangle that define the area to be printed. Then, the person draws a "P" letter inside a box. The BrightBoard system recognizes the "P letter in a box" as the "print" command, and captures and print what is inside the first rectangle. BrightBoard processes the output of a standard video camera which field of view is sufficiently narrow to obtain a usable capture resolution. This constraint puts a strong limit on the size of the active area of the whiteboard (the area where scanning and command issuing can occur). For the system feedback, BrightBoard emits simple sounds, thus it can only scan the information on the whiteboard, but not modify it (for a spatial reorganization for example).

Saund's Zombie Board system [5] alleviates the limitation on the size of the active area by the use of a steerable camera and image mosaicking techniques. We retain this approach and we will detail it in section 3.2. The drawback of the mosaicking approach is the induced delay of a scan: the camera has to move to orient itself toward the many pictures of the mosaic. This physical motion requires time to complete. More time consuming still is the processing that must be done on every image of the mosaic in order to recover the relative arrangement of the images, and to correct the perspective projection of the camera. As with BrightBoard, ZombieBoard recognizes special shapes drawn on the board as commands, such as "print". There is no system feedback, so the only possible interaction is to request a parameterized copy of the board: shape recognition allows the desired number of prints to be set, and to define a sub-region of the board that is to be printed. As with BrightBoard, it is not possible to interact with the strokes.

A-posteriori visual capture of strokes has gained interest also in the commercial world: the makers of the SoftBoard and the SmartBoard have both introduced a visual capture product in their product line ("CopyCam", "WhiteBoard Photo", "CamFire"). These products are based on one or more high definition (3 mega pixels) stills cameras attached at the end of an arm and oriented toward the board. Here again, the only service offered is the copy of the content (for saving, printing, or sending), but no interaction with the content is possible.

It is interesting to note that trajectory capture systems often provide visual feedback by means of a video projector, but visual capture systems don't. The rationale is that visual capture systems are more lightweight and non-intrusive than trajectory capture systems. Their main goal is to provide a content copy service while getting the computer out of the way.

We think that, even if copy is the service most often missing from a regular whiteboard, the spatial reorganization service is still very important for supporting creative meetings and thus interaction with the board's content must be provided. Our system, the Magic Table, introduces the facility for interacting with a digital copy of the strokes, thanks to the addition of two components: a video projector that allows visual feedback on the board, and a visual tracker the allows user pointing on the board. Moreover, we will show that the addition of a projector greatly simplifies the mosaicking problem, thus making it much faster.

Having motivated our work, we present in the next section the Magic Table services and the interaction that activates them. We report our observations on initial user experimentation with the system. In section 3 we detail the implementation of the two main components of the system: the token tracker and the whiteboard scanner. We conclude in section 4 with envisioned improvements.

2. The system

2.1. Hardware setup

The main components of the Magic Table are a regular whiteboard with its pens and eraser, a video projector and a video camera both oriented toward the board, and a computer. The projector is a 1024x768 DMD video projector. The camera is a computer controlled pan / tilt / zoom PAL (768x576 at 25 fps.) video camera. The computer is a standard bi-processor PC equipped with a graphics (for projector output) card and a video acquisition (for camera input) card. The system has been experimented with vertical (board) orientation and horizontal (table) orientation. We now favor the table orientation because it allows for more people interacting at the same time. A board allows only two persons to work in a comfortable way: a third person would occlude the work of the collaborators. A table allows at least four persons (one at each side of the table) working at the same time without obstructing each other's activity. Thus, the name of the system has shift from "Magic Board" to "Magic Table". A photo of the Magic Table is shown in figure 4.

The Magic Table setup has been recently enhanced with the addition of a second video camera, as shown in figure 1, in order to allow parallel operation of scanning and tracking services. Only one camera has to be steerable: the one devoted to scanning. The tracking camera has a fixed field of view set to encompass the entire projected image on the table. We have been working on tracking either bare fingers or plastic tokens, though until very recently the finger tracking was not robust enough for use in the Table system. In this paper, we will only report about interacting

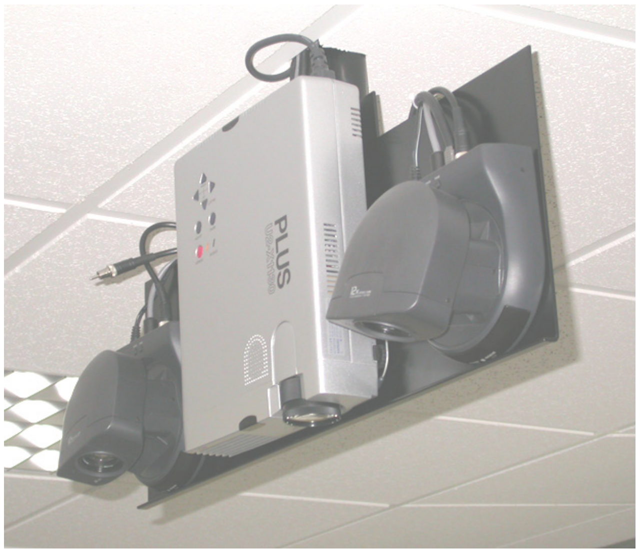

Figure 1: The Projector-Camera setup

using the tokens. The tokens are small (2 to 4 centimeter) disks of bright colored plastic. When used on a vertical board, the tokens are glued to small magnets so that they stick to the surface. At the moment, we use tokens of a single color, thus the color has no semantic; the color only purpose is to simplify the task of visually tracking the tokens. We envision further developments to track tokens of different colors and to associate a different semantic to each color.

2.2. Interaction

Users write, draw and erase on the Magic Table as on a regular whiteboard: it *is* primarily a regular whiteboard. When they need to access more advanced digital services, users *scan* physical strokes into *patches*. A patch is a rectangle that contains a digital copy of the physical strokes located at the initial location of the patch. Once created, the patch and its content can be moved around on the surface. Videos of the system in use are available for download from the web (http://iihm.imag.fr/demos/magicboard/).

Users interact with the Magic Table by manipulating the tokens. In the design of the interaction, we try to match the simplicity and the rapidity of working on a regular whiteboard. We define a small set of gestures that allow three fundamental operations on the patches: creation, transformation, and deletion. They are described in the three following paragraphs.

To create a patch, users first need to designate the area of interest to the system. They do so by picking up two tokens and putting them in contact as illustrated in figure 2. The system understands that a selection is initiated, and projects an elastic rectangle representing the selection

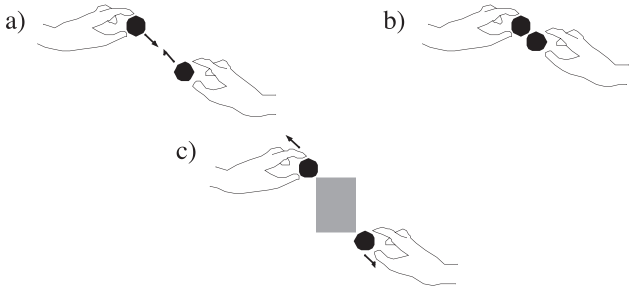

Figure 2: Creating a patch
Users grab 2 tokens (a), put them in contact (b), then stretch an elastic selection box (c).

between the two tokens. The rectangle is updated in real time according to the motion of the tokens. Here, the user can choose to either cancel the selection by hiding one of the tokens, or begin the scan by not moving (leaving alone) the two tokens for 2 seconds. Users are made aware of the beginning of a scan when the selection rectangle disappears. The scan camera then orients itself toward the selection and grabs one or more images depending on the size of the selection (in order to maintain a good scan resolution). Once the scan is finished, the patch is immediately projected on top of the physical strokes, informing the user that he can now manipulate it.

Patches can be applied three kinds of transformation: translation, rotation, and uniform scaling (scaling that does not modify the initial width / height ratio of the patch). Transformations are applied by *attaching* one or two tokens to a patch. Attaching a token to a patch is simply done by moving an unattached token on top of the patch limits. As soon as the token encounters the patch limits, feedback is displayed (the rectangular limits of the patch quickly flashes in red, as shown in figure 4) and a *control point* is defined on the patch at the exact location where the token went in contact with the patch. A token is detached from a patch by simply hiding the token. A control point can be defined *inside* the limits of a patch by bringing a token on top of the patch while *hiding it* in the hand, then showing the token to the system when at the desired location. As

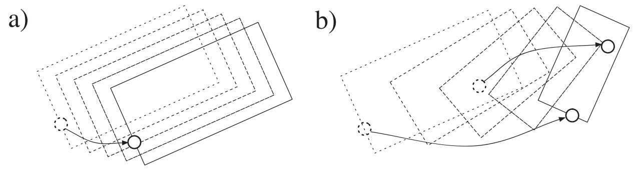

Figure 3: Patch transformations
The rectangle is the patch, the circles are the control points. Dashed drawing represent previous positions.
(a) one control point: translation,
(b) two control points: translation + rotation + uniform scaling

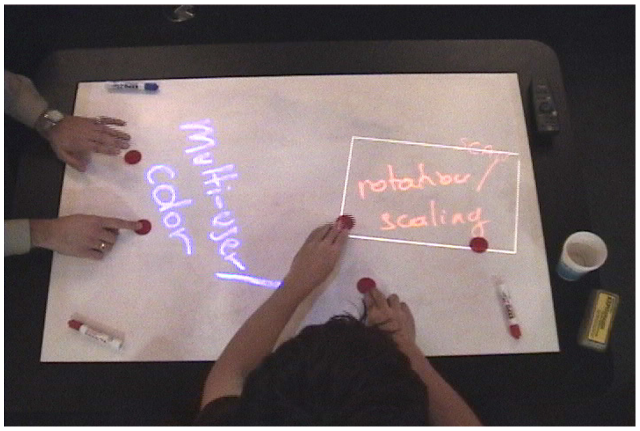

Figure 4: The Magic Table in use
Two users doing two-handed interaction.

shown in figure 3, one control point allows the translation of a patch, while two control points allow the simultaneous translation, rotation, and uniform scaling of a patch. A token already attached to a patch will not attach to other patches, until it is detached. A patch that has two tokens attached to it will not let other tokens attach to it, until at least one token is detached.

Deleting a patch is done by scaling it to zero: two tokens are attached to the patch, then put in contact.

2.3. Discussion

With the proposed interaction, we define a small set of quick gestures that activate the electronic services of the Magic Table. We make the most of the two-handed interaction and we avoid typical Graphical User Interface (GUI) elements such as buttons and menus. By doing so we approach our fundamental goals of simplicity and rapidity as explained below.

The simple gestures are easy to understand and learn: after learning one single (trivial) gesture (attaching a token to a patch) users can issue move, rotate, scale, iconify (move to the side while scaling down), and delete commands. Translations and rotations are done in the exact same manner as when moving or rotating a sheet of paper (equivalent of a patch) with one or two fingers. Scaling and deletion, even if not doable on a sheet of paper, are naturally guessed from translation and rotation without requiring any explanation.

The rapidity of the interaction is insured by the rapidity of the gestures themselves: the translation of a patch is done by a fast gesture where the token is put in contact with the patch then pushed toward the desired destination in a continuous gesture. There is no need for an object selection step (by clicking on it) or a command selection step (on a button or a menu) before being able to move the patch. Additionally, the unlimited number of tracked tokens allows to scale to multiple simultaneous users, as shown in figure 4.

At the moment, patches are not persistent: once deleted they are lost. Moreover, when users leave the room they have no way to get access to the work they did on the Table. In the conclusion of this paper, we will present the services we envision to offer information persistence and access. In its current state, the Magic Table is only a test bed for designing and testing new forms of interaction, as well as the computer vision techniques that support them.

2.4. Usage report

The Magic Table has been used many times during informal experiments, and at two formal creative meeting experiments. In one of the two formal experiment, users were asked to execute the same task on the Magic Table and on a SmartBoard.

One of the key missing element is the visual feedback on the tokens: once attached to a patch, there is no visual feedback informing that the token is attached. Thus, users often grab a token that they think is not attached when it actually is. This results in erroneous transformation of the attached patch. As this happens quite often, it becomes a source of frustration. Another problem is the natural tendency of users to move the tokens with one or two fingers covering much of the token. This causes the token tracker to fail. We hope that projecting a feedback on the token will entice users to put their finger at the periphery of the tokens. Finally, the non persistent characteristic of the patch has often been a problem, specially when a patch was inadvertently deleted.

On the good side, we observed that users were quick to understand the interaction and master it. We received a lot of positive feedback about the interaction with two hands and the possibility of several people to interact at the same time. When compared to the SmartBoard, user expressed their preference to the non computer-centric interaction of the Magic Table.

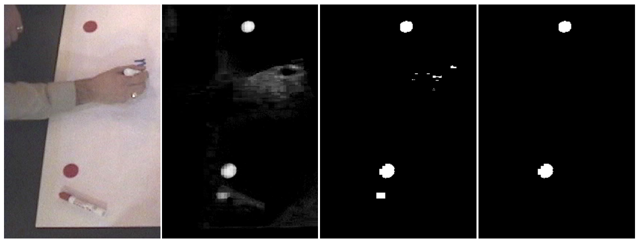

Figure 5: Image processing for token tracking Input image (a), Distance to model (b), Threshold (c), Connected component analysis and shape filtering (d)

Having presented our system and its interaction, we now get into the details of its internal components.

3. Implementation

3.1. Token tracker

Tokens are tracked in the video stream by color detection. A model of the color of the tokens is build from a set of sample pixels that are part of a token. In order to limit the effect of the variation of light intensity, we work in the (r,g) space instead of the (R,G,B) space, with

$$r = \frac{R}{R+G+B} \qquad g = \frac{G}{R+G+B} \tag{1}$$

The color model is the Gaussian distribution of the sample set in the (r,g) space. It is represented by a two-dimensional center $\boldsymbol{m}$ and a 2x2 covariance matrix $\boldsymbol{C}$. Token tracking is implemented as follow:

In the input camera image (figure 5a), every pixel is substituted by its Mahalanobis distance d to the color model, with

$$d = \sqrt{(\boldsymbol{p}-\boldsymbol{m})^T \cdot \boldsymbol{C}^{-1} \cdot (\boldsymbol{p}-\boldsymbol{m})} \tag{2}$$

where p if a (r,g) pixel. The resulting image is a map of the distance of each pixel to the color model (figure 5b). A distance threshold is empirically chosen and used to transform the distance map to binary (figure 5c). For efficiency, distance to model and threshold are precomputed for every possible (r,g) couple in a 256x256 lookup table. Thus, the binary image is obtained by projection of every (R,G,B) pixel into the (r,g) space (equation 1) followed by a simple lookup in the table.

From the binary image, non-null pixels are classified into connected components (sets of neighbor non-null pixels). Every component is then submitted to the shape filter. In our tracker, the shape filter simply checks for a minimum size (number of pixels) of the components. This minimum depend on the geometry of the scene. It must be smaller than the size of the tokens because tokens are often partially occluded by fingers. In our setup, we reject component that have a surface smaller than 40 pixels. The shape filter is efficient in filtering noise or small objects (such as a pen cap) as illustrated in figure 5d.

The output of the shape filter is a set of connected component that are considered as representing tokens. The algorithm in figure 6 updates the set of know tokens S, from time step t to time step t+1, by associating each component to a known token, or creating a new token, and deleting unseen tokens. The threshold values of the association algorithm are chosen empirically. We set the distance threshold *disT* to 40 pixel. It is the maximal possible

```
for each component c in the last image
    if there is no token in $S^t$
        create a new token at the center of c in $S^{t+1}$
    else
        compute the distance d between the center of c and
                                        the closest token t in $S^t$
        if d > disT
            create a new token at the center of c in $S^{t+1}$
        else
            if t is scheduled for deletion, remove schedule
            mark t as being seen
            if d > moT
                update location of t to center of c in $S^{t+1}$

for each token t in $S^t$
    if t is not marked as being seen
        schedule t for deletion in delT ms.
```

Figure 6: Token association algorithm (symbols are defined in the article)

translation for a token between two images. We set the motion threshold *moT* to 3 pixels. It is the minimal translation of a token that is being reported. This threshold avoid the static instability of the tokens due to the instability of the color detection. We set the deletion threshold *delT* to 500 ms. It is the maximum amount of time a token will exist even if it has not been seen in the images. This threshold avoids the phenomenon of disappearance and re-appearance of tokens when they are briefly occluded by a moving object (an arm for example).

The output of the token tracking system is a sequence of event records in the form <*Type*,*Id*,*x*,*y*,*t*> where *Type* is one of (Appear, Disappear, Motion), *id* is a unique number that remains the same in all events concerning a particular token between its appearance and its disappearance. *x* and *y* are the coordinates of the event, and *t* is the date of the event in millisecond. The token tracker extract the *x* and *y* coordinates of the tokens in the camera image reference. But tokens are used to control patches that are managed in the projector image reference. Thus, token coordinates are first transformed to projector reference before inclusion in the event record. The computation of the transformation is detailed in paragraph 3.3.

The token tracking system, despite its simple image processing and naive token tracking algorithm, is fairly robust. It has been used for working sessions lasting half a day. In environments where the light is unconstrained, such as rooms with windows on the outside, the color model often has to be re-computed at the beginning of the working session for the current light condition. In a constrained environment, once the tracker has been setup it doesn't need anymore tuning.

The motion threshold *moT* sensibly decreases the effective definition of the tracker to 120 x 80 in our setup. In the context of spatial reorganization of big patches, this is not an issue. But it will be an issue if more accurate controls are to be performed on the Table. In this case, it will be necessary to stabilize the output of the image processing.

3.2. Scanner

The scanner component of the Magic Table system is in charge of creating digital patch from physical strokes on the Table. This leads to solving two different problems: stroke segmentation and mosaicking.

Stroke segmentation is the classification of pixels of an image into two classes: stroke and background. This classification is necessary to improve the quality of the scanning (dust, texture and shadows are removed) and to allow transparency in patches: only the stokes are set opaque, the background is set transparent. Transparent patches allows their composition by superposition. In our system, we use

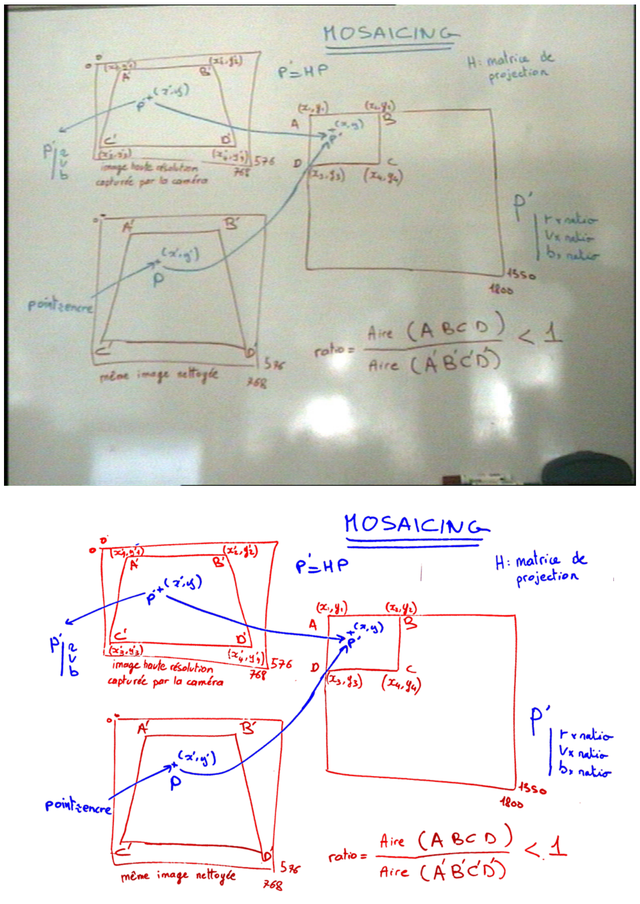
MOSAICING
P'=HP
H: matrice de projection
image haute résolution capturée par la caméra
même image nettoyée
point encre
ratio = Aire (ABCD) / Aire (A'B'C'D') < 1
576
768
1350
1800

Figure 7: Scanner sample result
Top: single 768x576 unprocessed image.
Bottom: result of a 3x3 mosaic, a 1800x1350 image.

Wellner's "Adaptive Thresholding" algorithm [8] to segment the strokes.

Mosaicking is necessary to scan high definition digital copies of the strokes from a (relatively) low definition camera. We use a PAL (768x576) pan / tilt / zoom camera with a 12x zoom lens. In our setup, a maximal zoom on the table yields a capture area of about 20 cm. in width, hence the capture resolution is about 100 dpi. A typical output of the scanner is shown in figure 7. With the increase of the definition of digital stills cameras, and the decrease of their cost, it may become practical to avoid mosaicking. In order to achieve 100 dpi on our 80 x 60 cm., the stills camera would need a sensor of about 7 M. pixels. Cameras capable of such a high definition are still very expensive, this is why we use a mosaic approach in our system.

Images of the mosaic are transformed by the perspective projection of the camera. The transformation is different for every image because it corresponds to different orientations of the camera. In order to "stitch" the mosaic images together, we need to correct this deformation. Traditional mosaicking approaches find corresponding features in the overlapping region between neighbor images. The corresponding features are used to compute the transformation between the images [5], [7] (figure 8). The search for corresponding features is computationally expensive. It may sometime fail when overlapping regions cover an empty region of the board. This problem can be limited by using bigger overlapping regions, but this increases the number of mosaic images, which in turn increases the scanning time.

We benefit from the availability of the video projector to greatly simplify the mosaicking problem. The projector allows the projection of a calibration pattern which size and location is known in the projector (table) reference. We

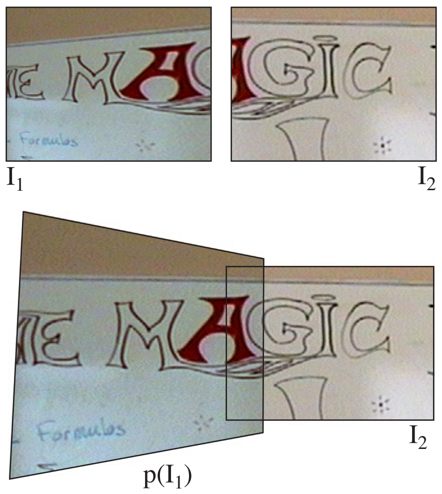

Figure 8: Image stitching for the mosaic I_1 and I_2 are the source images. $p(I_1)$ is the projection of I_1 in the I_2 reference.

recover the transformation between images and the projector references, rather than between neighbor images. This defines a fixed reference which simplifies the stitching of multiple images in two dimensions. The calibration pattern is chosen so that it is detectable by simple image processing, thus the search for features is greatly simplified and more robust: in our experiment, it almost never fails. The computation of the transformation between a camera image and the projector image is common to the tracked and the scanner. We detail it in the next paragraph.

3.3. Projector-Camera calibration

The tracking and scanning services of the Magic Table require that the transformation between the camera image and the projector image is known. In the two camera setup (as discussed in 2.1), one of the cameras is devoted to token tracking and has a fixed point of view. Hence, the transformation can be calibrated once for all. Re-calibration is only necessary if the relative position of the projector and the tracking camera is changed, which rarely occur with equipment fixed on the ceiling as shown in figure 1. On the other hand, the scanner camera is constantly steered to a different location of the table. Each time this occurs, a calibration is required.

In all cases, the transformation is assumed to be a perspective transformation between two planes (i.e. camera lens deformation is neglected). At least four couple of corresponding points are necessary to recover the transformation, or even more to achieve greater accuracy. A corresponding point is a couple of two-dimensional coordinates, each 2D coordinate representing the same physical point in the projector image and the camera image.

We project a grid of nine white rectangles on a black background in order to maximize the contrast (figure 9a). The captured image (figure 9b) is thresholded and a connected component analysis is executed. This results in a

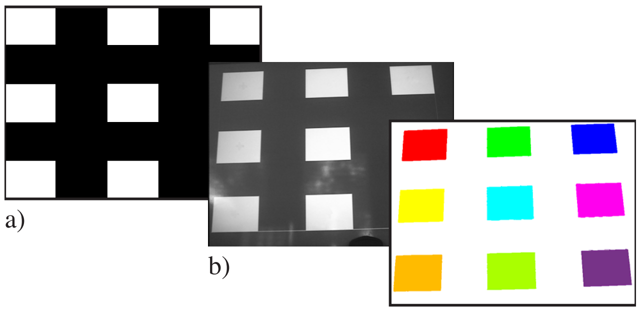

Figure 9: Grid detection (a) projected grid pattern, (b) as seen by the camera, (c) after thresholding and connected component analysis (each color is a different component)

set of nine connected component (figure 9c) for which the coordinates of the centers are computed. Centers are then sorted horizontally and vertically so that they can be paired with the corresponding center in the pattern (projector) reference. This yields to nine corresponding points which are used to compute the transformation according to the homogeneous estimation method as described in [1].

This calibration requires that we know where to orient the camera in order to capture an image of the grid. For the tracking camera this is not a problem: it is manually oriented once for all so that it sees the entire projected surface. For the scanning camera we build a lookup table that translates (*x*,*y*,*width*) to (*pan*,*tilt*,*zoom*), where (*x*,*y*) is the center of the area to be scanned, expressed in the projector reference, and *width* is the width of this area. The (*pan*, *tilt*, *zoom*) result of a lookup is the set of parameters that must be sent to the camera so that its field of view encompass the desired area. On request, the system builds this lookup table in an autonomous way by projecting white rectangles that are detected in the camera image by simple thresholding. The rectangles are projected in many different locations at many different sizes. The system iteratively orient the camera toward the rectangles, recording each time its pan, tilt, and zoom parameters to build the lookup table.

4. Conclusion

The Magic Board is an innovative attempt to complement a whiteboard surface with severely missing digital services such as copy and transformation of the content. Our strategy is to augment the surface only when and where required, in particular we don't modify what is working well: writing with pens. We implement a whiteboard scanner that is fast enough to be used interactively. The scanner is controlled by physical tokens, thanks to the token tracker that satisfies user-centered requirements of robustness and latency. The interaction that we implemented is designed to fit the characteristics of the whiteboard: easy and fast. Initial experiments show a strong acceptation of this approach from the users. But our system still lacks key services before being usable in real-world situation.

Our short term objectives are to correct small design errors such as the lack of visual feedback on the tokens, and to add the essential services of information saving and retrieving. Faithful to our "easy and fast" goals, we plan to display a *time line* that users could attach to with tokens in order to browse the history of the Table. Every patch and its transformations would be implicitly saved and users could walk away from the Table but still be able to browse its content from any remotely connected terminal (workstation or PDA). In parallel, other developments have lead to the implementation of a bare-finger tracker that will soon be integrated to the Table, we look forward to investigate new interactions with this new possibility.

The author wishes to thank Audrey Vidal and Brigitte Meillon for their participation in the evaluation of the system, and all the students and collaborators who worked on the Magic Board and Magic Table projects (the list is much too long to be reproduced here). This work is part of the FAME (Facilitating Agents in Multicultural Exchange) European Project.

5. References

[1] A. Criminisi, I. Reid, A. Zisserman, "*A plane measuring device*", in Proc. BMVC, UK, September 1997.

[2] S. Elrod, R. Bruce, R. Gold, D. Goldberg, F. Halasz, W. Janssen, D. Lee, K. McCall, E. Pedersen, K. Pier, J. Tang, B. Welch, "*Liveboard: a large interactive display supporting group meetings, presentations, and remote collaboration*", Proceedings of the SIGCHI conference on Human factors in computing systems, ACM Press, New York, NY, USA, 1992, pp. 599-607.

[3] T. Igarashi, S. Matsuoka, S. Kawachiya, H. Tanaka, "*Interactive beautification: a technique for rapid geometric design*", Proceedings of the 10th annual ACM symposium on User Interface Software and Technology (UIST), Banff, Alberta, Canada, October 14 -17, 1997, pp. 105-114.

[4] E. D. Mynatt, T. Igarashi, W. K. Edwards, and A. LaMarca, "*Flatland: New Dimensions in Office Whiteboards*", in Proceedings of CHI '99, ACM Press, Pittsburgh, PA, USA, May 1999, pp. 346-353.

[5] E. Saund, "*Bringing the Marks on a Whiteboard to Electronic Life*" Proc. Cooperative Buildings—Integrating Information, Organizations, and Architecture: Second Int'l Workshop, CoBuild '99 (Lecture Notes in Computer Science 1670), N. Streitz, J. Siegel, V. Hartkopf, and S. Konimi, eds., Springer, 1999.

[6] J. Q. Stafford-Fraser, "*Video-Augmented Environ-ments*" Doctor of Philosophy Thesis, Gonville & Caius College, University of Cambridge, february 1996.

[7] R. Szeliski, "*Image Mosaicing for Tele-Reality Applications*". Second IEEE Workshop on Applications of Computer Vision (WACV), Sarasota, Fl, 1994.

[8] P. Wellner, "*Adaptive Thresholding for the DigitalDesk*". Xerox Research Center Technical Report n. EPC-1993-110. 1993.

[9] P. Wellner, "*Interacting with paper on the DigitalDesk*". Communication of the ACM vol 36, n. 7, July 1993, p. 87-96.