Whiteboard It!

Convert Whiteboard Content into an Electronic Document

Zhengyou Zhang Li-wei He
Microsoft Research
Email: zhang@microsoft.com, lhe@microsoft.com

Aug. 12, 2002

Abstract

This ongoing project aims at converting the content on a physical whiteboard, captured by a digital camera, into an electronic document which can be easily manipulated as drawings in a MS Office document or as inks in a TabletPC. Through a series of image processing procedures, the originally captured image is first transformed into a rectangular bitmap with clean white background, and finally converted into a vectorized representation (inks).

Contents

1 Introduction

A whiteboard provides a large shared space for collaboration among knowledge workers. It is not only effective but also economical and easy to use – all you need is a flat board and several dry-ink pens. While whiteboards are frequently used, they are not perfect. The content on the whiteboard is hard to archive or share with others who are not present in the session. Our system aims at reproducing the whiteboard content as a faithful, yet enhanced and easily manipulable, electronic document through the use of a digital camera. It uses a series of image processing algorithms to clip the borders, rectify the whiteboard images and correct the colors.

2 Overview

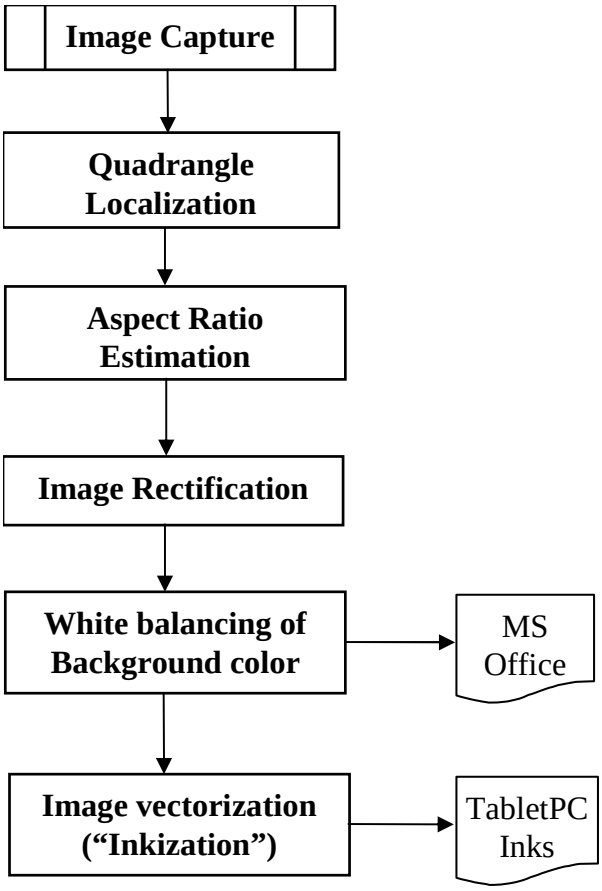


Figure 1: System architecture

The overview of the system is shown in Fig. 1. It starts by acquiring an image of the whiteboard with a digital camera.

The first step is to localize the borders of the whiteboard in the image. This is done by detecting four strong edges. The whiteboard usually appears to be a general quadrangle, rather than a rectangle, in an image because of camera's perspective projection. If a whiteboard does not have strong edges, an interface is provided for the user to localize the quadrangle manually.

The second step is to estimate the actual aspect ratio of the whiteboard from the quadrangle in the image. Besides the aspect ratio, we can also estimate the focal length of the camera. The theory is explained in the appendix.

The third step is image rectification. From the estimated aspect ratio, and by choosing the "largest" whiteboard pixel as the standard pixel in the final image, we can compute the desired resolution of the final image. A planar mapping (a $3\times$ 3 homography matrix) is then computed from the original image quadrangle to the final image rectangle, and the whiteboard image is rectified accordingly.

The fourth step is white balancing of the background color. This involves two procedures. The first is the estimation of the background color (the whiteboard color under perfect lighting without anything written on it). This is not a trivial task because of complex lighting environment, whiteboard reflection and strokes written on the board. The second concerns the actual white balancing. We make the background uniformly white and increase color saturation of the pen strokes. The output is a crisp image ready to be integrated with any office document.

The fifth step is image vectorization. It transforms a bitmap image into vector drawings such as free-form curves, lines and arcs. TabletPC inks use a vector representation, and therefore a whiteboard image after vectorization can be exported into TabletPC.

Note that at the time of this writing, we have not yet implemented the vectorization step.

3 Automatic Whiteboard Detection

As was mentioned in the introduction, this work was motivated by developing a useful tool to capture the whiteboard content with a digital camera rather coping the notes manually. If the user has to click on the corners of the whiteboard, we have not realized the full potential with digital technologies. In this section, we describe our implementation of automatic whiteboard detection. It is based on Hough transform, but needs a considerable amount of engineering because there are usually many lines which can form a quadrangle. The procedure consists of the following steps: 1. Edge detection; 2. Hough transform; 3. Quadrangle formation; 4. Quadrangle verification; 5. Quadrangle refining. Combining with the technique described earlier, we have a complete system for automatically rectifying whiteboard images. Experiments will be provided in subsection 3.2.

3.1 Technical Details

We describe the details of how a whiteboard boundary is automatically detected.

Edge detection. There are many operators for edge detection (see any textbook on image analysis and computer vision, e.g., [2, 4, 6]). In our implementation, we first convert the color image into a gray-level image, and use the Sobel filter to compute the gradient in x and y direction with the following masks:

$$G_x = \begin{array}{|c|c|c|}\hline -1 & -2 & -1 \\ \hline 0 & 0 & 0 \\ \hline 1 & 2 & 1 \\ \hline \end{array} \quad \text{and} \quad G_y = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$$

We then compute the overall gradient approximately by absolute values: $G = |G_x| + |G_y|$. If the gradient G is larger than a given threshold T_G, that pixel is considered as an edge. $T_G = 40$ in our implementation.

Hough transform. Hough transform is a robust technique to detect straight lines, and its description can be found in the books mentioned earlier. The idea is to subdivide the parameter space into accumulator cells. An edge detected earlier has an orientation, and is regarded as a line. If the parameters of that line fall in a cell, that cell receives a vote. At the end, cells that receive a significant number of votes represent lines that have strong edge support in the image. Our implementation differs from those described in the textbooks in that we are detecting *oriented* lines. The orientation information is useful in a later stage for forming a reasonable quadrangle, and is also useful to distinguish two lines nearby but with opposite orientation. The latter is important because we usually see two lines around the border, and if we do not distinguish them, the detected line is not very accurate. We use the normal representation of a line:

$$x\cos\theta + y\sin\theta = \rho\ .$$

The range of angle θ is $[-180^\circ, 180^\circ]$. For a given edge at (x_0, y_0), its orientation is computed by $\theta = \text{atan2}(G_y, G_x)$, and its distance $\rho = x_0\cos\theta + y_0\sin\theta$. In our implementation, the size of each cell in the $\rho\theta$-plane is 5 pixels by 2°.

Quadrangle formation. First, we examine the votes of the accumulator cells for high edge concentrations. We detect all reasonable lines by locating local maxima whose votes are larger than five percent of the maximum number of votes in the Hough space. Second, we form quadrangles with these lines. Any four lines could form a quadrangle, but the total number of quadrangles to consider could be prohibitively high. In order to reduce the number, we only retain quadrangles that satisfy the following conditions:

- The opposite lines should have quite opposite orientations (180° within 30°).
- The opposite lines should be quite far from each other (the difference in ρ is bigger than one fifth of the image width or height).
- The angle between two neighboring lines should be close to $\pm 90^\circ$ (within 30°).
- The orientation of the lines should be consistent (either clockwise or counter-clockwise).
- The quadrangle should be big enough (the circumference should be larger than $(W + H)/4$).

The last one is based on the expectation that a user tries to take an image of the whiteboard as big as possible.

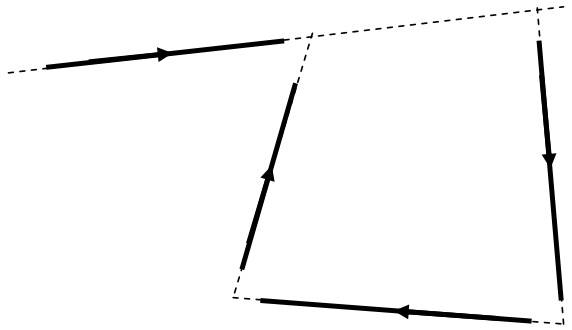


Figure 2: An example of bad quadrangles

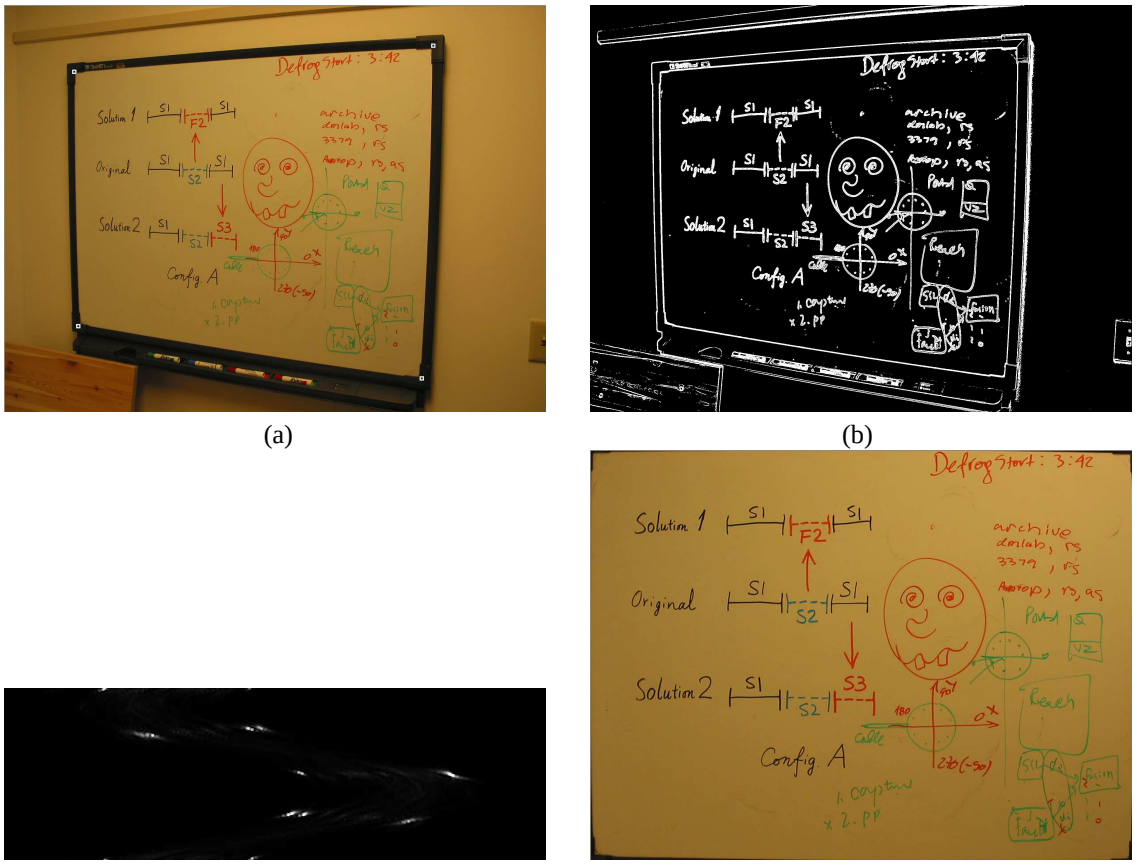
Quadrangle verification. The lines detected from Hough space are infinite lines: they do not say where the supporting edges are. For example, the four lines in Figure 2 would pass all the tests described in the previous paragraph, although the formed quadrangle is not a real one. To verify whether a quadrangle is a real one, we walk through the sides of the quadrangle and count the number of edges along the sides. An edge within 3 pixels from a side of the quadrangle and having similar orientation is considered to belong to the quadrangle. We use the ratio of the number of supporting edges to the circumference as the quality measure of a quadrangle. The quadrangle having the highest quality measure is retained as the one we are looking for.

Quadrangle refining. The lines thus detected are not very accurate because of the discretization of the Hough space. To improve the accuracy, we perform line fitting for each side. For that, we first find all edges with a small neighborhood (10 pixels) and having similar orientation to the side. We then use least-median squares to detect outliers [10], and finally we perform a least-squares fitting to the remaining good edges [2].

3.2 Experiments With Automatic Whiteboard Rectification

We have tested the proposed technique with more than 50 images taken by different people with different cameras in different rooms. All the tuning parameters have been fixed once for all, as we already indicated earlier. The success rate is more than 90%. The four failures are due to poor boundary contrast, or to too noisy edge detection. In this subsection, we provide three examples of success (Figures 3 to 5) and one example of failure (Figure 6).

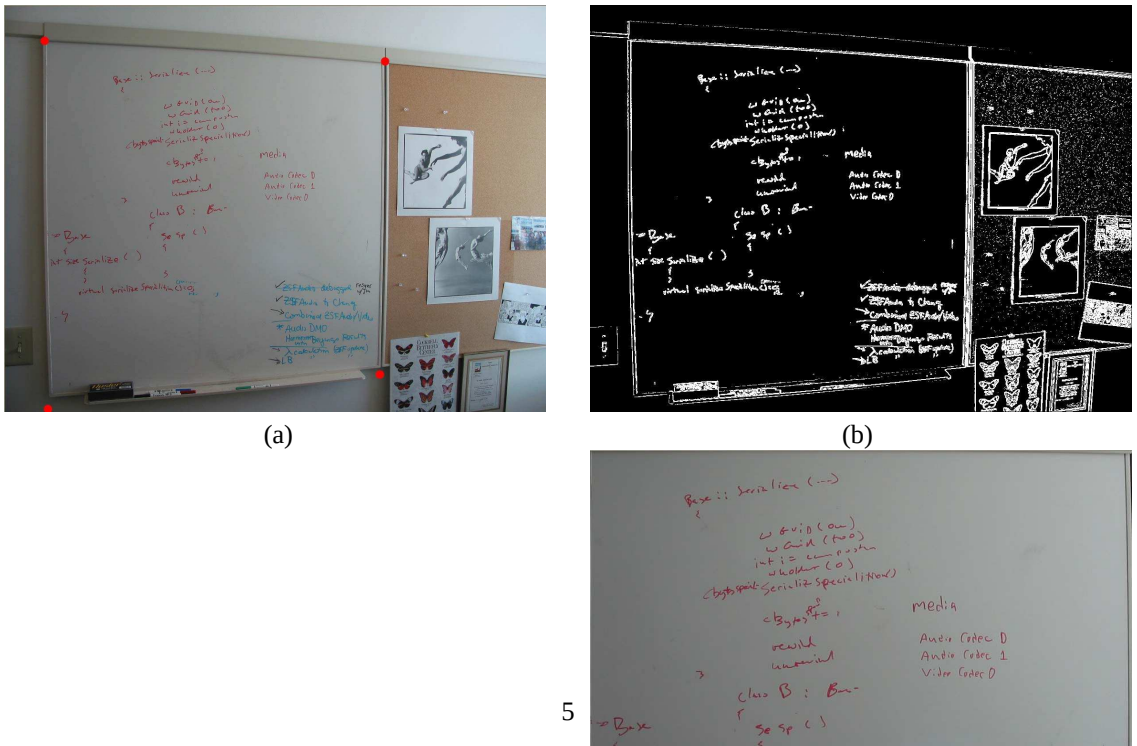
Figure 3 is a relatively simple example because the whiteboard boundary is very clear. The image resolution is 2272×1704 pixels. The detected edges are shown in white in Fig. 3b. As can be seen in



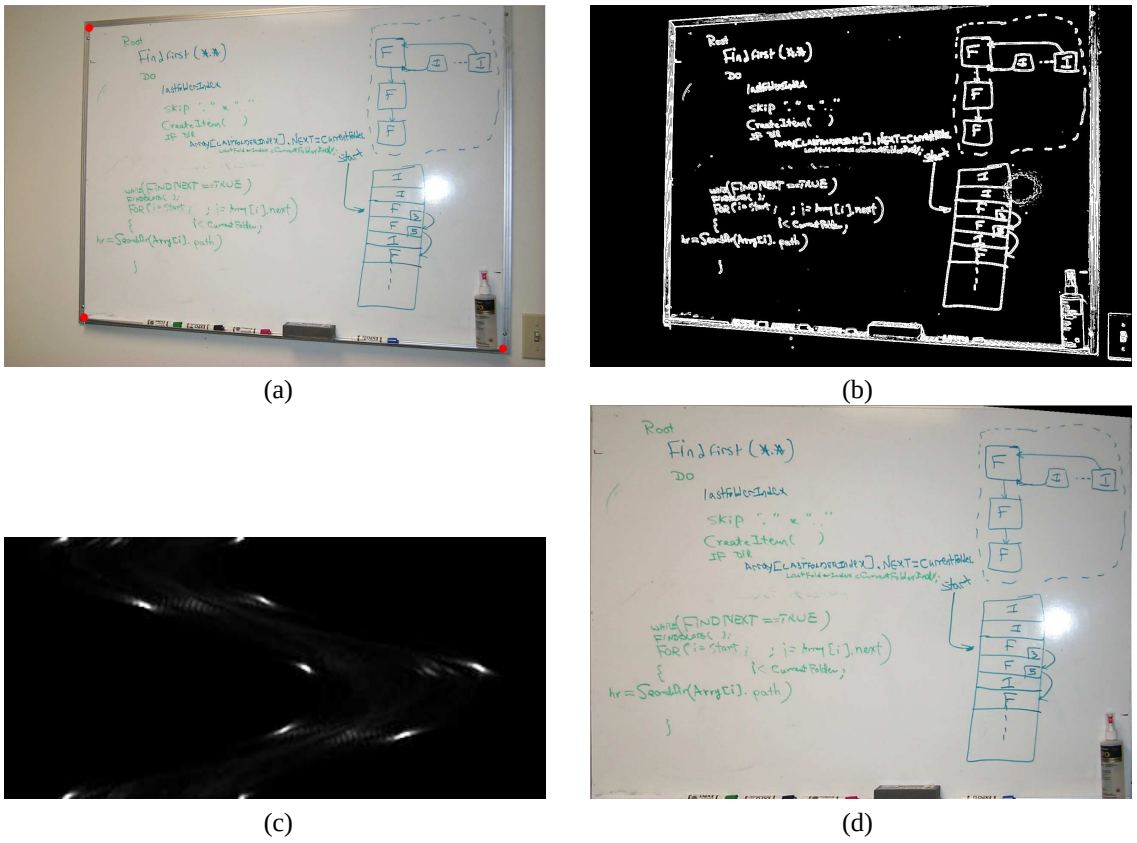
(a) (b)

(c) (d)

Figure 3: Example 1. Automatic whiteboard detection and rectification: (a) Original image together with the detected corners shown in small white squares; (b) Edge image; (c) Hough image with ρ in horizontal axis and θ in vertical axis; (d) Cropped and rectified whiteboard image.



(a) (b)



(a) (b) (c) (d)

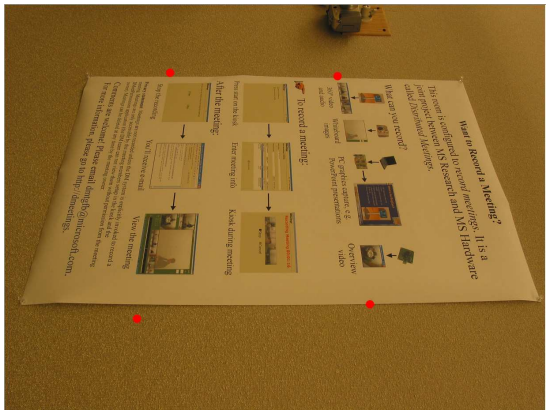
Figure 5: Example 3. Automatic whiteboard detection and rectification: (a) Original image together with the detected corners shown in small red dots (note that the upper right corner is outside of the image); (b) Edge image; (c) Hough image with ρ in horizontal axis and θ in vertical axis; (d) Cropped and rectified whiteboard image.

the Hough image (Fig. 3c), the peaks are quite clear. The corners of whiteboard are accurately estimated, as shown in small white squares in Fig. 3a. The cropped and rectified image is shown in Fig. 3d. The estimated aspect ratio is 1.326, very close to the ground truth $4/3$. The estimated focal length is 2149 pixels.

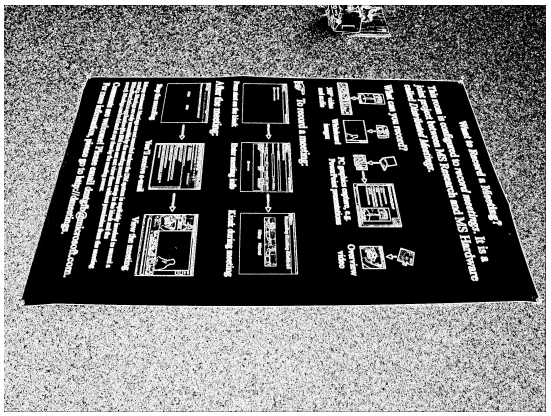
Figure 4 shows a different example. The image resolution is still 2272×1704 pixels. As can be seen in the edge image (Fig. 4), the actual lower border of the whiteboard does not have strong edge information. Our technique thus detects the line corresponding to the pen holder, which is perfectly reasonable. The whiteboard corners estimated by intersecting the detected lines are shown in small red dots in Fig. 4a. The cropped and rectified image is shown in Fig. 4d. The estimated aspect ratio is 1.038. The ground truth is 1.05 (the whiteboard is of the same type as in Fig. 8). Since the detected whiteboard includes the pen holder, the estimated aspect ratio (width/height) should be a little bit smaller than 1.05. The estimated focal length is 3465 pixels. We cannot compare the focal lengths because of different zoom settings.

Figure 5 shows yet another example. The resolution is 1536×1024 pixels. This example has one particular thing to notice: the upper right corner is not in the image. It does not affect the performance of our technique since we first detect boundary lines rather than corners. In Fig. 5a, the three detected visible corners are shown in small red discs. The fourth corner, although invisible, is also accurately estimated, as can be verified by the cropped and rectified image shown in Fig. 5d, where the invisible region (upper right corner) is filled with black pixels due to lack of information. The estimated aspect ratio is 1.378. We do not have the ground truth because the image was provided by an external person. The estimated focal length is 2032 pixels.

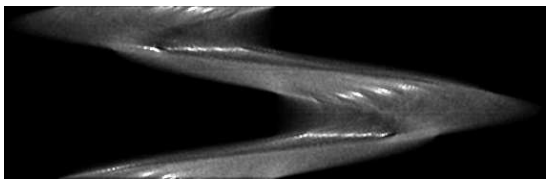
Our technique applies not only to whiteboard images but also to posters, as long as a poster's color is



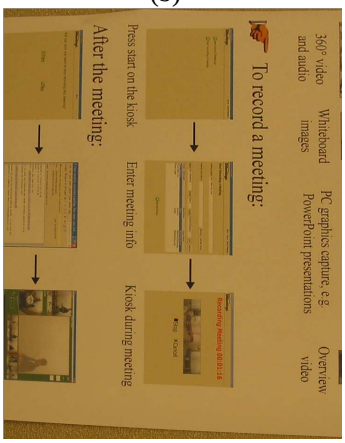
(a)



(b)



(c)



(d)

Figure 6: A failure example. (a) Original image together with the detected corners shown in small red dots; (b) Edge image; (c) Hough image with ρ in horizontal axis and θ in vertical axis; (d) Cropped and rectified whiteboard image.

distinct from the walls. Here, however, we show a failure example with a poster (Figure 6a). The failure is due to the fine texture on the wall. As can be seen in Figure 6b, the edge image is very noisy, and the number of edges is huge. The noise is also reflected in the Hough image (Figure 6c). The detected corners, as shown in red dots in Figure 6a, are not what we expected. For this example, we could apply a different edge detector or change the threshold to ignore the fine texture, but our point is to check the robustness of our technique without changing any parameters.

4 Estimating Pose and Aspect Ratio of a Rectangular Shape from One Image, and the Camera's Focal Length

Because of the perspective distortion, the image of a rectangle appears to be a quadrangle. However, since we know that it is a rectangle in space, we are able to estimate both the camera's focal length and the rectangle's aspect ratio.

4.1 Geometry of a Rectangle

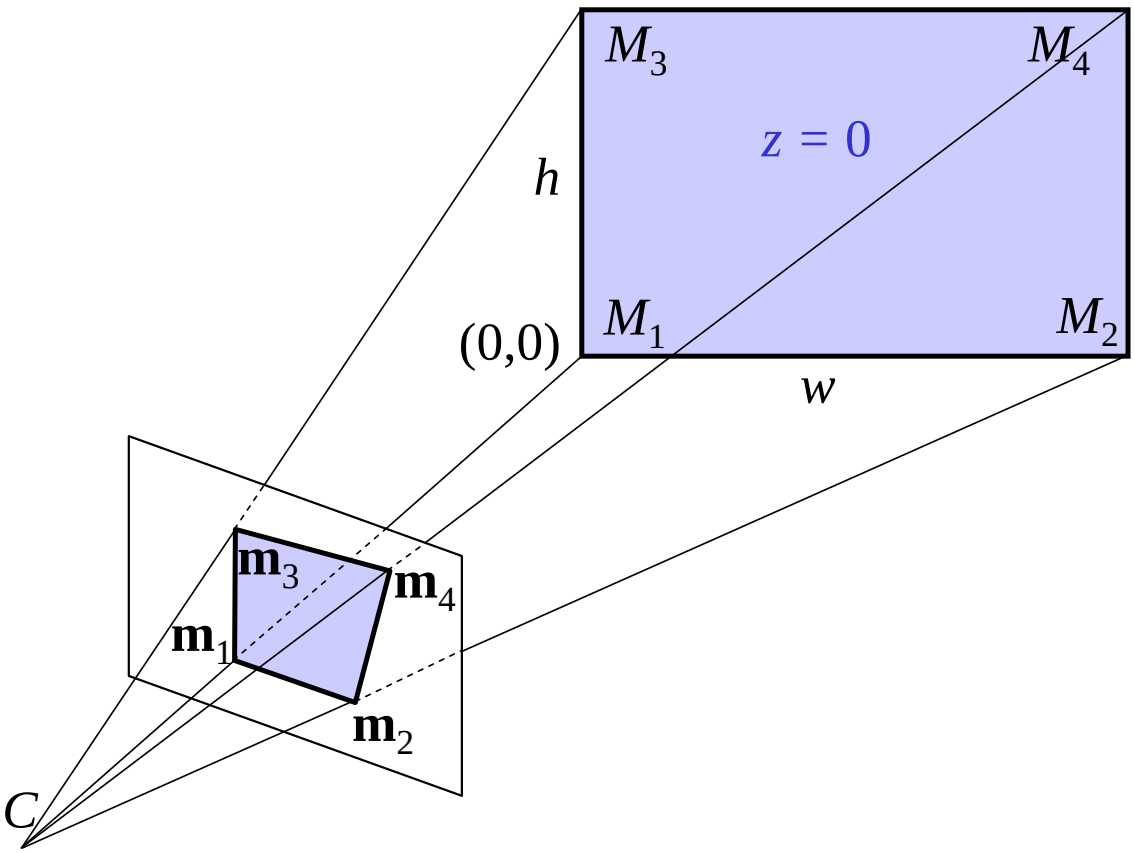


Figure 7: Geometry of a rectangle

Consider Figure 7. Without loss of generality, we assume that the rectangle is on the plane $z = 0$ in the world coordinate system. Let the width and height of the rectangular shape be w and h. Let the coordinates of the four corners, M_i ($i = 1, \ldots, 4$), be $(0, 0)$, $(w, 0)$, $(0, h)$, and (w, h) in the plane coordinate system ($z = 0$). The projection of the rectangle in the image is an quadrangle. The observed corners in the image are denoted by $\mathbf{m}_1$, $\mathbf{m}_2$, $\mathbf{m}_3$, and $\mathbf{m}_4$, respectively. Furthermore, we will use $\widetilde{\mathbf{x}}$ to denote the augmented $\mathbf{x}$ vector by adding 1 as the element to vector $\mathbf{x}$, i.e., $\widetilde{\mathbf{x}} = [x_1, \ldots, x_n, 1]^T$ if $\mathbf{x} = [x_1, \ldots, x_n]^T$.

We use the standard pinhole model to model the projection from a space point M to an image point $\mathbf{m}$:

$$\lambda \widetilde{\mathbf{m}} = \mathbf{A}[\mathbf{R} \quad \mathbf{t}]\widetilde{\mathtt{M}} \tag{1}$$

with

$$\mathbf{A} = \begin{bmatrix} f & 0 & u_0 \\ 0 & sf & v0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{R} = [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{r}_3]$$

where f is the focal length of the camera, s is the pixel aspect ratio, and $(\mathbf{R}, \mathbf{t})$ describes the 3D transformation between the world coordinate system, in which the rectangle is described, and the camera coordinate system. (In the above model, we assume that the pixels are not skew.) Substituting the 3D coordinates of the corners yields

$$\lambda_1 \widetilde{\mathbf{m}}_1 = \mathbf{A}\mathbf{t} \tag{2}$$
$$\lambda_2 \widetilde{\mathbf{m}}_2 = w\mathbf{A}\mathbf{r}_1 + \mathbf{A}\mathbf{t} \tag{3}$$
$$\lambda_3 \widetilde{\mathbf{m}}_3 = h\mathbf{A}\mathbf{r}_2 + \mathbf{A}\mathbf{t} \tag{4}$$
$$\lambda_4 \widetilde{\mathbf{m}}_4 = w\mathbf{A}\mathbf{r}_1 + h\mathbf{A}\mathbf{r}_2 + \mathbf{A}\mathbf{t} \tag{5}$$

Performing (3) - (2), (4) - (2) and (5) - (2) gives respectively

$$\lambda_2 \widetilde{\mathbf{m}}_2 - \lambda_1 \widetilde{\mathbf{m}}_1 = w\mathbf{A}\mathbf{r}_1 \tag{6}$$
$$\lambda_3 \widetilde{\mathbf{m}}_3 - \lambda_1 \widetilde{\mathbf{m}}_1 = h\mathbf{A}\mathbf{r}_2 \tag{7}$$
$$\lambda_4 \widetilde{\mathbf{m}}_4 - \lambda_1 \widetilde{\mathbf{m}}_1 = w\mathbf{A}\mathbf{r}_1 + h\mathbf{A}\mathbf{r}_2 \tag{8}$$

Performing (8) - (6) - (7) yields

$$\lambda_4 \widetilde{\mathbf{m}}_4 = \lambda_3 \widetilde{\mathbf{m}}_3 + \lambda_2 \widetilde{\mathbf{m}}_2 - \lambda_1 \widetilde{\mathbf{m}}_1 \tag{9}$$

Performing cross product of each side with $\widetilde{\mathbf{m}}_4$ yields

$$\mathbf{0} = \lambda_3 \widetilde{\mathbf{m}}_3 \times \widetilde{\mathbf{m}}_4 + \lambda_2 \widetilde{\mathbf{m}}_2 \times \widetilde{\mathbf{m}}_4 - \lambda_1 \widetilde{\mathbf{m}}_1 \times \widetilde{\mathbf{m}}_4 \tag{10}$$

Performing dot product of the above equation with $\widetilde{\mathbf{m}}_3$ yields

$$\lambda_2 (\widetilde{\mathbf{m}}_2 \times \widetilde{\mathbf{m}}_4) \cdot \widetilde{\mathbf{m}}_3 = \lambda_1 (\widetilde{\mathbf{m}}_1 \times \widetilde{\mathbf{m}}_4) \cdot \widetilde{\mathbf{m}}_3$$

i.e.,

$$\lambda_2 = k_2 \lambda_1 \qquad \text{with } k_2 \equiv \frac{(\widetilde{\mathbf{m}}_1 \times \widetilde{\mathbf{m}}_4) \cdot \widetilde{\mathbf{m}}_3}{(\widetilde{\mathbf{m}}_2 \times \widetilde{\mathbf{m}}_4) \cdot \widetilde{\mathbf{m}}_3} \tag{11}$$

Similarly, performing dot product of (10) with $\widetilde{\mathbf{m}}_2$ yields

$$\lambda_3 = k_3 \lambda_1 \qquad \text{with } k_3 \equiv \frac{(\widetilde{\mathbf{m}}_1 \times \widetilde{\mathbf{m}}_4) \cdot \widetilde{\mathbf{m}}_2}{(\widetilde{\mathbf{m}}_3 \times \widetilde{\mathbf{m}}_4) \cdot \widetilde{\mathbf{m}}_2} \tag{12}$$

Substituting (11) into (6), we have

$$\mathbf{r}_1 = \lambda_1 w^{-1} \mathbf{A}^{-1} \mathbf{n}_2 \tag{13}$$

with

$$\mathbf{n}_2 = k_2 \widetilde{\mathbf{m}}_2 - \widetilde{\mathbf{m}}_1 \tag{14}$$

Similarly, substituting (12) into (7) yields

$$\mathbf{r}_2 = \lambda_1 h^{-1} \mathbf{A}^{-1} \mathbf{n}_3 \tag{15}$$

with

$$\mathbf{n}_3 = k_3 \widetilde{\mathbf{m}}_3 - \widetilde{\mathbf{m}}_1 \tag{16}$$

From the properties of a rotation matrix, we have $\mathbf{r}_1 \cdot \mathbf{r}_2 = 0$. Therefore, from (13) and (15), we obtain

$$\boxed{\mathbf{n}_2^T \mathbf{A}^{-T} \mathbf{A}^{-1} \mathbf{n}_3 = 0} \tag{17}$$

Again, from the properties of a rotation matrix, we have $\mathbf{r}_1 \cdot \mathbf{r}_1 = 1$ and $\mathbf{r}_2 \cdot \mathbf{r}_2 = 1$. Therefore, from (13) and (15), we obtain respectively

$$1 = \lambda_1^2 w^{-2} \mathbf{n}_2^T \mathbf{A}^{-T} \mathbf{A}^{-1} \mathbf{n}_2 \tag{18}$$

$$1 = \lambda_1^2 h^{-2} \mathbf{n}_3^T \mathbf{A}^{-T} \mathbf{A}^{-1} \mathbf{n}_3 \tag{19}$$

Dividing these two equations gives the aspect ratio of the rectangular shape:

$$\boxed{\left(\frac{w}{h}\right)^2 = \frac{\mathbf{n}_2^T \mathbf{A}^{-T} \mathbf{A}^{-1} \mathbf{n}_2}{\mathbf{n}_3^T \mathbf{A}^{-T} \mathbf{A}^{-1} \mathbf{n}_3}} \tag{20}$$

This equation says clearly that the absolute size of the rectangle cannot be determined from an image. This is obvious since a bigger rectangular shape will give the same image if it is located further away from the camera.

4.2 Estimating Camera's Focal Length and Rectangle's Aspect Ratio

In the last subsection, we derived two fundamental constraints (17) and (20). We are now interested in extracting all useful information from the quadrangle in the image.

We do not assume any knowledge of the rectangle in space (i.e., unknown width and height). Since we only have two constraints, we have to assume some knowledge of the camera. Fortunately, with modern cameras, it is very reasonable to assume that the pixels are square (i.e., $s = 1$) and the principal point is at the image center (i.e., known u_0 and v_0). Given u_0, v_0 and s, we are able to compute the focal length f from equation (17). Indeed, we have

$$\begin{aligned} f^2 = & -\frac{1}{n_{23} n_{33} s^2} * \\ & \{[n_{21} n_{31} - (n_{21} n_{33} + n_{23} n_{31}) u_0 + n_{23} n_{33} u_0^2] s^2 \\ & + [n_{22} n_{32} - (n_{22} n_{33} + n_{23} n_{32}) v_0 + n_{23} n_{33} v_0^2]\} \end{aligned} \tag{21}$$

where n_{2i} (resp. n_{3i}) is the i-th component of $\mathbf{n}_2$ (resp. $\mathbf{n}_3$). The solution does not exist when $n_{23} = 0$ or $n_{33} = 0$. It occurs when $k_2 = 1$ or $k_3 = 1$, respectively.

As soon as f is estimated, the camera's intrinsic parameters are all known, and the aspect ratio of the rectangle is readily computed by equation (20).

(Equation (20) can be used in a different way. If the aspect ratio of the rectangle is given, we can also use that equation to estimate the focal length. Together with (17), we'll have two equations to estimate the focal length, leading a more reliable estimation. However, this is not what we assume in this work.)

Once $\mathbf{A}$ is known, the pose of the rectangular shape can be determined. We have from (13)

$$\mathbf{r}_1 = \mathbf{A}^{-1} \mathbf{n}_2 / \|\mathbf{A}^{-1} \mathbf{n}_2\| \tag{22}$$

and from (15)

$$\mathbf{r}_2 = \mathbf{A}^{-1} \mathbf{n}_3 / \|\mathbf{A}^{-1} \mathbf{n}_3\| \tag{23}$$

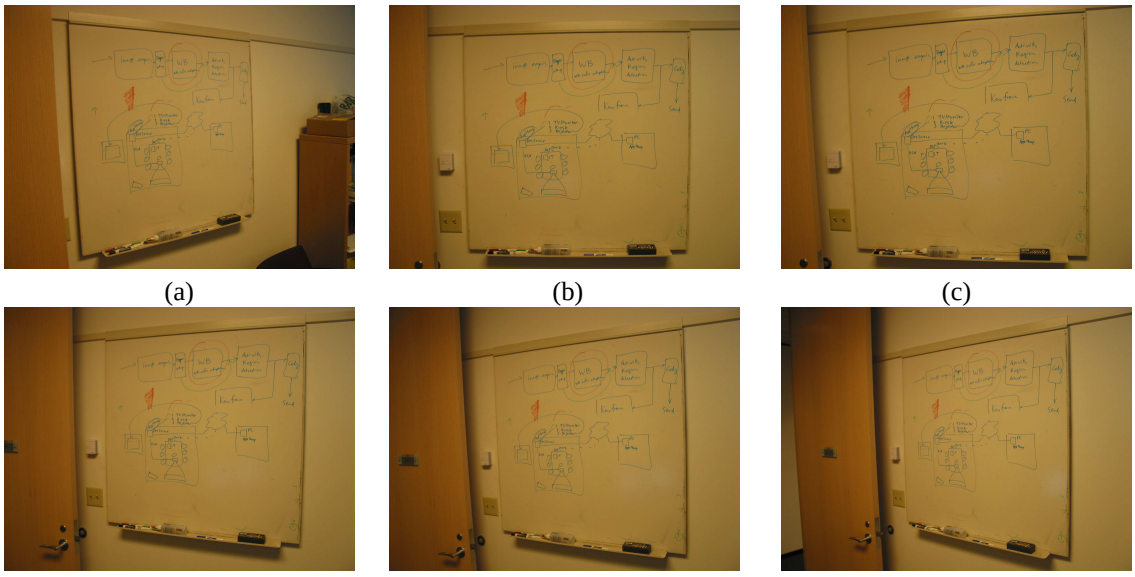
In turn,

$$\mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2 \tag{24}$$

The translation vector can be determined from (2), i.e.,

$$\mathbf{t} = \lambda_1 \mathbf{A}^{-1} \widetilde{\mathbf{m}}_1 \tag{25}$$

Note that the translation can only be determined up to a scale factor λ_1, which depends on the size of the rectangle as can be seen in (18) and (19). This is obvious since a bigger rectangular shape will give the same image if it is located further away from the camera.



(a) (b) (c)

(d) (e) (f)

Figure 8: Six images of the same whiteboard, taken from different angles

Table 1: Results with images shown in Figure 8

image	(a)	(b)	(c)	(d)	(e)	(f)
aspect ratio	1.028	1.035	1.031	1.021	1.019	0.990
error (%)	2.1	1.4	1.8	2.8	3.0	5.7
bounding box	0.966	1.035	0.981	0.892	0.843	0.727
difference (%)	5.1	1.4	6.6	15.1	19.7	30.8
focal length	2202	2442	2073	2058	2131	2030

4.3 Experimental Results on Aspect Ratio Estimation

In this section, we provide experimental results with real data. Six images of the same whiteboard, as shown in Figure 8, were taken from different angles. The most frontal view is image (b). We manually measured the whiteboard with a ruler, and the size is about 42'$\times$ 40' (note: 1'$\approx$2.54cm). The aspect ratio is therefore 1.05, and we use this as the ground truth.

In each image, we manually clicked on the four corners of the whiteboard, and use the technique described in the last section to estimate the focal length of the camera and the aspect ratio of the whiteboard. The results are shown in Table 1. The second row shows the estimated values of the aspect ratio, while the third row shows its relative error compared with the ground truth. The error is mostly less than 3%, except for image (f) which was taken from a very skewed angle. There are two major sources contributing to the errors: the first is the precision of the manually clicked points; the second is lens distortion that is currently not modeled. Lens distortion can be clearly observed in Figure 8. The error of the estimated aspect ratio tends to be higher for images taken from a larger angle. This is expected because the relative precision of the corner points is decreasing. For reference, we also provide the aspect ratio of the bounding box of the whiteboard image in the fourth row of Table 1, and its relative difference with respect to the ground truth in the fifth row. The relative difference can go up to 30%. It is clear that it is not reasonable to use the aspect ratio of the bounding box to rectify the whiteboard images. The sixth row of Table 1 shows the estimated focal length, which varies around 2200.

5 Rectification

The next task is to rectify the whiteboard image into a rectangular shape with the estimated aspect ratio. For that, we need to know the size of the final image. We determine the size in order to preserve in the rectified image maximum information of the original image. In other words, a pixel in the original image should be mapped to at least one pixel in the rectified image. Refer to Figure 9. The side lengths of the quadrangle in the original image are denoted by W_1 and W_2 for the upper and lower sides, and by H_1 and H_2 for the left and right side. Let $\hat{W} = \max(W_1, W_2)$ and $\hat{H} = \max(H_1, H_2)$. Let $\hat{r} = \hat{W}/\hat{H}$. Denote the estimated aspect ratio by r. We determine the size of the rectified image as follows: $W = \hat{W}$ and $H = W/r$ if $\hat{r} \geq r$; otherwise, $H = \hat{H}$ and $W = rH$. Once the size is determined, the rectifying matrix **H** (homography) can be easily computed, and the color in the rectified image is computed through bilinear or bicubic interpolation from the original image.

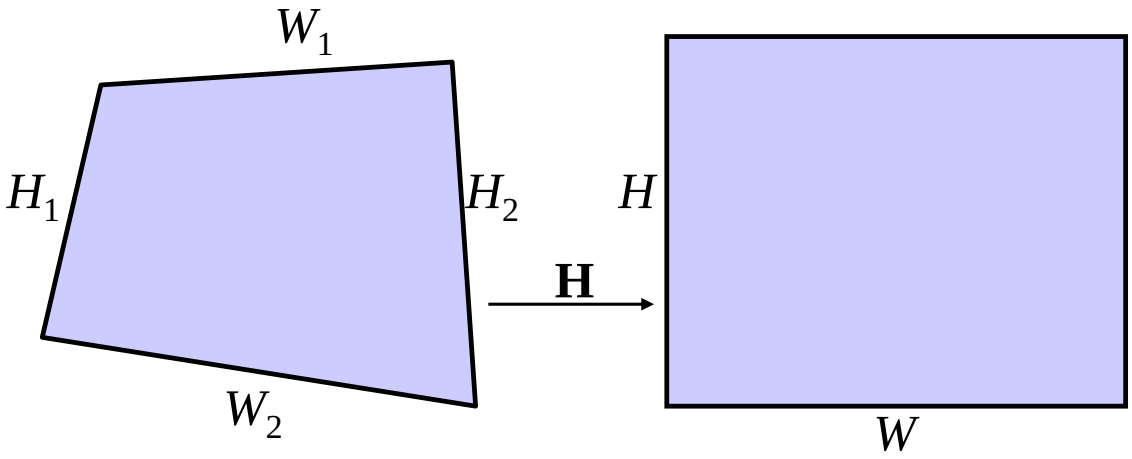
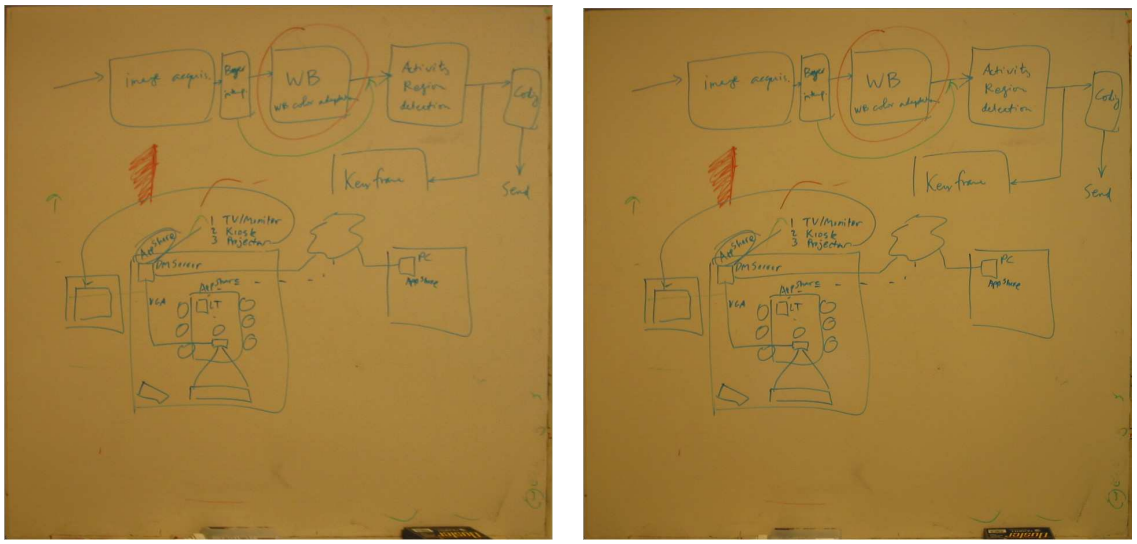


Figure 9: Rectification of an whiteboard image. Left: original shape; Right: rectified shape



(a) (b)

Figure 10: Rectified version of the first two images shown in Figure 8, using the estimated aspect ratios

Figure 10 shows two rectified images of the whiteboard using the estimated aspect ratios. They correspond to images (a) and (b) in Figure 8. The rectified images look almost identical despite that the original

images were taken from quite different angles. Other rectified images are also similar, and are thus not shown.

6 White Balancing

The goal of color enhancement is to transform the input whiteboard image into an image with the same pen strokes on uniform background (usually white). For each pixel, the color value C_{input} captured by the camera can be approximated by the product of the incident light C_{light}, the pen color C_{pen}, and the whiteboard color C_{wb}. Since the whiteboard is physically built to be uniformly color, we can assume C_{wb} is constant for all the pixels, the lack of uniformity in the input image is due to different amount of incident light to each pixel. Therefore, the first procedure in the color enhancement is to estimate C_{light} for each pixel, the result of which is in fact an image of the blank whiteboard.

Our system computes the blank whiteboard image by inferring the value of pixels covered by the strokes from their neighbors. Rather than computing the blank whiteboard color at the input image resolution, our computation is done at a coarser level to lower the computational cost. This approach is reasonable because the blank whiteboard colors normally vary smoothly. The steps are as follows:

1. Divide the whiteboard region into rectangular cells. The cell size should be roughly the same as what we expect the size of a single character on the board (15 by 15 pixels in our implementation).

2. Sort the pixels in each cell by their luminance values. Since the ink absorbs the incident light, the luminance of the whiteboard pixels is higher than stroke pixels'. The whiteboard color within the cell is therefore the color with the highest luminance. In practice, we average the colors of the pixels in the top 25 percentile in order to reduce the error introduced by sensor noise.

3. Filter the colors of the cells by locally fitting a plane in the RGB space. Occasionally there are cells that are entirely covered by pen strokes, the cell color computed in Step 2 is consequently incorrect. Those colors are rejected as outliers by the locally fitted plane and are replaced by the interpolated values from its neighbors.

Once the image of the blank whiteboard is computed, the input image is color enhanced in two steps:

1. Make the background uniformly white. For each cell, the computed whiteboard color (equivalent to the incident light C_{light}) is used to scale the color of each pixel in the cell: $C_{out} = \min(1, C_{input}/C_{light})$.

2. Reduce image noise and increase color saturation of the pen strokes. We remap the value of each color channel of each pixel according to an S-shaped curve: $0.5 - 0.5 * \cos(C_{out}^{p}\pi)$. The steepness of the S-curve is controlled by p. In our implementation, p is set to 0.75.

7 Examples

In this section, we provide a few results.

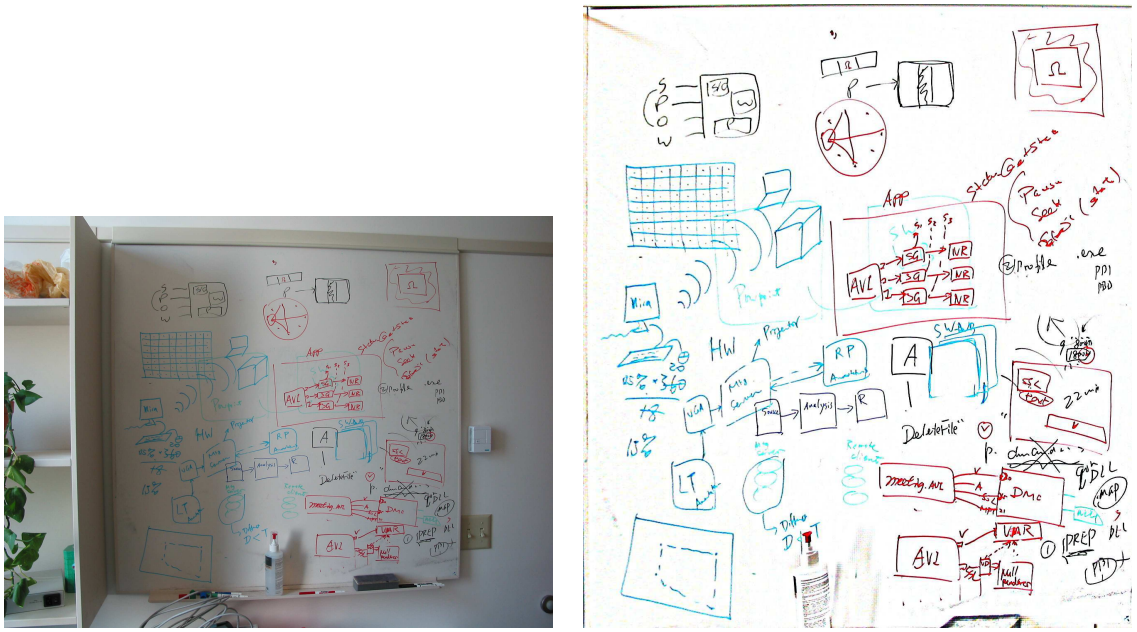
Figure 11 shows our system working on whiteboard with complex drawings and complex lighting condition.

Figure 12 shows that our system also works on a poster.

Figure 13 shows two images of the same whiteboard but taking from very different angles. The aspect ratio estimated from both images is very close to the ground truth.

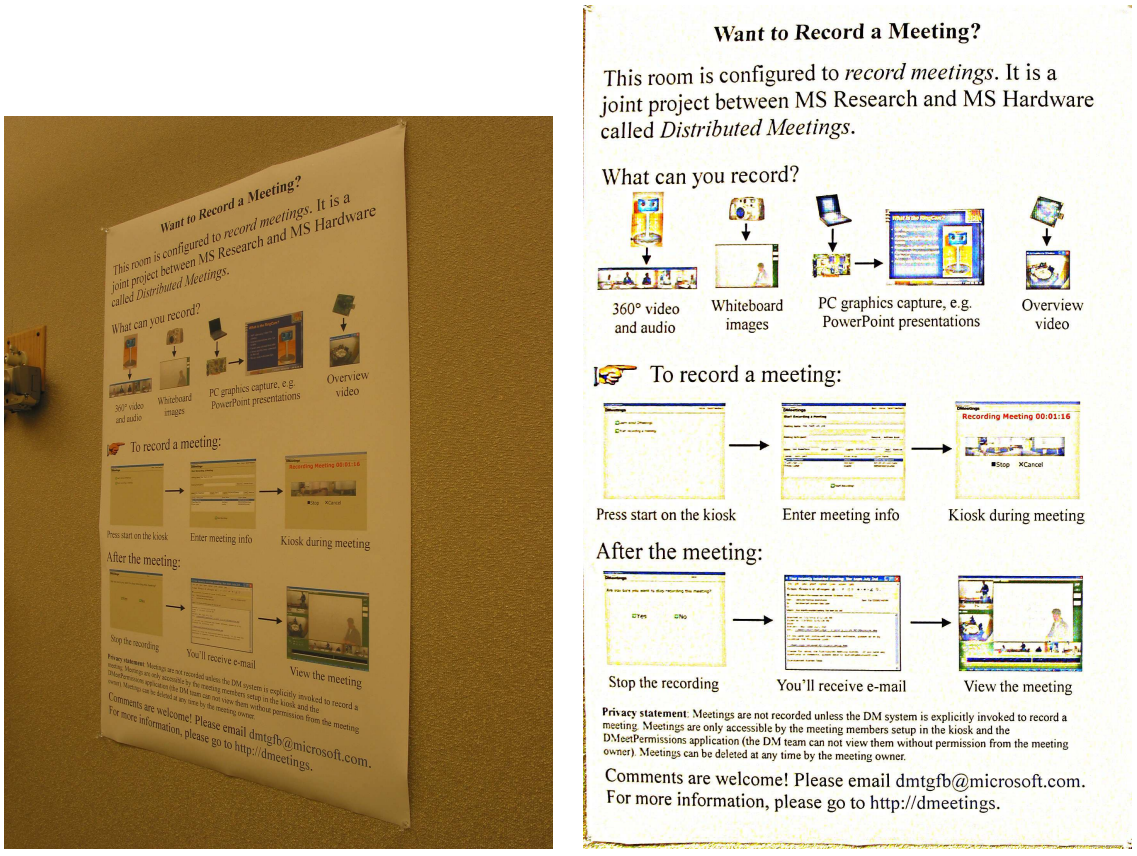
8 Competitive Product

The only commercial product we are aware of is ***Whiteboard Photo*** from PolyVision (http://www.websterboards.com/products/wbp.html). The list price is $249.00. Compared with our system, it lacks two features:



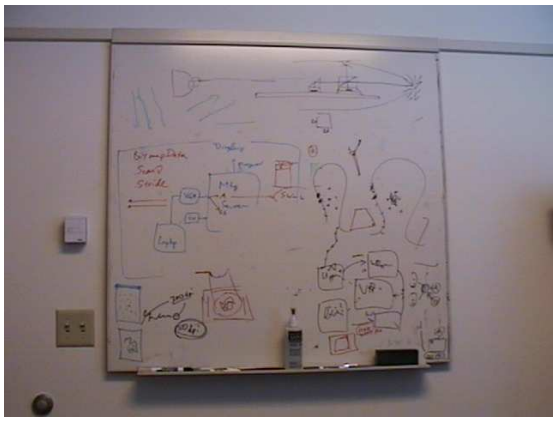
(a) (b)

Figure 11: An example with complex drawings on the whiteboard: (a) original image; (b) transformed one.

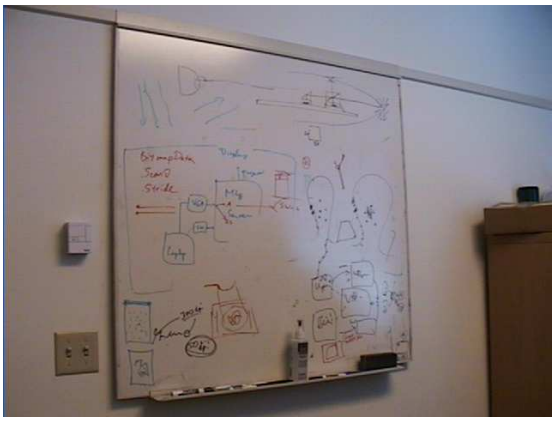


(a) (b)

Figure 12: An example with a poster: (a) original image; (b) transformed one.



(a)



(b)

Figure 13: Two images of the whiteboard in Zhengyou's office: (a) from the front; (b) from the side. The estimated aspect ratio is 1.06 for (a) and 1.056 for (b) while the manually measured aspect ratio is 1.05 (the whiteboard size is 42'$\times$ 40'). The estimated focal length is 842 pixels for (a) and 781 pixels for (b): image (b) was taken with a small zoom out.

- Aspect ratio estimation: Whiteboard Photo uses either the original image size or the aspect ratio of the bounding box for the final image, therefore aspect ratio of the final image does not correspond to the actual aspect ratio of the whiteboard.
- Vectorization: Whiteboard Photo does not produce a document in vectorized form, therefore it is impossible to manipulate individual or group of strokes, or to perform handwriting recognition.

9 Technical Challenges

There are a number of interesting research issues to be dealt with:

- Optimal detection of background pixels. This is a must in order to produce a perfect electronic document with complete white background. This is challenging because of complex lighting condition, whiteboard reflection, camera noise, etc.
- Stroke color enhancement. The stroke color from a digital camera usually has significant distortion because of Bayer interpolation, sensor variation, etc. For example, the green color is usually washed-out.
- Optimal filtering for resampling. In rectifying images, an optimal filtering needs to be designed to take into account the fact that whiteboard contents contain mostly thin strokes.
- Image vectorization. New techniques should be developed to deal with issues such as image noise, inherent lack of precision in a bitmap image, free-form writing, etc.

References

[1] A. Criminisi, *Accurate Visual Metrology from Single and Multiple Uncalibrated Images*, Springer-Verlag, 2001.

[2] O. Faugeras, *Three-Dimensional Computer Vision: A Geometric Viewpoint*, MIT Press, 1993.

[3] O. Faugeras and Q.-T. Luong, *The Geometry of Multiple Images*, MIT Press, 2001.

[4] R.C. Gonzalez and R.E. Woods, *Digital Image Processing*, 2nd edition, Prentice Hall, 2002.

[5] R. Hartley and A. Zisserman, *Multiple View Geometry*, Cambridge University Press, 1998.

[6] R. Jain, R. Kasturi, and B.G. Schunck, *Machine Vision*, McGraw-Hill, Inc., 1995.

[7] K. Kanatani, *Geometric Computation for Machine Vision*, Oxford University Press, 1993.

[8] C. Lin and R. Nevatia, “Building Detection and Description from a Single Intensity Image”, *Computer Vision and Image Understanding*, 72(2):101–121,1998.

[9] PolyVision, *Whiteboard Photo*, http://www.websterboards.com/products/wbp.html

[10] Z. Zhang, Parameter estimation techniques: a tutorial with application to conic fitting”, *Image and Vision Computing*, 15(1):59–76, 1997.