# 算法接口说明文档

目前有背景替换和人脸关键点（98）检测模型

## 一函数说明

### 背景替换

void SetThreadNum(int num);设置omp 线程数。 在定于BgReplace 对象前调用。

ImagSegment类的构造函数ImagSegment (AiLabel ai\_tpye);

AiLabel: 是一个枚举类型 包含BSHUFFLENET。BSHUFFLENE表示算法模型。

cv::Mat &ImagSegment:: cv::Mat BackgroundRelpace(const cv::Mat &inImg, const cv::Mat &backImg,const cv::Mat &mask);

返回替换背景后的图

InImg: 需要做背景替换的图像。

BackImg: 背景图像。

mask:mask图。

cv::Mat ImagSegment::GetScoreMask(const cv::Mat &inImg);返回mask。inImg： 输入图像。

### 人脸关键点检测

FaceInformation 人脸关键检测类，成员函数void GetFaceDetails(const cv::Mat &Imgstd::vector<FaceDetails> & vFaceInor)，输入图像返回人脸信息。

Void OptFaceDetails(constcv::Mat&Img,std::vector<

FaceDetails> & vFaceInor), 返回优化后的人脸信息。

FaceDetails 是一个结构体，存储人脸信息，其中包含是个数据成员，FaceLands(人脸关键点)，FaceBox(人脸框)， conf(人脸的自信度)，angle(人脸角度，占时不能用)。具体看eeodnnl.h 文件。

## 二、实列

## Windos 例子

### 1.背景替换

#include "eeodnnl.h"

std::string back\_dir = "back\_img.jpg"; 背景图片

cv::Mat backImg = cv::imread(back\_dir);

std::string input\_dir = "inputImag.jpg"; 需替换的图片

cv::Mat inputImg = cv::imread(input\_dir);

BgReplace bgmode(SHUFFLENET);

cv::Mat mask = bgmode.GetScoreMask(inputImg) //返回 放大后的mask

cv::Mat UnitImg = bgmode.BackgroundRelpace(inputImg, backImg,mask); //返回合并后的图像

### 2.人脸关键点检测：

FaceInformation face\_infor = FaceInformation();

cv::VideoCapture capture(0);

cv::Mat img;

while (true)

{

Sleep(30);

capture >> img;

std::vector<FaceDetails> face\_details;

face\_infor.GetFaceDetails(img, face\_details);

//draw

for(int i=0; i<face\_details.size(); i++)

{

std::vector<int> box = face\_details.at(i).FaceBox;

std::vector<int> facland = face\_details.at(i).FaceLands;

int x1 = box.at(0);

int y1 = box.at(1);

int x2 = box.at(2);

int y2 = box.at(3);

//画人脸框

cv::rectangle(img, cv::Rect(x1, y1, x2 - x1, y2 - y1), cv::Scalar(0, 255, 0), 1, 1, 0);

for (int i = 0; i < facland.size(); i = i + 2)

{

int x = facland.at(i);

int y = facland.at(i + 1);

//画人脸点

cv::circle(img, cv::Point(x, y), 2, cv::Scalar(255, 0, 255),2);

}

}

cv::imshow("face\_landmark\_68", img);

cv::waitKey(2);

}

## Mac 下例子

### 1.背景替换

void LoadModelAndShowVido()

{

SetThreadNum(1);

std::string model\_path="/Users/liuguopeng/CLionProjects/Testdll/model";

std::string back\_dir = "/Users/liuguopeng/CLionProjects/Testdll/back.jpeg"; //back\_img.jpg

cv::Mat backImg = cv::imread(back\_dir);

ImagSegment bgmode(SHUFFLENET,192, model\_path); //BISENET SHUFFLENET

cv::VideoCapture capture(0);

int iH = 720;

int iW = 1280;

capture.set(3, iW);

capture.set(4, iH);

backImg = ReSizeImg(backImg, iH, iW);

clock\_t start\_clock, end\_clock;

while (true)

{

cv::Mat img;

capture >> img;

start\_clock = clock();

cv::Mat maks = bgmode.GetScoreMask(img);

end\_clock = clock();

std::cout << "interval time: " << end\_clock - start\_clock << std::endl;

start\_clock = clock();

bgmode.BackgroundRelpace(img, backImg, maks);

end\_clock = clock();

std::cout << "backReplace time: " << end\_clock - start\_clock << std::endl;

cv::imshow("test\_vido", img);

cv::waitKey(2);

}

}

### 2.人脸关键点检测：

void test\_face\_points()

{

std::string model\_path="/Users/liuguopeng/CLionProjects/Testdll/model";

int input\_size = 192;

SetThreadNum(1);

FaceInformation face\_infor = FaceInformation(0.7,0.5,0.5,8,5, input\_size,model\_path);

cv::VideoCapture capture(0);

capture.set(3, 1280);

capture.set(4, 720);

cv::Mat img;

while (true)

{

capture >> img;

std::vector<FaceDetails> face\_details;

clock\_t startClock, endClock;

startClock = clock();

std::vector<float> input=ImageTrance(img, input\_size);

face\_infor.GetOptFaceDetails(img, input, face\_details);

endClock = clock();

int interval = endClock - startClock;

std::cout << "FaceDetect times: " << interval << std::endl;

//draw

for(int i=0; i<face\_details.size(); i++)

{

std::vector<int> box = face\_details.at(i).FaceBox;

std::vector<int> facland = face\_details.at(i).FaceLands;

int x1 = box.at(0);

int y1 = box.at(1);

int x2 = box.at(2);

int y2 = box.at(3);

cv::rectangle(img, cv::Rect(x1, y1, x2 - x1, y2 - y1), cv::Scalar(0, 255, 0), 1, 1, 0);

for (int i = 0; i < facland.size(); i = i + 2)

{

int x = facland.at(i);

int y = facland.at(i + 1);

cv::circle(img, cv::Point(x, y), 2, cv::Scalar(255, 0, 255), 2);

}

}

cv::imshow("face\_landmark\_98", img);

cv::waitKey(2);

}

}

int main()

{

//LoadModelAndShowVido();

test\_face\_points();

return 0;

}