

## 使用说明书

void ImageFusion(char* input1, char* input2, char* MaskImage, char* output, int dx[], int dy[], int a, double b1, int DX1, int DY1, double EPS)	图像融合。参考：a=3, b1=4, DX1=-68, DY1=-99, EPS=1, input1="图像融合 1. jpg", input2="图像融合 2. jpg", MaskImage="掩膜 .png", output="output. jpg"。 int dx[] = {0, 0, -1, 1}; int dy[] = {-1, 1, 0, 0};
void Screenshot1(HWND hWnd, LPCWSTR OutputImage)	截屏函数。hWnd 是要截屏的窗口句柄，如：GetDesktopWindow(); OutputImage 截图名称。
void Screenshot2(HWND hWnd, LPCWSTR OutputImage)	截屏函数。hWnd 是要截屏的窗口句柄，如：GetDesktopWindow(); OutputImage 截图名称。
void Screenshot3(HWND hWnd, LPCWSTR OutputImage)	截屏函数。hWnd 是要截屏的窗口句柄，如：GetDesktopWindow(); OutputImage 截图名称。
uint8_t* AESencrypt(uint8_t* input, uint8_t* key, int size)	AES 加密函数，input 是原数据，key 是密钥，size 是 input 的大小。返回加密结果数据。
uint8_t* AESdecrypt(uint8_t* input, uint8_t* key, int size)	AES 解密函数，input 是已加密数据，key 是密钥，size 是 input 的大小。返回解密结果数据。
void DES_Encrypt(char* PlainFile, char* Key, char* CipherFile)	DES 加密函数，支持多种文件。PlainFile 是原文件的文件名，Key 是密钥字符，CipherFile 是加密后的文件名。
void DES_Decrypt(char* CipherFile, char* Key, char* PlainFile)	DES 解密函数，支持多种文件。CipherFile 是已加密文件的文件名，Key 是密钥字符，PlainFile 是解密后的文件名。
int Equal(char* input1, char* input2, double c)	若比对图像的梯度幅相似性偏差值等于 c 则通过。input1 和 input2 是要比对的两个图像。c 是参考的阈值。支持 24 位 BMP 图像。
int GreaterThan(char* input1, char* input2, double c)	若比对图像的梯度幅相似性偏差值大于 c 则通过。input1 和 input2 是要比对的两个图像。c 是参考的阈值。支持 24 位 BMP 图像。
int LessThan(char* input1, char* input2, double c)	若比对图像的梯度幅相似性偏差值小于 c 则通过。input1 和 input2 是要比对的两个图像。c 是参考的阈值。支持 24 位 BMP 图像。
double GMSD(char* input1, char*	求两幅图像的梯度幅相似性偏差值并返

input2)	回结果。input1 和 input2 是要比对的两个图像。支持 24 位 BMP 图像。
void FileWrite(char* BMP, char* TXT)	图像隐写之文件写入，将文本文件写入图像。支持 32 位 BMP 图像。BMP 是要写入的图像文件名，TXT 是要写入图像的文本文件名。
void FileWriteOut(char* BMP, char* TXT)	图像隐写之文件写出，将文本文件从图像中取出来。支持 32 位 BMP 图像。BMP 是要写出的图像文件名，TXT 是写出图像后信息保存的文本文件名。
void Watershed2(char* input, char* inputMarqueurs, char* output, int r, unsigned char R, unsigned char G, unsigned char B)	图像分割之分水岭算法。inputMarqueurs 是输入图像的标记图像。R=230, G=0, B=0, r=1。支持 24 位 BMP 图像。
void EcrireImage1(char* input, char* output, uint32_t rayon)	图像分割。rayon=5。支持 24 位 BMP 图像。
void EcrireImage2(char* input, char* inputMarqueurs, char* output, uint32_t rayon)	图像分割。rayon=5。支持 24 位 BMP 图像。
void EcrireLPECouleur1(char* input, char* inputMarqueurs, char* output, uint32_t rayon)	图像分割。rayon=5。支持 24 位 BMP 图像。
void Watershed1(char* input, char* inputMarqueurs, char* output, uint32_t rayon)	图像分割之分水岭算法。inputMarqueurs 是输入图像的标记图像。rayon=5。支持 24 位 BMP 图像。
void EcrireImage3(char* input, char* inputMarqueurs, char* output, uint16_t rayon)	图像分割。rayon=1。支持 24 位 BMP 图像。
void EcrireImageCouleursAleatoires(char* input, char* inputMarqueurs, char* output, uint8_t r, uint8_t g, uint8_t b, uint16_t rayon)	图像分割。rayon=1。支持 24 位 BMP 图像。
void Watershed(char* input, char* inputMarqueurs, char* output, uint8_t r, uint8_t	图像分割之分水岭算法。inputMarqueurs 是输入图像的标记图像。a 一般为 255, rayon=1。支持 24 位 BMP 图像。

g, uint8_t b, uint8_t a, uint16_t rayon)	
double CharacterRecognition(char* TargetImage, char* TemplateFileGroup[])	<p>字符匹配，支持 BMP 图像，返回值是目标图像匹配到的模板文件的序号，如返回值是 2 则说明图像与序号为 2（序号从零开始）的模板匹配。</p> <p>参 考 :</p> <pre>TemplateFileGroup[]={ "0. txt", "1. txt", "2. txt", "3. txt", "4. txt", "5. txt", "6. txt", "7. txt", "8. txt", "9. txt" };</pre>
double CharacterRecognition1(char* TargetImage, char* TemplateFileGroup[])	<p>字符匹配，支持 BMP 图像，返回值是目标图像匹配到的模板文件的序号，如返回值是 2 则说明图像与序号为 2（序号从零开始）的模板匹配。</p> <p>参 考 :</p> <pre>TemplateFileGroup[]={ "0. txt", "1. txt", "2. txt", "3. txt", "4. txt", "5. txt", "6. txt", "7. txt", "8. txt", "9. txt" };</pre>
void CharacterSegmentation(char* input, string OutputFolder, int YHistogramValleyMaxPixelNumber, int XHistogramValleyMaxPixelNumber, double SubImgBlackPixelPercentage, int SingleNumberImgBoundary, int Infinite, double NumberImageBlackPixelPercentage)	<p>字符分割。支持 BMP 图像。</p> <p>OutputFolder 是结果输出的文件夹，如“output”，输出结果的文件名的构成方式为: 左上角的 X 坐标-左上角的 Y 坐标-右下角的 X 坐标-右下角的 Y 坐标，YHistogramValleyMaxPixelNumber 是求 Y 方向直方图，谷的最少黑色像素个数，YHistogramValleyMaxPixelNumber=0，XHistogramValleyMaxPixelNumber 是求 X 方向直方图，谷的最少黑色像素个数，XHistogramValleyMaxPixelNumber=4，SubImgBlackPixelPercentage 是一张子图内黑色像素超过一定百分比才算有数字，SubImgBlackPixelPercentage=0.001，SingleNumberImgBoundary 是单张数字图像边缘填充宽度，SingleNumberImgBoundary=5，Infinite 视作无穷大，Infinite=249480，NumberImageBlackPixelPercentage 是单张数字图像黑色像素个数超过所有数字图像，</p>

	NumberImageBlackPixelPercentage=0.35。
void CharacterSegmentation(char* input, char* output, int BoundaryRemoveGap, int BinaryGap, int YHistogramValleyMaxPixelNumber, double SubImgBlackPixelPercentage, int Infinite, int XHistogramValleyMaxPixelNumber, double NumberImageBlackPixelPercentage , int SingleNumberImgBoundary)	字符分割。支持 BMP 图像。 BinaryGap 是图像二值化全局阈值， BinaryGap=135, BoundaryRemoveGap 是 边缘全设为白色的距离， BoundaryRemoveGap=7, Infinite 是视 作无穷大，Infinite=249480， SingleNumberImgBoundary 是单张数字 图像边缘填充宽度， SingleNumberImgBoundary=5， YHistogramValleyMaxPixelNumber 是 求 Y 方向直方图，谷的最少黑色像素个 数， YHistogramValleyMaxPixelNumber=0， XHistogramValleyMaxPixelNumber 是 求 X 方向直方图，谷的最少黑色像素个 数， XHistogramValleyMaxPixelNumber=4， SubImgBlackPixelPercentage 是一 张子图内黑色像素超过一定百分比才算有 数字， SubImgBlackPixelPercentage=0.001， NumberImageBlackPixelPercentage 是 单张数字图像黑色像素个数超过所有数 字图像， NumberImageBlackPixelPercentage=0. 35。 参考：output="output"。
void CodeEncoding(std::string input, char* output, int width, int height, int margin, int eccLevel, int stride_bytes, int comp, int a)	二维码编码。input 是要编码的字符串， output 是生成的二维码图像文件名。 margin: 条形码周围的边距 ecc: 纠错级别，[0-8] a=1: AZTEC a=2: CODABAR a=3: CODE_39 a=4: CODE_93 a=5: CODE_128 a=6: DATA_MATRIX a=7: EAN_8 a=8: EAN_13 a=9: ITF a=10: MAXICODE a=11: PDF_417

	a=12: QR_CODE a=13: RSS_14 a=14: RSS_EXPANDED a=15: UPC_A a=16: UPC_E a=17: UPC_EAN_EXTENSION 参考: margin=10, eccLevel=-1, stride_bytes=0, comp=1。
std::string CodeDecoding(char* input, int req_comp, int a)	二维码解码。input 是输入的二维码图像文件名, 返回解码结果。 a=1: Lum a=2: RGB a=3: BGR a=4: RGBX a=5: XRGB a=6: BGRX a=7: XBGR 参考: req_comp=4, a=4。