using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.IO;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Diagnostics;

using System.Data.OleDb;

using Emgu.CV;

using Emgu.CV.Structure;

using Emgu.CV.CvEnum;

using Accord.Vision.Detection;

using Accord.Vision.Detection.Cascades;

using Accord.Imaging.Filters;

using AForge.Vision;

namespace SampleFaceApplication

{

public partial class FrmEnrollUser : Form

{

private string picture;

private HaarObjectDetector detector;

private Accord.Vision.Detection.HaarCascade cascade;

private OleDbConnection cn = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" + Application.StartupPath + "\\dbFace.accdb;Persist Security Info=True");

//private OleDbConnection cn = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=C:\Project 2016\SampleFaceApplication\SampleFaceApplication\dbFace.accdb;Persist Security Info=True");

private OleDbCommand cmd = new OleDbCommand();

private OleDbDataReader dr;

public FrmEnrollUser()

{

try

{

InitializeComponent();

cascade = new FaceHaarCascade();

}

catch (Exception er)

{

MessageBox.Show(er.Message, "Face Detection and Recognition Failure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void btnUpload\_Click(object sender, EventArgs e)

{

try

{

//MessageBox.Show("Always ensure that load ");

OpenFileDialog openFileDialog = new OpenFileDialog();

//openFileDialog.InitialDirectory = Environment.GetFolderPath(Environment.SpecialFolder.Personal);

//openFileDialog.InitialDirectory = Application.StartupPath + "\\TrainDatabase";

openFileDialog.Filter = "Image Files (\*.jpg;\*.jpeg;\*.gif;\*.bmp)|\*.jpg;\*.jpeg;\*.gif;\*.bmp|All Files (\*.\*)|\*.\*";

if (openFileDialog.ShowDialog(this) == DialogResult.OK)

{

pictureBox1.Image = Image.FromFile(openFileDialog.FileName);

//Bitmap uploadImage = new Bitmap(openFileDialog.FileName);

//pictureBox1.Image = resizeImage(uploadImage, 300, 300);

//pictureBox1.Size = pictureBox1.Image.Size;

picture = openFileDialog.FileName;

//MessageBox.Show(picture);

pictureBox2.Image = null;

}

btnUpload.Enabled = false;

btnTrainFace.Enabled = true;

btnSave.Enabled = false;

}

catch (Exception er)

{

MessageBox.Show(er.Message, "Face Detection and Recognition Failure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private static Image resizeImage(Image image, int imgWidth, int imgHeight)

{

Bitmap newImage = new Bitmap(imgWidth, imgHeight);

Graphics g = Graphics.FromImage((Image)newImage);

g.InterpolationMode = System.Drawing.Drawing2D.InterpolationMode.High;

g.DrawImage(image, 0, 0, imgWidth, imgHeight);

return newImage;

}

private void btnTrainFace\_Click(object sender, EventArgs e)

{

try

{

detector = new HaarObjectDetector(cascade, 30);

detector.SearchMode = ObjectDetectorSearchMode.Single;//.NoOverlap;//.Default; //.Average;

detector.ScalingFactor = 1.5f;

detector.ScalingMode = ObjectDetectorScalingMode.GreaterToSmaller;

detector.UseParallelProcessing = true;

detector.Suppression = 3;

Bitmap picture = new Bitmap(pictureBox1.Image);

Image<Bgr, byte> Frame = new Image<Bgr, byte>(picture);

Stopwatch sw = Stopwatch.StartNew();

Rectangle[] faceObjects = detector.ProcessFrame(picture);

sw.Stop();

if (faceObjects.Length > 0)

{

RectanglesMarker marker = new RectanglesMarker(faceObjects, Color.Fuchsia);

pictureBox1.Image = marker.Apply(picture);

//Graphics g = Graphics.FromImage(pictureBox1.Image);

foreach (var face in faceObjects)

{

//g.DrawRectangle(Pens.DeepSkyBlue, face);

Frame.Draw(face, new Bgr(Color.Red), 3);

Bitmap c = Frame.ToBitmap();

Bitmap bmp = new Bitmap(face.Width, face.Height);

Graphics gg = Graphics.FromImage(bmp);

gg.DrawImage(c, 0, 0, face, GraphicsUnit.Pixel);

pictureBox2.Image = bmp;

//bmp.Save("myface(accord) " + DateTime.Now.Second.ToString());

gg.Dispose();

}

//g.Dispose();

//label1.Text = "Completed operation!! " + faceObjects.Length.ToString() + " Face detected";

MessageBox.Show("Train Face operation successful!!! " + faceObjects.Length.ToString() + " Face detected", "Train face", MessageBoxButtons.OK, MessageBoxIcon.Information);

txtFaceId.Text = genFaceId();

txtUsername.Text = "User" + txtFaceId.Text;

btnUpload.Enabled = false;

btnTrainFace.Enabled = false;

btnSave.Enabled = true;

}

else

{

MessageBox.Show("Image cannot be trained!!! No face detected in the current image", "Fail to Train face", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

catch (Exception er)

{

MessageBox.Show(er.Message, "Face Detection and Recognition Failure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private string genFaceId()

{

string FaceId = "";

try

{

int rowCount = 0;

cn.Open();

cmd.CommandText = "select count(\*) as RowCount from tblUser";

dr = cmd.ExecuteReader();

while (dr.Read())

rowCount = (int)(dr["RowCount"]);

rowCount++;

FaceId = FaceId + rowCount;

dr.Close();

cn.Close();

}

catch (Exception Er)

{

dr.Close();

cn.Close();

MessageBox.Show(Er.Message, "Face Detection and Recognition App", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

return FaceId;

}

private void btnCancel\_Click(object sender, EventArgs e)

{

this.Hide();

}

private void btnReset\_Click(object sender, EventArgs e)

{

pictureBox1.Image = null;

pictureBox2.Image = null;

txtFaceId.ResetText();

txtUsername.ResetText();

cboSex.ResetText();

btnUpload.Enabled = true;

btnTrainFace.Enabled = false;

btnSave.Enabled = false;

}

private void FrmEnrollUser\_Load(object sender, EventArgs e)

{

cmd.Connection = cn;

txtFaceId.Enabled = false;

btnReset\_Click(sender, e);

}

private void btnSave\_Click(object sender, EventArgs e)

{

try

{

if (isAllEntryFilled())

{

int FaceId = Convert.ToInt32(txtFaceId.Text);

string Username = txtUsername.Text;

string Sex = cboSex.Text;

if(picture != "") pictureBox1.Image = Image.FromFile(picture);

if (isSavePicture())

{

cn.Open();

cmd.CommandText = "insert into tblUser values(" + FaceId + ", '" + Username + "','" + Sex + "')";

cmd.ExecuteNonQuery();

MessageBox.Show("New Face Details successfully saved!!!", "Face Detection and Recognition App.", MessageBoxButtons.OK, MessageBoxIcon.Information);

cn.Close();

btnReset\_Click(sender, e);

}

}

}

catch (Exception er)

{

cn.Close();

MessageBox.Show(er.Message, "Face Detection and Recognition App.", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private bool isAllEntryFilled()

{

bool isEntryFilled = false;

if ((txtFaceId.Text == ""))

{

MessageBox.Show("Please click Reset or Load Image button to generate new User Image(Face)", "Face Detection and Recognition App.", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

btnUpload.Focus();

return isEntryFilled;

}

else if ((txtUsername.Text == ""))

{

MessageBox.Show("Please enter the username", "Face Detection and Recognition App.", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

txtUsername.Focus();

return isEntryFilled;

}

else if ((cboSex.Text == ""))

{

MessageBox.Show("Please set the sex", "Face Detection and Recognition App.", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

cboSex.Focus();

return isEntryFilled;

}

else

return true;

}

private bool isSavePicture()

{

bool isSaved = false;

try

{

if ((picture != "") && (pictureBox1.Image != null) && (pictureBox2.Image != null))

{

if (txtFaceId.Text != "")

{

//Save the test face and trained face

string testPictureFileName = txtUsername.Text + txtFaceId.Text;

if (File.Exists(Application.StartupPath + "\\Face Db\\Test Face\\" + testPictureFileName + ".jpg"))

{

MessageBox.Show("Picture Cannot be Saved!!!, Duplicate Found", "Filename Already Exists");

}

else if (File.Exists(Application.StartupPath + "\\Face Db\\Trained Face\\" + txtFaceId.Text + ".jpg"))

{

MessageBox.Show("Picture Cannot be Saved!!!, Duplicate Found", "Filename Already Exists");

}

else

{

pictureBox1.Image.Save(Application.StartupPath + "\\Face Db\\Test Face\\" + testPictureFileName + ".jpg");

pictureBox2.Image.Save(Application.StartupPath + "\\Face Db\\Trained Face\\" + txtFaceId.Text + ".jpg");

MessageBox.Show("Trained Face Saved!!!");

//picture = Application.StartupPath + "\\Pictures\\" + pictureFileName + ".jpg";

isSaved = true;

}

}

}

}

catch (Exception er)

{

MessageBox.Show(er.Message, "Face Detection and Recognition Failure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

return isSaved;

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace SampleFaceApplication

{

public partial class MDIFace : Form

{

FrmSplash frmSplash;

public MDIFace(FrmSplash frmSpl)

{

InitializeComponent();

frmSplash = frmSpl;

}

public MDIFace()

{

InitializeComponent();

}

private void ShowNewForm(object sender, EventArgs e)

{

FrmEnrollUser childForm = new FrmEnrollUser();

childForm.MdiParent = this;

childForm.Show();

}

private void OpenFile(object sender, EventArgs e)

{

/\* OpenFileDialog openFileDialog = new OpenFileDialog();

openFileDialog.InitialDirectory = Environment.GetFolderPath(Environment.SpecialFolder.Personal);

openFileDialog.Filter = "Text Files (\*.txt)|\*.txt|All Files (\*.\*)|\*.\*";

if (openFileDialog.ShowDialog(this) == DialogResult.OK)

{

string FileName = openFileDialog.FileName;

}\*/

}

private void SaveAsToolStripMenuItem\_Click(object sender, EventArgs e)

{

/\*SaveFileDialog saveFileDialog = new SaveFileDialog();

saveFileDialog.InitialDirectory = Environment.GetFolderPath(Environment.SpecialFolder.Personal);

saveFileDialog.Filter = "Text Files (\*.txt)|\*.txt|All Files (\*.\*)|\*.\*";

if (saveFileDialog.ShowDialog(this) == DialogResult.OK)

{

string FileName = saveFileDialog.FileName;

}\*/

}

private void ExitToolsStripMenuItem\_Click(object sender, EventArgs e)

{

this.Close();

Application.Exit();

}

private void CutToolStripMenuItem\_Click(object sender, EventArgs e)

{

}

private void CopyToolStripMenuItem\_Click(object sender, EventArgs e)

{

//face recognition

FrmFaceRecognition childForm = new FrmFaceRecognition();

childForm.MdiParent = this;

childForm.Show();

}

private void PasteToolStripMenuItem\_Click(object sender, EventArgs e)

{

}

private void ToolBarToolStripMenuItem\_Click(object sender, EventArgs e)

{

toolStrip.Visible = toolBarToolStripMenuItem.Checked;

}

private void StatusBarToolStripMenuItem\_Click(object sender, EventArgs e)

{

statusStrip.Visible = statusBarToolStripMenuItem.Checked;

}

private void CascadeToolStripMenuItem\_Click(object sender, EventArgs e)

{

LayoutMdi(MdiLayout.Cascade);

}

private void TileVerticalToolStripMenuItem\_Click(object sender, EventArgs e)

{

LayoutMdi(MdiLayout.TileVertical);

}

private void TileHorizontalToolStripMenuItem\_Click(object sender, EventArgs e)

{

LayoutMdi(MdiLayout.TileHorizontal);

}

private void ArrangeIconsToolStripMenuItem\_Click(object sender, EventArgs e)

{

LayoutMdi(MdiLayout.ArrangeIcons);

}

private void CloseAllToolStripMenuItem\_Click(object sender, EventArgs e)

{

foreach (Form childForm in MdiChildren)

{

childForm.Close();

}

}

private void saveToolStripMenuItem\_Click(object sender, EventArgs e)

{

FrmFaceDetection childForm = new FrmFaceDetection();

childForm.MdiParent = this;

childForm.Show();

}

private void MDIFace\_Load(object sender, EventArgs e)

{

frmSplash.Hide();

}

private void indexToolStripMenuItem\_Click(object sender, EventArgs e)

{

AboutFaceDetection childForm = new AboutFaceDetection();

childForm.MdiParent = this;

childForm.Show();

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Diagnostics;

using Emgu.CV;

using Emgu.CV.Structure;

using Emgu.CV.CvEnum;

using Accord.Vision.Detection;

using Accord.Vision.Detection.Cascades;

using Accord.Imaging.Filters;

using AForge.Vision;

namespace SampleFaceApplication

{

public partial class FrmFaceDetection : Form

{

private Capture webcamVideo;

private Emgu.CV.HaarCascade faceDetection;

private HaarObjectDetector detector;

private Accord.Vision.Detection.HaarCascade cascade;

public FrmFaceDetection()

{

try

{

InitializeComponent();

webcamVideo = new Capture();

//faceDetection = new HaarCascade(Application.StartupPath + "\\haarcascade\_frontalface\_default.xml");

faceDetection = new Emgu.CV.HaarCascade("haarcascade\_frontalface\_default.xml");

cascade = new FaceHaarCascade();

}

catch (Exception er)

{

MessageBox.Show(er.Message, "Face Detection and Recognition Failure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void FrmFaceDetection\_Load(object sender, EventArgs e)

{

button3.Text = "&Stop";

}

private void timer1\_Tick(object sender, EventArgs e)

{

try

{

using (Image<Bgr, byte> Frame = webcamVideo.QueryFrame().Resize(397, 292, Emgu.CV.CvEnum.INTER.CV\_INTER\_CUBIC))

{

if (Frame != null)

{

Image<Gray, byte> grayFrame = Frame.Convert<Gray, byte>();

//var faces = grayFrame.DetectHaarCascade(faceDetection, 1.1, 10, Emgu.CV.CvEnum.HAAR\_DETECTION\_TYPE.DO\_CANNY\_PRUNING, new Size(20, 20)new Size(Frame.Width / 8, Frame.Height / 8))[0];

var faces = grayFrame.DetectHaarCascade(faceDetection, 1.1, 10, Emgu.CV.CvEnum.HAAR\_DETECTION\_TYPE.DO\_CANNY\_PRUNING, new Size(20, 20))[0];

foreach (var face in faces)

{

Frame.Draw(face.rect, new Bgr(Color.Red), 3);

Bitmap c = Frame.ToBitmap();

Bitmap bmp = new Bitmap(face.rect.Width, face.rect.Height);

Graphics g = Graphics.FromImage(bmp);

g.DrawImage(c, 0, 0, face.rect, GraphicsUnit.Pixel);

pictureBox2.Image = bmp;

bmp.Save("myface" + DateTime.Now.Second.ToString());

label1.Text = "Operation Completed!!! " + faces.Length.ToString() + " Face detected";

timer1.Enabled = false;

button3.Text = "&Play Again";

}

pictureBox1.Image = Frame.ToBitmap();

}

}

}

catch (Exception er)

{

MessageBox.Show(er.Message, "Face Detection and Recognition Failure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void button1\_Click(object sender, EventArgs e)

{

try

{

detector = new HaarObjectDetector(cascade, 30);

detector.SearchMode = ObjectDetectorSearchMode.Single;//.Default;//.NoOverlap;//

detector.ScalingFactor = 2.5f;

detector.ScalingMode = ObjectDetectorScalingMode.SmallerToGreater;//.GreaterToSmaller;

detector.UseParallelProcessing = true;

detector.Suppression = 3;

Bitmap picture = new Bitmap(pictureBox2.Image);

Image<Bgr, byte> Frame = new Image<Bgr, byte>(picture);

Stopwatch sw = Stopwatch.StartNew();

Rectangle[] faceObjects = detector.ProcessFrame(picture);

sw.Stop();

if (faceObjects.Length > 0)

{

RectanglesMarker marker = new RectanglesMarker(faceObjects, Color.Fuchsia);

pictureBox2.Image = marker.Apply(picture);

}

label1.Text = "Operation Completed!!! " + faceObjects.Length.ToString() + " Face detected";

Graphics g = Graphics.FromImage(pictureBox2.Image);

foreach (var face in faceObjects)

{

g.DrawRectangle(Pens.DeepSkyBlue, face);

Frame.Draw(face, new Bgr(Color.Red), 3);

Bitmap c = Frame.ToBitmap();

Bitmap bmp = new Bitmap(face.Width, face.Height);

Graphics gg = Graphics.FromImage(bmp);

gg.DrawImage(c, 0, 0, face, GraphicsUnit.Pixel);

//pictureBox2.Image = bmp;

bmp.Save("myface(accord) " + DateTime.Now.Second.ToString());

MessageBox.Show("Face Detected!!!, Face Save as:" + "myface(accord) " + DateTime.Now.Second.ToString(), "Face Detection Successfully", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

//g.Dispose()

}

catch (Exception er)

{

MessageBox.Show(er.Message,"Face Detection Failure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void btnUpload\_Click(object sender, EventArgs e)

{

pictureBox2.Image = pictureBox1.Image;

//Bitmap uploadImage = new Bitmap(pictureBox1.Image);

//pictureBox2.Image = resizeImage(uploadImage, 300, 300);

//pictureBox2.Size = pictureBox2.Image.Size;

//pictureBox2.Image = Image.FromFile(Application.StartupPath + "\\test.jpg");

// pictureBox2.Image = Image.FromFile(Application.StartupPath + "\\Lucky.jpg");

}

private static Image resizeImage(Image image, int imgWidth, int imgHeight)

{

Bitmap newImage = new Bitmap(imgWidth, imgHeight);

Graphics g = Graphics.FromImage((Image)newImage);

g.InterpolationMode = System.Drawing.Drawing2D.InterpolationMode.High;

g.DrawImage(image, 0, 0, imgWidth, imgHeight);

return newImage;

}

private void button2\_Click(object sender, EventArgs e)

{

timer1.Enabled = false;

this.Hide();

}

private void button3\_Click(object sender, EventArgs e)

{

if (button3.Text == "&Stop")

{

timer1.Enabled = false;

button3.Text = "&Play Again";

}

else if (button3.Text == "&Play Again")

{

timer1.Enabled = true;

button3.Text = "&Stop";

}

}

}

}