Manual do utilizador

void ImageFusion(char* input1,char* input2,char* MaskImage,char* output,int dx[],int dy[],int a,double b1,int DX1,int DY1,double EPS)	Fusão de imagens. referência: a=3, b1=4， DX1=-68， DY1=-99， EPS=1，input1="Fusão de imagens 1.jpg"，input2="Fusão de imagens 2.jpg"，MaskImage="Máscara.png"，output="output.jpg"。 int dx[] = {0,0,-1,1}; int dy[] = {-1,1,0,0};
void DES_Encrypt(char *PlainFile, char *Key,char *CipherFile)	Função de encriptação DES, suportando vários ficheiros. PlainFile é o nome do arquivo original, Key é o caractere chave e CipherFile é o nome do arquivo criptografado.
void DES_Decrypt(char *CipherFile, char *Key,char *PlainFile)	DES função de descriptografia, suportando vários arquivos. CipherFile é o nome do arquivo criptografado, Key é o caractere de chave e PlainFile é o nome do arquivo descriptografado.
void FileWrite(char* BMP,char* TXT)	Escreva o arquivo de esteganografia da imagem e escreva o arquivo de texto na imagem. Suporta imagens BMP de 32 bits. BMP é o nome do arquivo da imagem a ser escrita, e TXT é o nome do arquivo de texto da imagem a ser escrita.
void FileWriteOut(char* BMP,char* TXT)	Escreva o arquivo de esteganografia da imagem e extraia o arquivo de texto da imagem. Suporta imagens BMP de 32 bits. BMP é o nome do arquivo de imagem a ser escrito, e TXT é o nome do arquivo de texto onde as informações são salvas após a gravação da imagem.
void Watershed2(char* input,char* inputMarqueurs,char* output,int r,unsigned char R,unsigned char G,unsigned char B)	O algoritmo divisor de águas para segmentação de imagens. inputMarqueurs é a imagem anotada da imagem de entrada. R=230, G=0, B=0, r=1. Suporta imagens BMP de 24 bits.
void EcrireImage1(char* input,char* output,uint32_t	Segmentação de imagens. rayon=5. Suporta imagens PNG.

rayon)	
void EcrireImage2(char* input,char* inputMarqueurs,char* output,uint32_t rayon)	Segmentação de imagens. rayon=5. Suporta imagens PNG.
void EcrireLPECouleur1(char* input,char* inputMarqueurs,char* output,uint32_t rayon)	Segmentação de imagens. rayon=5. Suporta imagens PNG.
void Watershed1(char* input,char* inputMarqueurs,char* output,uint32_t rayon)	O algoritmo divisor de águas para segmentação de imagens. inputMarqueurs é a imagem anotada da imagem de entrada. rayon=5. Suporta imagens PNG.
void EcrireImage3(char* input,char* inputMarqueurs,char* output,uint16_t rayon)	Segmentação de imagens. rayon=1. Suporta imagens PNG.
void EcrireImageCouleursAleatoires(char* input,char* inputMarqueurs,char* output,uint8_t r,uint8_t g,uint8_t b,uint16_t rayon)	Segmentação de imagens. rayon=1. Suporta imagens PNG.
void Watershed(char* input,char* inputMarqueurs,char* output,uint8_t r,uint8_t g,uint8_t b,uint8_t a,uint16_t rayon)	O algoritmo divisor de águas para segmentação de imagens. InputMarqueurs é a imagem anotada da imagem de entrada. a é geralmente 255, e rayon=1. Suporta imagens PNG.
double CharacterRecognition(char* TargetImage,char* TemplateFileGroup[])	Correspondência de caracteres, suporta imagens BMP, e o valor de retorno é o número de sequência do arquivo de modelo correspondente à imagem de destino. Se o valor de retorno é 2, ele indica que a imagem corresponde ao modelo com o número de sequência 2 (a partir de zero). referência : TemplateFileGroup[]={ "0.txt", "1.txt", "2.txt", "3.txt", "4.txt", "5.txt", "6.txt", "7.txt", "8.txt", "9.txt" };

double CharacterRecognition1(char* TargetImage,char* TemplateFileGroup[])	Correspondência de caracteres, suporta imagens BMP, e o valor de retorno é o número de sequência do arquivo de modelo correspondente à imagem de destino. Se o valor de retorno é 2, ele indica que a imagem corresponde ao modelo com o número de sequência 2 (a partir de zero). referência : TemplateFileGroup[]={ "0.txt", "1.txt", "2.txt", "3.txt", "4.txt", "5.txt", "6.txt", "7.txt", "8.txt", "9.txt" };
void CharacterSegmentation(char* input, string OutputFolder, int YHistogramValleyMaxPixelNumber, int XHistogramValleyMaxPixelNumber, double SubImgBlackPixelPercentage, int SingleNumberImgBoundary, int Infinite, double NumberImageBlackPixelPercentage)	Segmentação de caracteres. Suporta imagens BMP. OutputFolder é a pasta para a saída de resultados, como "output". O formato do nome do arquivo para a saída de resultados é: X coordenada no canto superior esquerdo - Y coordenada no canto superior esquerdo - X coordenada no canto inferior direito - Y coordenada no canto inferior direito , YHistogramValleyMaxPixelNumber é o número mínimo de pixels pretos no vale do histograma de direção Y, YHistogramValleyMaxPixelNumber=0, XHistogramValleyMaxPixelNumber é o número mínimo de pixels pretos no vale do histograma de direção X, XHistogramValleyMaxPixelNumber=4, SubImgBlackPixelPercentage é a porcentagem de pixels pretos em um subgrafo que é considerado um número , SubImgBlackPixelPercentage=0.001, SingleNumberImgBoundary é a largura de preenchimento de borda de uma única imagem digital , SingleNumberImgBoundary=5,Infinito é considerado infinito , Infinite=249480 ,

	NumberImageBlackPixelPercentage é o número de pixels pretos em uma única imagem digital que excede todas as imagens digitais , 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)	Segmentação de caracteres. Suporta imagens BMP. BinaryGap é o limite global para binarização de imagens , BinaryGap=135, BoundaryRemoveGap é a distância em que todas as bordas estão definidas para branco , BoundaryRemoveGap=7 , Infinito é considerado infinito , Infinite=249480 , SingleNumberImgBoundary é a largura de preenchimento de borda de uma única imagem digital , SingleNumberImgBoundary=5 , YHistogramValleyMaxPixelNumber é o número mínimo de pixels pretos no vale do histograma de direção Y, YHistogramValleyMaxPixelNumber=0, XHistogramValleyMaxPixelNumber é o número mínimo de pixels pretos no vale do histograma de direção X, XHistogramValleyMaxPixelNumber=4, SubImgBlackPixelPercentage é a porcentagem de pixels pretos em um subgrafo que é considerado um número , SubImgBlackPixelPercentage=0.001, NumberImageBlackPixelPercentage é o número de pixels pretos em uma única imagem digital que excede todas as imagens digitais , NumberImageBlackPixelPercentage=0.35。 Referência: output="output"。
void CodeEncoding(std::string input,char* output, int width,int height, int margin, int eccLevel, int stride_bytes,	Codificação de código QR. Entrada é a string a ser codificada, e saída é o nome do arquivo da imagem gerada do código QR.

int comp,int a)	margem: A margem em torno do código de barras ecc: Nível de correcção de erros, [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 Referência：margin=10，eccLevel=-1，stride_bytes=0，comp=1。
std::string CodeDecoding(char* input,int req_comp,int a)	Descodificação de código QR. Entrada é o nome do arquivo da imagem de código QR de entrada e retorna o resultado de decodificação. a=1：Lum a=2：RGB a=3：BGR a=4：RGBX a=5：XRGB a=6：BGRX a=7：XBGR Referência：req_comp=4，a=4。