```
1. paper_template_definition.h 新增加一个类 temp_dict_utils, 可以用来辅助制作模板字典, 也可以用来使用字典处理新图片。
enum temp_data_block_attr
                             // 未定义
     UNDEFINED.
                            // 试卷类型
// 准考证号
     EXAM_TYPE,
     STUDENT_NUM,
     SUBJECT_SCORE,
                            // 主观题分数
                            // 行单项选择题
// 行多项选择题
     ROW_SINGLE_CHOICE,
     ROW_MULTI_CHOICE,
                            // 列单项选择题
// 列多项选择题
     COL_SINGLE_CHOICE,
     COL_MULTI_CHOICE,
};
class IMPORT temp_dict_utils
public:
     temp_dict_utils(paper_template_definition *ptd);
     ~temp_dict_utils();
     void set_temp_data(temp_data *td, int width, int height, byte *data); // binary image data[width*height]
     bool add_markers(int num, int markers[][2]);
     bool add_block_undefined(int x0, int y0, int x1, int y1);
    bool add_block_digits(temp_data_block_attr attr, int x0, int y0, int x1, int y1, int y01, int col, int *thresh); bool add_block_choices(temp_data_block_attr attr, int x0, int y0, int x1, int y1, int z01, int row, int col,
int *thresh);
     bool proc_blocks(bool save, const char *file);
     temp_answer_sheet tas;
private:
     void read_choice(int wb, int hb, byte *db, temp_data_block *tdb);
     paper_template_definition *ptd;
     temp_data *td;
     int width, height;
     byte *data;
     void *hdc;
};
2. 制作模板的例子
模板图片: 60t.jpg
代码可以参考 make_dict_by_temp_dict_utils.cpp
void main()
{
     paper_template_definition ptd;
     make_temp_dict_1(&ptd);
     ptd.save_dict("data.dic");
}
bool make_temp_dict_1(paper_template_definition *ptd)
     Bitmap bmp;
     if(!bmp.ReadImage("60t.jpg", false)) return false;
int width = bmp.GetWidth(), height = bmp.GetHeight();
     byte *data = new byte[width*height];
    bmp.GetData(Bitmap::GRAY, data);
printf("image width = %d, height = %d\n", width, height);
     temp_data *td = ptd->push_back_temp_data();
     td->id = 1;
     td->subject = 1;
     temp_dict_utils tdu(ptd);
     tdu.set_temp_data(td, width, height, data);
     int markers[4][2] = \{0\};
    markers[0][0] = 88, markers[0][1] = 50; // 左上 markers[1][0] = 1547, markers[1][1] = 50; // 右上 markers[2][0] = 88, markers[2][1] = 2224; // 左下 markers[3][0] = 1547, markers[3][1] = 2224; // 右下
    ptd->image_gray_to_binary(width, height, data, data);
     tdu.add_markers(4, markers);
     //bmp.SetData(Bitmap::GRAY, data);
     //bmp.WriteImage("temp.png");
     bool ret = true;
     int alphas[4] = {600, 600, 600, 600}; // A~D
```

```
if(ret) ret = tdu.add_block_undefined(173, 184, 781, 296); // 姓名区域
if(ret) ret = tdu.add_block_choices(EXAM_TYPE, 909, 186, 957, 300, 226, 2, 1, alphas); // 试卷类型
if(ret) ret = tdu.add_block_digits(STUDENT_NUM, 1109, 165, 1352, 614, 232, 6, digits); // 学生考号
if(ret) ret = tdu.add_block_digits(SUBJECT_SCORE, 1383, 165, 1517, 614, 232, 3, digits); // 主观分
if(ret) ret = tdu.add_block_choices(ROW_SINGLE_CHOICE, 166, 699, 361, 890, 188, 5, 4, alphas); // 单选题
if(ret) ret = tdu.add_block_choices(ROW_MULTI_CHOICE, 477, 699, 679, 890, 510, 5, 4, alphas); // 多选题
delete[] data;
return ret;
}
```

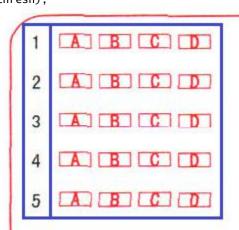
定义了三个接口 add_block_undefined, add_block_choices, add_block_digits。

(1) 对于数字类的选择涂块区域,可以使用

bool add_block_digits(temp_data_block_attr attr, int x0, int y0, int x1, int y1, int y01, int col, int *thresh); 最外围蓝色矩形区域, (x0,y0) 是左上角, (x1,y1) 是右下角,y01 是介于 y0 和 y1 之间的一条水平线,去掉了上部的非选择涂块区域。行数 row 确定为 10,列数 col 是待输入参数,此处 6 列。thresh 是阈值数组,0~1000 之间,若为 600,代表当填充率>=600/1000,则为选项被填充。

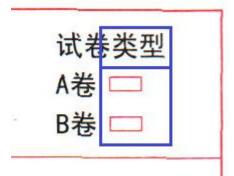


(2) 对按行排列的选择题,可以使用 bool add_block_choices(temp_data_block_attr attr, int x0, int y0, int x1, int y1, int z01, int row, int col, int *thresh);



attr 可取 ROW_SINGLE_CHOICE 或 ROW_MULTI_CHOICE。外围蓝色矩形区域,(x0,y0)是左上角,(x1,y1)是右下角,z01 是介于 x0 和 x1 之间的一条竖直线,去掉了左侧的题号区域。row=5 行,col=4 列。thresh[4]是阈值数组,代表填充率是千分之几。

(3) 对按列排列的选择题,可以使用 bool add_block_choices(temp_data_block_attr attr, int x0, int y0, int x1, int y1, int z01, int row, int col, int *thresh);



attr 可取 EXAM_TYPE(仅当试卷类型时)或 COL_SINGLE_CHOICE 或 COL_MULTI_CHOICE。用法类似上面两种,z01 是介于 y0 和 y1 之间的一条水平线,去掉了上部的题号区域或文字区域。

```
3. 使用模板字典测试新图片的例子
void main()
     paper_template_definition ptd;
     if(ptd.load_dict("data.dic"))
         printf("load_dict(\"%s\") success\n", "data.dic");
         test_temp_dict_1(&ptd);
    }
}
bool test_temp_dict_1(paper_template_definition *ptd)
{
     temp_data *td = ptd->get_temp_data(0);
     if(td==NULL) return false;
     const char* files[] =
     {
         "60t_1574389594.png"
    };
     int nfiles = sizeof(files)/sizeof(char*);
     for(int i=0; i<nfiles; i++)</pre>
     {
         const bool print = true;
const char *file = files[i];
         Bitmap bmp;
         if(bmp.ReadImage(files[i], false))
         {
              int width = bmp.GetWidth(), height = bmp.GetHeight();
              byte *data = new byte[width*height];
              bmp.GetData(Bitmap::GRAY, data);
              if(print) printf("%s\n width = %d, height = %d\n", file, width, height);
              if(ptd->auto_alignment_by_markers(td, width, height, data, print))
                   temp_dict_utils tdu(ptd);
                   tdu.set_temp_data(td, width, height, data);
                   tdu.set_temp_data(ta, m.se., series)
tdu.proc_blocks(true, file);
printf(" exam_type = %c\n", tdu.tas.exam_type);
printf(" student_num = %d\n", tdu.tas.student_num);
printf(" subject_score = %d\n", tdu.tas.student_score
                                                      , tdu.tas.subject_score);
                   }
              delete[] data;
         else if(print) printf(" error: can not read the image file %s\n", file);
    return false;
}
整个程序的运行输出如下,结构 tas 定义了输出结果。
load_dict("data.dic") success
60t_1574389594.png
 width = 1647, height = 2335
 alignment: a = -0.00, tx = 1, ty = -24
 proc block 0 ...
 proc block 1 ...
 proc block 2 ...
```

```
proc block 3 ...
 proc block 4 ...
 proc block 5 ...
 exam\_type = A
 student_num = 100
 subject_score = 95
 choice_num = 10
 1.
 2.
       В
 3.
       C
 4.
       D
 5.
       C
 6.
       Α
 7.
       AΒ
 8.
       В
 9.
       C
 10.
基于内容的模板定位
类 temp_dict_utils 新增 add_content 函数。处理函数 proc_blocks 是与原来版本共用的。
class EXPORT temp_dict_utils
public:
    bool add_content();
    bool proc_blocks(bool save, const char *file);
private:
以 40T 类型图片为例,制作模板的过程
bool make_temp_dict_1(paper_template_definition *ptd)
    Bitmap bmp;
    if(!bmp.ReadImage("40T/1.tif", false)) return false;
    int width = bmp.Getwidth(), height = bmp.GetHeight();
byte *data = new byte[width*height];
    bmp.GetData(Bitmap::GRAY, data);
printf("image width = %d, height = %d\n", width, height);
    temp_data *td = ptd->push_back_temp_data();
    td \rightarrow id = 1:
    td->subject = 1;
    temp_dict_utils tdu(ptd);
    tdu.set_temp_data(td, width, height, data);
    ptd->image_gray_to_binary(width, height, data, data);
    tdu.add_content();
    bmp.SetData(Bitmap::GRAY, data);
bmp.WriteImage("40T/temp.png"); // 添加块的坐标是基于 temp.png 的坐标
    bool ret = true;
int alphas[4] = {400, 400, 400, 400}; // A~D
    if(ret) ret = tdu.add_block_undefined(64, 120, 233, 221); // 姓名区域
    if(ret) ret = tdu.add_block_choices(EXAM_TYPE, 257, 236, 288, 392, 313, 2, 1, alphas); // 试卷类型
    if(ret) ret = tdu.add_block_digits(STUDENT_NUM, 288, 149, 654, 438, 199, 9, digits); // 学生考号
    if(ret) ret = tdu.add_block_choices(COL_SINGLE_CHOICE, 56, 368, 249, 487, 393, 4, 5, alphas); // 单选题 if(ret) ret = tdu.add_block_choices(COL_SINGLE_CHOICE, 56, 513, 648, 632, 539, 4, 15, alphas); // 单选题
    delete[] data;
    return ret;
}
识别测试的过程
bool test_temp_dict_1(paper_template_definition *ptd)
{
    temp_data *td = ptd->get_temp_data(0);
    if(td==NULL) return false;
     const char* files[] =
    {
         "40T/1.tif"
    };
    int nfiles = sizeof(files)/sizeof(char*);
```

```
for(int i=0; i<nfiles; i++)
{
    const bool print = true;
    const char *file = files[i];

Bitmap bmp;
    if(bmp.ReadImage(files[i], false))
{
        int width = bmp.GetWidth(), height = bmp.GetHeight();
        byte *data = new byte[width*height];
        bmp.GetData(Bitmap::GRAY, data);
        if(print) printf("%s\n width = %d, height = %d\n", file, width, height);

        if(ptd->auto_alignment_by_content(td, width, height, data, print))
        {
            temp_dict_utils tdu(ptd);
            tdu.set_temp_data(td, width, height, data);
            tdu.proc_blocks(true, file);
            printf(" exam_type = %c\n", tdu.tas.exam_type);
            printf(" student_num = %d\n", tdu.tas.student_num);
            printf(" student_num = %d\n", tdu.tas.student_num);
            printf(" choice_num = %d\n", tdu.tas.choice_num);
            for(int j=0; j<tdu.tas.choice_num; j++) printf(" %d.\t%s\n", 1+j, tdu.tas.choice[j]);
        }

        delete[] data;
        else if(print) printf(" error: can not read the image file %s\n", file);
        return false;
}
</pre>
```