time, it is difficult to capture and highlight the attention of the audience.

1.3 Content Analysis in Lecture Videos

To produce high quality videos, expert cameramen and professional editors are usually required to handle the capture and editing work. This process is impractical in most cases due to the associated costs and labourious work involved. The advances in content-based video analysis, nevertheless, have brought new opportunities for the automatic indexing and editing of lecture videos due to two facts. Firstly, the classroom environment is structured and this makes it easier to detect the dynamic changes such as moving objects and handwritten annotations. Secondly, the captured videos are usually associated with external textual documents (e.g., PowerPoint). The linking of videos and documents could be accomplished by exploiting the relationship between visual, audio and texts.
comparing the OCR outputs of different textboxes, multi-frame integration
achieves significantly higher recognition accuracy than linear interpolation.
Figure 2.3 shows another example that compares our approach with the
algorithm proposed in [20]. As seen in this figure, in lecture videos, when
the text and background are both static, the averaged textbox from multiple
frames does not help most of the time. Our approach employs the local
information in color distribution for high-resolution reconstruction, which
proves useful in this case.

![Diagram of OCR with multi-frame integration and super-resolution]

Figure 2.2: Super-resolution reconstruction for OCR. (a) & (b) are textboxes in
low resolution, (d) & (g) are the reconstructed high resolution text boxes, (e) &
(h) are the binarized textboxes, (c), (f) & (i) are OCR output characters.

2.4 Text Recognition

Since most OCR systems use binary images as input, binarization is a pre-
processing step of text recognition. Given a high-resolution textbox, the task
is to determine whether the pixels belong to foreground characters or just lie
Figure 2.3: Comparison between our approach and the algorithm proposed in [20]. (a) and (b) are two text boxes in low resolution, (c) is the averaged textbox by the algorithm in [20]. (d) is the high-resolution textbox reconstructed by our approach. (e)-(h) are the binarized textboxes of (a)-(d).

in the background scene. The high-resolution texts usually have distinguishable colors between the foreground and background, and also have a high intensity contrast in a gray scale image. This makes it easy to segment text and describe the characters using the marginal distribution in a color space.

We utilize R/G/B/H/I components for text binarization. Figure 2.4 shows the pixel intensity distribution of a textbox in I space. The foreground mean $\mu_f$, background mean $\mu_b$, foreground variance $\sigma_f$, and background variance $\sigma_b$ are calculated for each component. Then the GMM (Gaussian mixture model) parameters of a text box are calculated and they can reflect how well each component is in segmenting and describing character properties. Each component is associated with a confidence as follows:

\begin{align}
C_i &= \frac{|\mu_b^i - \mu_f^i|}{\sigma_b^i + \sigma_f^i} \\
C_H &= \frac{\min(|\mu_b^H - \mu_f^H|, 256 - |\mu_b^H - \mu_f^H|)}{\sigma_b^H + \sigma_f^H}
\end{align}
where $i = \{R, G, B, I\}$. The higher the value $C$, the more confident the corresponding component. The component with the highest confidence is selected to carry out the segmentation of foreground texts and background scene. As shown in Figure 2.4, we select the value, $\frac{\mu_I + \mu_B}{2}$, for binarization.

![Figure 2.4: The intensity distribution of a textbox in I space.](image)

The binarized text boxes are fed to OCR system for character recognition. In our experiment, we use the commercial OCR system in [64] and the recognition results can be found in Table 2.1.

### 2.5 Experiments

We conduct experiments on five videos taped in different classrooms. The duration of each video is about 45 to 60 minutes. The five videos consist of nine different presentations. For each shot in the videos, we evenly extract five keyframes along the time dimension. The textboxes from multiple frames are integrated and reconstructed as one high-resolution textbox before text binarization. The binary textboxes are then fed to the OCR system. Fig-
Figure 2.5 shows the detected text boxes of several keyframes. We can see that when the background is not too complicated, the text detection algorithm works well. Some noise may be included if the text connects with other edges.

Figure 2.5: Experimental results for video text detection.

Figure 2.6 shows the binarized high-resolution textboxes of a keyframe. Figure 2.7 further shows some of the high-resolution textboxes obtained from keyframes in Figure 2.5. In fact, either low or high-resolution, most of the
The Handshake Problem every other person shakes hands once with.
There are n people in a room. If each person shakes hands with every other person, what is the total number h(n) of handshakes?

Recursion Fibonacci Numbers Other Recursive. Binary search: element of the array:
be produced from a. How many pairs of rabbits can single pair in Assumptions:
new pair of offspring every month, fertile at the age of one month.

Just as one student can, there is a price we have to pay for recursion: binary search to

Figure 2.7: Some of the high-resolution text boxes extracted from the video frames shown in Figure 2.5.

Table 2.1: Results of video text recognition (High Resolution)

<table>
<thead>
<tr>
<th>Lecture</th>
<th>Title</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video</td>
<td>$N_g$</td>
<td>$N_c$</td>
</tr>
<tr>
<td>1</td>
<td>620</td>
<td>494</td>
</tr>
<tr>
<td>2</td>
<td>230</td>
<td>218</td>
</tr>
<tr>
<td>3</td>
<td>560</td>
<td>515</td>
</tr>
<tr>
<td>4</td>
<td>849</td>
<td>792</td>
</tr>
<tr>
<td>5</td>
<td>705</td>
<td>673</td>
</tr>
</tbody>
</table>

($N_g$: number of ground-truth characters, $N_c$: number of correctly recognized characters, $N_{ocr}$: number of characters output by OCR, $N_h$: number of characters recognized by human.)

27
Gaussian, multiple Gaussian and Bayes have been proposed to model the skin color distribution. These models, in general, require a large amount of training data for classifiers. A relatively straightforward approach to build a skin classifier is by explicitly defining (through a number of rules) the boundaries of skin cluster in a color space. Here we adopt the rules in [35] for our application. A pixel \( \{R, G, B\} \) is classified as skin color if \( \{R > 95, G > 40, B > 20, |R - G| > 15, R > G, R > B\} \) or \( \{R > 220, G > 210, B > 170, |R - G| \leq 15, R > B, G > B\} \). One major advantage of this approach is that a rapid classifier can be easily constructed. The classified skin-color pixels are further clustered and those clusters with too small or too large sizes are excluded. Figure 3.1(a) shows one example where a hand gesture is successfully detected by our approach.

![Gesture detection and tracking](image)

Figure 3.1: Gesture detection and tracking

Once a gesture is detected, it is tracked after every three frames. The tracking is based on skin-color detection and clustering. The detected skin-color pixels are grouped into several clusters. A gesture is the cluster that satisfies the following criteria: 1) near the old location; 2) approximately equal dimensions; and 3) similar color distribution. Figure 3.1(b) depicts the moving path of a tracked gesture.
The purpose is to look for the possible closed circles. The path between the two end points should have large enough height and width to avoid pointing gestures, which also need to be satisfied when checking conditions 3 and 4.

The observed path is separated into several parts at these breakpoints. Each resulting part may be one of the following shapes or gestures: circle, line, pointing, or some non-gesture movement. The portions that last too short a duration (less than 10 frames) are excluded as non-gesture. Also, the first and last parts are regarded as entering and leaving slide region. All the segmented path portions will fall into 4 classes: circle, line, pointing and non-gesture. Figure 3.2 illustrates the procedure. Fig 3.2(a)-(c) are 3 frames on the gesture path. Four points, $A, B, C, D$ are found by Condition 1, and $E, F$ by 2. $IA, BC, ED$ and $DO$ are excluded after segmentation.

![Figure 3.2: Gesture Segmentation](image)
3.3.2 Speech in Gesture Detection

Relations Between Visual, Speech and Slides

In Figure 3.6, when a gesture is pointing to a paragraph in the slide, a keyword “model” is found in speech, which lies in the paragraph interacting with the gesture. We employ the correspondence between gesture and speech to improve the performance of gesture detection and reduce the response time.

As illustrated in Figure 3.6, electronic slides are used to find out the relationship between speech and visual. In Chapter 2, we synchronize videos with electronic slides by text analysis. The spatial correspondence between them is also discovered by computing the homography based on the corresponding points in the video and slides. After registration, the spatial relationship between video and the slides can be easily realized. In this work, the transcripts of speech are generated by an ASR (Automatic Speech Recognition) engine and matched with texts extracted from the slides. The video texts interacting with the gesture are obtained by the spatial correspondence between the video and the slides. Gesture-speech correspondence is then detected by video text and speech matching.
Gesture-Speech Correspondence

As shown in Figure 3.6, if a candidate gesture is detected and a gesture-speech correspondence is found, it is more likely that an intentional gesture is present. A confidence value is calculated based on this correspondence.

With a PowerPoint slide as example in Figure 3.6, the paragraphs are semantically grouped into separate object instances. By constructing the one-to-one mapping between the instances in slides and videos through homography projection, we can easily organize and structure the layout of videos. In Figure 3.6, the slide region in videos is partitioned into different ROIs (Region Of Interest). Each ROI is a unit that may be pointed by a gesture. The transcript of video texts in the ROIs can be obtained by the registered
proposed approach has potential use for pose estimation in other videos, and for other human activity recognition problems. Finally we describe how the head pose is used for lecture video editing to estimate the focus of lecturing.

Recent works related to our approach include [34, 39]. Both approaches aim for real-time camera management. In [39], basic techniques such as frame difference are used and thus only simple gestures can be recognized to determine the focus of lecturing. In [34], head pose is simply estimated based on the number of skin color pixels lying on each side of a face. Most existing approaches including [34, 39] do not address the issues of lighting conditions particularly for classrooms equipped with LCD projected screens and a presenter interacting with the screen.

4.2 Video Preprocessing

In this section, we describe our algorithm for robust face detection and tracking. Skin color has been shown to be a reliable cue for face detection in videos and color images [17]. To rapidly locate candidate faces, we adopt a rule-based classifier in [35] to efficiently detect skin color pixels. The detection is carried out in a multi-resolution manner. Potential pixels are initially de-
human face, some clusters are excluded based on ellipse size, ratio of height and width, skin color density and color variance.

4.3 Offline Pose Estimation

4.3.1 Face Detection and Tracking

Given the skin clusters, we use facial features (eyes and mouths) to verify the candidate face regions. Given a candidate cluster $\mathcal{R}$, a filtered image $\mathcal{F}$ is computed by

$$\mathcal{F} = g \ast \mathcal{R}_{pu}$$  \hspace{1cm} (4.1)

where

$$\mathcal{R}_{pu} = |\mathcal{R} - \text{open}(\mathcal{R})| + |\mathcal{R} - \text{close}(\mathcal{R})|$$  \hspace{1cm} (4.2)

g is a Gaussian filter and $\ast$ represents convolution. The $\text{open}(\mathcal{R})$ and $\text{close}(\mathcal{R})$ are the gray-scale morphological open and close operators respectively. The $|\mathcal{R} - \text{open}(\mathcal{R})|$ enhances the peaks in a cluster while $|\mathcal{R} - \text{close}(\mathcal{R})|$ enhances the valleys. For a candidate region with face, the facial features will be highlighted by these two operators. Figure 4.3(a) shows a candidate face,
while (b) shows $\mathcal{R}_{pv}$ with peaks and valleys and (c) shows the resulting filtered image $\mathcal{F}$.

Figure 4.3: Face detection: (a) Candidate cluster with face, (b) peaks and valleys, (c) filtered image.

We use a face template (see Figure 4.4(a)) to locate the facial features on the filtered image $\mathcal{F}$. A face template is adaptively generated based on a given candidate region and its morphological filtered image $\mathcal{R}_{pv}$. The template is basically formed by two bounding boxes $P$ (Figure 4.3(a)) and $B$ (Figure 4.3(b)) that minimally enclose a face region and the peaks and valleys of $\mathcal{R}_{pv}$ respectively. Three search regions $B_R$, $B_L$ and $B_M$ are adaptively defined, based on the orientation of a face induced by $P$ and $B$, to effectively locate the eyes and mouth. The search regions are determined by the centers $C_P$ and $C_B$ of $P$ and $B$ respectively. Let $C_P = (C_{Px}, C_{Py})$ and $C_B = (C_{Bx}, C_{By})$, we define a deviation term $\Delta$ as

$$\Delta = \alpha \frac{C_{Bx} - C_{Px}}{W_P} \frac{W_B}{W_B}$$  \hspace{1cm} (4.3)$$

where $W_P$ and $W_B$ are respectively the width of $P$ and $B$. The parameter $\alpha = 1.1$ is an empirical constant estimated from a face database of 300 images in different head poses. The term $\Delta$ estimates the degree of deviation from a frontal face. Ideally, $\Delta = 0$ if a lecturer directly faces to the front. Based on
Δ, the width of $B_R$ and $B_L$ are respectively $\frac{w_R}{2} + \Delta$ and $\frac{w_R}{2} - \Delta$ as shown in Figure 4.4(a). When a face turns to left, for instance, the width of $B_L$ will be relatively narrow than $B_R$. The center of $B_M$ is at $C_B x + \Delta$ while its width is $\frac{w_M}{2}$. In each search region, five points with maximal values are selected from $\mathcal{F}$. The three points, one from each region, that fit the triangle\(^1\) best (see Figure 4.4(a)) are selected as the locations of facial features. Figures 4.4(b) and (c) show two examples of locating facial features. In the detection phase, all the candidate skin clusters are tested and the clusters that do not show salient facial features are excluded.

Figure 4.4: Facial feature tracking: (a) adaptive face template, (b) and (c) located facial features.

Once a face is detected, its template is used to continuously track the facial features in every three frame. The tracking is based on skin-color detection and the continuous update of face template by $P$, $B$ and $\mathcal{F}$. A smoothness constraint inferred from the old feature locations is imposed on

\(^1\)The best triangle is the one formed by the centers of $B_R$, $B_L$ and $B_M$.  

57
the face template to ensure the robustness of tracking. Basically, the new located features should be near to old ones. In case the facial points found are not salient enough, or cannot be found, the old locations are used as the new ones. This happens frequently when a lecturer faces to an extreme left or right direction and some facial features are hidden. By using the old locations, the lecturer is assumed to face the same direction as the previous frame, which is the case in most time. Figure 4.5 shows the results of tracking facial features when the lecturer turns from frontal direction to the extreme left.

![Figure 4.5: Face tracking](image)

4.3.2 Pose Estimation

For pose estimation, face orientation is quantized into 3 different poses from extremely left to right direction. We extract parameters from the face template (Figure 4.4(a)) for head pose estimation by neural network. Let $E_L$, $E_R$ and $M$ be the detected facial feature positions. The parameters are generated from the set \{ $P, B, C_P, C_B, E_L, E_R, M$ \}. Any pair from the set gives two parameters: the length and direction of the line connecting them. Four additional parameters are the ratios of widths and heights of two rectangles:
face tracking and pose recognition. Once the value of $\hat{n}$ is fixed, five pose templates $T_{\hat{n}}^{\rho_i}, 1 \leq i \leq 5$, are directly used to fit and estimate the pose of the tracked face in the remaining frames.

### 4.4.3 Self-Adaptive Estimation

With the low resolution and complex lighting conditions, the direct pose estimation from one single image is usually not reliable enough. We exploit the temporal smoothness of head movement to refine and improve pose estimation. The probability of a pose at frame $t + 1$ can be inferred by the pose at frame $t$ and the head movement between frames $t - 1$ and $t$. By maximizing the confidence of pose transitions over frames, the state or pose $s_{t+1}$ at frame $t + 1$ is estimated as

$$C = C(F_{t+1}, s_{t+1}) \prod_{k=t-7}^{t} C(F_k, s_k)p(s_{k+1}/s_k s_{k-1})$$

where $C(F_k, s_k)$ is the confidence value of a face $F_k$ in frame $k$ with pose $s_k$, and $p(s_{k+1}/s_k s_{k-1})$ is the probability of pose $s_{k+1}$ in frame $k + 1$ when the two previous poses are $s_{k-1}$ and $s_k$. We use eight frames to estimate $C$. When estimating the pose at $t + 1$, we use the confidence values, rather than
Figure 5.1: Synchronization of lecture videos and slides

consist of nine different presentations. All the external documents are prepared by speakers with PowerPoint. Basically a variety of master templates are used in different presentations. In the first three videos, most slides contain only texts. In the last two videos, most slides are mixed with texts, images, tables and figures. The flipping time of most slides involves only two frames. The first three videos consist of one speaker each, while the last two videos consist of three different presenters respectively. Each speaker in the last two videos presents for approximately 20 minutes.

In our approach, the effectiveness of synchronization is heavily dependent on the recognition of characters in videos. No temporal assumption or other visual feature such as shape or color is used for matching. The selected test videos were actually taped in different classrooms of varying settings.
Figure 6.2: Structuring video content (a) with the text layout of its electronic slide (b)

which is denoted as

\[ Ah = 0 \]  \hspace{1cm} (6.4)

This is a problem of linear least squares. We minimize

\[ ||Ah||^2 = (Ah)^T Ah = h^T A^T Ah \]  \hspace{1cm} (6.5)

where \( h \) is the eigenvector of \( A^T A \) with the smallest eigenvalue. Since \( h \) is only defined up to a scale, we just solve for the unit vector \( h \). Eqn (6.5) requires four or more points.

The corresponding points between shots and slides are found by utilizing the positions of recognized texts. Since the positions of recognized video texts are known during video-slide synchronization in Chapter 2 and 5, they are matched with the content and positions of texts extracted from electronic slides. Because recognition of titles is usually more precise, we only utilize titles in slides for registration, which are enough for solving \( h \).

After registration, the spatial relationship between a video and its external document can be easily realized. With a PowerPoint slide as example, the
non-text regions. The result is shown in Figure 6.3(c). After text detection, the whiteboard is segmented into several regions by grouping neighboring textboxes into the same region. Figure 6.3(d) shows two resulting regions on the whiteboard.

6.4.2 Focus Estimation

Video text, gesture and posture are three major visual cues for lecture activity analysis. Text is closely related to the materials or the content presented in the lecture, while gesture and posture describe how the lecturer presents them to the students. In Chapter 2, 3 and 4, we have presented our algorithms for text, gesture and posture analysis. In this section, the three cues are jointly
initially extracted from the registered slide, by which we can precisely locate the corresponding pointed textbox in the video frame. Because the pointed textbox is aimed to be displayed at the center of the edited video, we can easily compute the transformation to zoom. The slide image to be projected is automatically extracted from the external document. The resolution of the slide image is normally higher than a video frame. In Figure 6.4, the value of a pixel $p$ in the edited frame is the calculus of a small region $R$ in the slide image. To augment the presenter in front of the edited frame, object segmentation is done prior to the composition. Since we adopt static camera setting, the presenter can be easily detected and tracked over frames by motion segmentation.
<table>
<thead>
<tr>
<th>(a) Original frame</th>
<th>(b) Detected difference</th>
<th>(c) Detected text region</th>
</tr>
</thead>
<tbody>
<tr>
<td>(d) Zoomed image</td>
<td>(e) Contrast-enhanced image</td>
<td>(f) Edited whiteboard image</td>
</tr>
</tbody>
</table>

Figure 6.5: Edited whiteboard video by enhancing handwriting visual quality.
Figure 6.6: Left: some snapshots of the original videos; Right: the corresponding ones in the edited video.
7.1 System Setup

Instead of using a touch screen, we use a slide screen to simulate most functions of a smartboard. Figure 7.2 illustrates the setup of the system. The classroom is equipped with an LCD projector. During a presentation, the uploaded slides are projected to the screen. To realize the human-computer interaction, a video camera is mounted at the back of the classroom to capture the screen and the presenter. The camera is connected to a computer through an IEEE 1394 cable to transfer the captured video for processing.

Figure 7.1(b) shows the interface projected to the screen. When the application is started, uploaded electronic slides are displayed at the lefttop of the screen as shown in Figure 7.1(b). Different tools, including the buttons for circling, lining, rubber, forward, backward, and a scrolling bar are displayed for the presenter to use. To interact with the computer, the presenter may
use hand or a laser pen to press the buttons for different commands. The interactions between the user and the objects on the screen are captured by the video camera and transferred to the computer. When the commands are recognized by the computer, corresponding operations are performed on the projected slides.

![Diagram showing system setup]

Figure 7.2: System setup
• *Forward:* when this button is pressed, the next slide will be shown (see Figure 7.3).

• *Backward:* when this button is pressed, the previous slide will be shown.

• *Scroll bar:* the slide can be scrolled up or down by pressing the up/down arrows.

![Figure 7.3: Slide control](image)

Besides the buttons, the presenter may make simple annotations by drawing or writing on the screen with a laser pen as illustrated in Figure 7.4. The path of the laser pen will be tracked and displayed in the slide. Due to the precision and efficiency considerations, we do not support annotations or handwritings of small scales. The laser pen should not be moved too fast. Some examples are available online [55].

Another function is to highlight some objects in the slide by making defined gestures (*circling*, *lining*, and *pointing*). When such an intentional
gesture is detected and recognized, the specific regions will be shown with a higher resolution. An example is shown in Figure 7.5.

7.4 Experiments

We have conducted experiments to calculate the accuracy and responsiveness of gesture detection and laser pen tracking using the simulated smartboard for presentation. In this section, the response time is calculated as the number of frames in the captured videos. The experiments are conducted on 3 hours of videos with 24 frames per second.
7.4.1 Gesture Detection

Accuracy

Table 7.1 shows the accuracy of gesture detection. For button pressing, few gestures are missed due to occlusion. Some false alarms are inserted when the hand stays near or is projected onto the buttons. For gesture highlighting, since gesture recognition is carried out before a gesture is completed, a few more errors are introduced in order to improve the responsiveness.

Responsiveness

To press a button, the hand needs to stay on top of it for at least 0.5 second. The responsiveness is measured as the time it takes for the computer to accept the command after the hand presses the button.

To detect and recognize a gesture made by a presenter for content highlighting, we set $C_{\text{thresh}} = 0.5$. A gesture is verified only after the calculated confidence value by the algorithm in Section 3.3 is larger than this threshold.