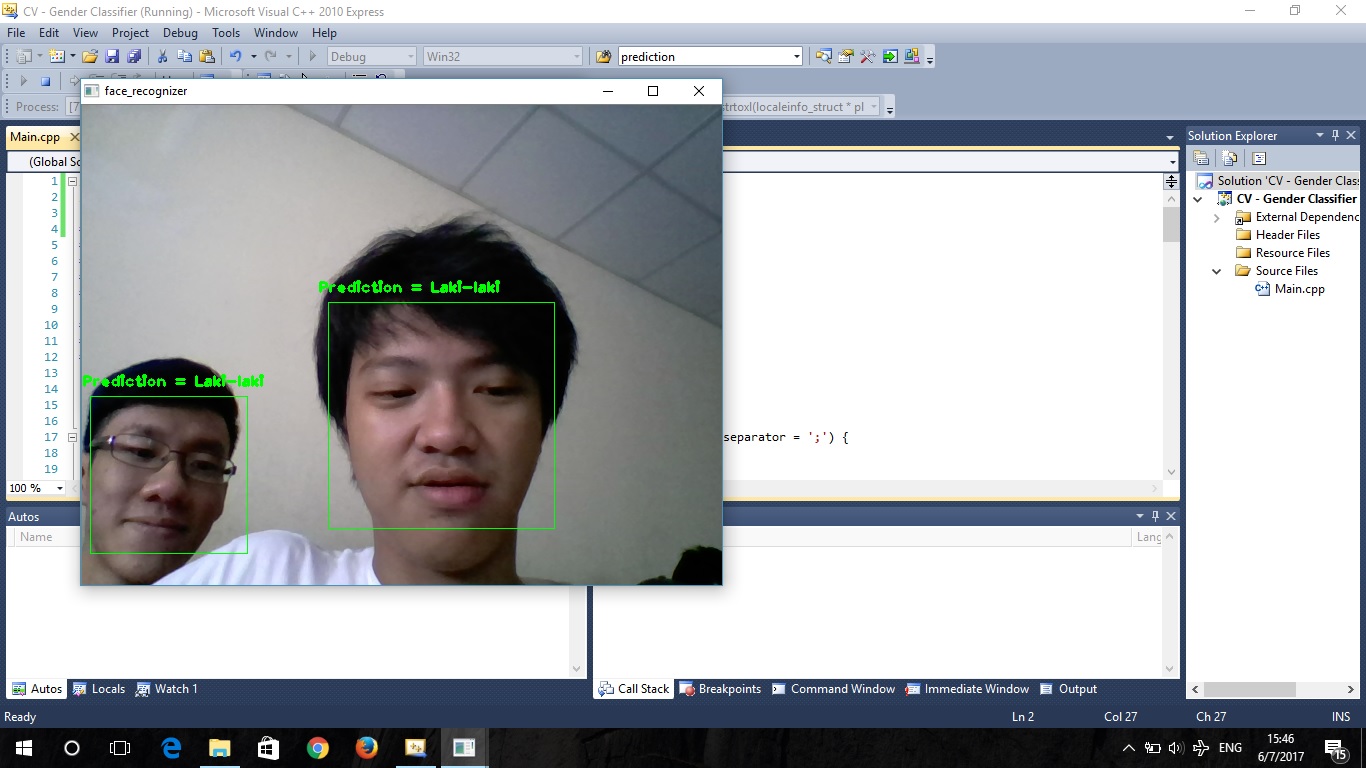
Dokumentasi Project Computer Vision – Gender Classification

Project ini merupakan program untuk mendeteksi jenis kelamin berdasarkan wajah manusia. Data training berasal dari beberapa foto pria dan wanita yang kami dapatkan dari internet.

Hasil output Program:



Source Code:

//Giovanno Sebastian 1901456905

//Robby Pranata 1901460474

#include "opencv2/core/core.hpp"

#include "opencv2/contrib/contrib.hpp"

#include "opencv2/highgui/highgui.hpp"

#include "opencv2/imgproc/imgproc.hpp"

#include "opencv2/objdetect/objdetect.hpp"

#include <iostream>

#include <fstream>

#include <sstream>

using namespace cv;

using namespace std;

static void read\_csv(const string& filename, vector<Mat>& images, vector<int>& labels, char separator = ';') {

std::ifstream file(filename.c\_str(), ifstream::in);

if (!file) {

string error\_message = "No valid input file was given, please check the given filename.";

CV\_Error(CV\_StsBadArg, error\_message);

}

string line, path, classlabel;

while (getline(file, line)) {

stringstream liness(line);

getline(liness, path, separator);

getline(liness, classlabel);

if(!path.empty() && !classlabel.empty()) {

images.push\_back(imread(path, 0));

labels.push\_back(atoi(classlabel.c\_str()));

}

}

}

int main() {

// Get the path to your CSV:

string fn\_haar = "haarcascade\_frontalface\_alt.xml";

string fn\_csv = "photo.csv";

int deviceId = 0;

vector<Mat> images;

vector<int> labels;

try {

read\_csv(fn\_csv, images, labels);

} catch (cv::Exception& e) {

cerr << "Error opening file \"" << fn\_csv << "\". Reason: " << e.msg << endl;

exit(1);

}

int im\_width = images[0].cols;

int im\_height = images[0].rows;

// Create a FaceRecognizer and train it on the given images:

Ptr<FaceRecognizer> model = createFisherFaceRecognizer();

model->train(images, labels);

CascadeClassifier haar\_cascade;

haar\_cascade.load(fn\_haar);

// Get a handle to the Video device:

VideoCapture cap(deviceId);

if(!cap.isOpened()) {

cerr << "Capture Device ID " << deviceId << "cannot be opened." << endl;

return -1;

}

// Holds the current frame from the Video device:

Mat frame;

for(;;) {

cap >> frame;

// Clone the current frame:

Mat original = frame.clone();

// Convert the current frame to grayscale:

Mat gray;

cvtColor(original, gray, CV\_BGR2GRAY);

// Find the faces in the frame:

vector< Rect\_<int> > faces;

haar\_cascade.detectMultiScale(gray, faces);

for(int i = 0; i < faces.size(); i++) {

// Process face by face:

Rect face\_i = faces[i];

// Crop the face from the image. So simple with OpenCV C++:

Mat face = gray(face\_i);

// Resizing the face is necessary for Eigenfaces and Fisherfaces.

Mat face\_resized;

cv::resize(face, face\_resized, Size(im\_width, im\_height), 1.0, 1.0, INTER\_CUBIC);

int prediction = model->predict(face\_resized);

rectangle(original, face\_i, CV\_RGB(0, 255,0), 1);

// Create the text we will annotate the box with:

char gender[20];

//validate that 0 is female and 1 is male

if(prediction==0) strcpy(gender, "Perempuan");

else if(prediction==1) strcpy(gender, "Laki-laki");

string box\_text = format("Prediction = %s", gender);

int pos\_x = std::max(face\_i.tl().x - 10, 0);

int pos\_y = std::max(face\_i.tl().y - 10, 0);

putText(original, box\_text, Point(pos\_x, pos\_y), FONT\_HERSHEY\_PLAIN, 1.0, CV\_RGB(0,255,0), 2.0);

}

imshow("face\_recognizer", original);

char key = (char) waitKey(20);

//esc

if(key == 27)

break;

}

return 0;

}