Vidzemes Augstskola

Inženierzinātņu fakultāte

**Attēlu apstrādes programma, izmantojot “opencv” bibliotēku**

Kursa darbs

Autors: Elīna Smoļaka

Stud. apl. Nr.: 17025

Valmiera 2019

Saturs

[1. Programmas apraksts 3](#_Toc7447152)

[2. Bibliotēkas “OPENCV” apraksts 4](#_Toc7447153)

[3. Bibliotēkas pielietošana programmā 5](#_Toc7447154)

[4. Programmas darbības demonstrācija 6](#_Toc7447155)

[5. Komentēts programmas teksts 8](#_Toc7447156)

[Literatūra 16](#_Toc7447157)

# Programmas apraksts

Programma paredzēta attēlu apstrādei. Programmā iespējams izvēlēties attēlu no mapes datorā, izmantojot “Browse” opciju. Lai iegūtu ceļu (path) uz lietotāja izvēlēto attēlu un to apstrādātu, tiek izmantota Māršala metode, kas pārveido ceļu no “string” datu tipa uz “char” datu tipu. Lietotājam atļauts izvēlēties tikai tādus attēlus, kas satur paplašinājumu .bmp un .jpg.

BMP faila formāts, kas pazīstams arī kā “bitmap”, ir rastra grafikas attēla faila formāts, ko izmanto, lai saglabātu bitkartes digitālos attēlus neatkarīgi no displeja ierīces. JPEG saspiešanu izmanto vairākos attēlu failu formātos. JPEG ir visizplatītākais attēla formāts, ko izmanto digitālās kameras un citas fotografēšanas attēlu uzņemšanas ierīces (Nichols 2018).

Pēc attēla izvēlēšanās un apstiprināšanas, izmantojot “Process” jeb attēla apstrādi, izvēlētais attēls tiek pārveidots, pamatojoties uz četriem programmā iekļautajiem apstrādes efektiem.

Attēla izvēlēšanos, pārlūkojot datora resursus, un apstrādi iespējams veikt atkāroti. Programmai izveidota lietotāja saskarne, kas nodrošina mijiedarbību starp izstrādāto programmu un lietotāju. Programmas logu var samazināt, paplašināt, aizvērt.

**1.1. tabula.** *Funkcijas un to apraksti*

|  |  |
| --- | --- |
| **Funkcija** | **Apraksts** |
| ToColorful | Pārveido attēlu, piešķirot tam efektu. |
| ToRotate | Pārveido attēlu, piešķirot tam efektu. |
| ToBinary | Pārveido attēlu, piešķirot tam efektu. |
| ToAdaptiveBinary | Pārveido attēlu, piešķirot tam efektu. |
| ToResize | Maina attēla izmēru, lai attēlotu tā oriģinālu. |
| ConvertString2Char | Māršala metode, kas pārveido attēla ceļu (path) no “string” datu tipa uz “char” datu tipu. |
| btnBrowser\_Click | Nosaka, kā reaģē poga “Browse”. |
| buttonProcess\_Click | Nosaka, kā reaģē poga “Process”. |
| DrawCVImage | Ievieto attēlu tam paredzētajā laukumā, attēlo to, piešķirot efektu. |

# 

# Bibliotēkas “OPENCV” apraksts

“OpenCV” (Open Source Computer Vision Library) ir atvērtā koda datoru redzes un mašīnu mācīšanās programmatūras bibliotēka. “OpenCV” bibliotēka tika uzbūvēta, lai nodrošinātu kopīgu infrastruktūru datora redzes lietojumprogrammām un paātrinātu mašīnas uztveres izmantošanu komerciālos produktos. OpenCV ir BSD (Berkeley Software Distribution) licencēts produkts, un tas atvieglo uzņēmumiem izmantot un modificēt kodu.

Bibliotēkā ir vairāk nekā 2500 optimizēti algoritmi, kas ietver visaptverošu gan klasisku, gan mūsdienīgu datoru redzes un mašīnu mācīšanās algoritmu kopumu. Šos algoritmus var izmantot, lai atklātu un atpazītu sejas, identificētu objektus, klasificētu cilvēka darbības videoklipos, izsekotu kameras kustības, izsekotu kustīgus objektus, iegūtu trīs dimensiju objektu modeļus, ražotu trīs dimensiju punktu mākoņus no stereo kamerām, kopētu attēlus kopā, lai iegūtu augstu izšķirtspēju attēlos, atrastu līdzīgus attēlus no datu bāzes, noņemt sarkanās acs efektu no attēliem, kas uzņemti, izmantojot zibspuldzi, izsekot acu kustībām, atpazīt ainavu un izveidot marķierus, lai pārklātu to ar paplašināto realitāti utt. “OpenCV” ir vairāk nekā 47 tūkstoši lietotāju. Kopējais un paredzamais lejupielāžu skaits pārsniedz 18 miljonus. Bibliotēku plaši izmanto uzņēmumos, pētniecības grupās un valdības struktūrās.

Kopā ar pazīstamu zīmolu uzņēmumiem, piemēram, “Google”, “Yahoo”, “Microsoft”, “Intel”, “IMB”, “Sony”, “Honda”, “Toyota”, kas izmanto bibliotēku, ir arī daudz salīdzinoši jaunu uzņēmumu, piemēram, “Applied Minds”, “VideoSurf” un “Zeitera”, kas plaši izmanto “OpenCV”. “OpenCV” ir plašs pielietojums, sākot no ielu attēlu apvienošanu, to izmanto novērošanas video radīšanā Izraēlā, uzraugot raktuvju aprīkojumu Ķīnā, palīdzot robotiem pārvietoties un uzņemt objektus, kā arī pārbaudot etiķetes uz produktiem rūpnīcās visā pasaulē uz sejas noteikšanu. “OPENCV” palīdz uzlabot interaktīvās mākslas jomu Eiropā, piemēram, Spānijā un Ņujorkā.

Tajā ir C ++, Python, Java un MATLAB programmēšanas valodu saskarne un atbalsts “Windows”, “Linux”, “Android” un “Mac” operētājsistēmām. “OpenCV” galvenokārt orientējas uz reāllaika lietojumiem (OPENCV team 2019).

# Bibliotēkas pielietošana programmā

Bibliotēka tiek izmantota, lai pielietotu “Mat” datu tipu, kas nodrošina darbības ar attēliem, piemēram, attēla glabāšanu, apstrādi. Piemērs attēla glabāšanai un pārveidošanai:

Mat img;

//resize image

Mat ToResize(Mat img) {

Mat done;

//define new size

resize(img, done, cv::Size(320, 240), 0, 0, 1);

return done;

}

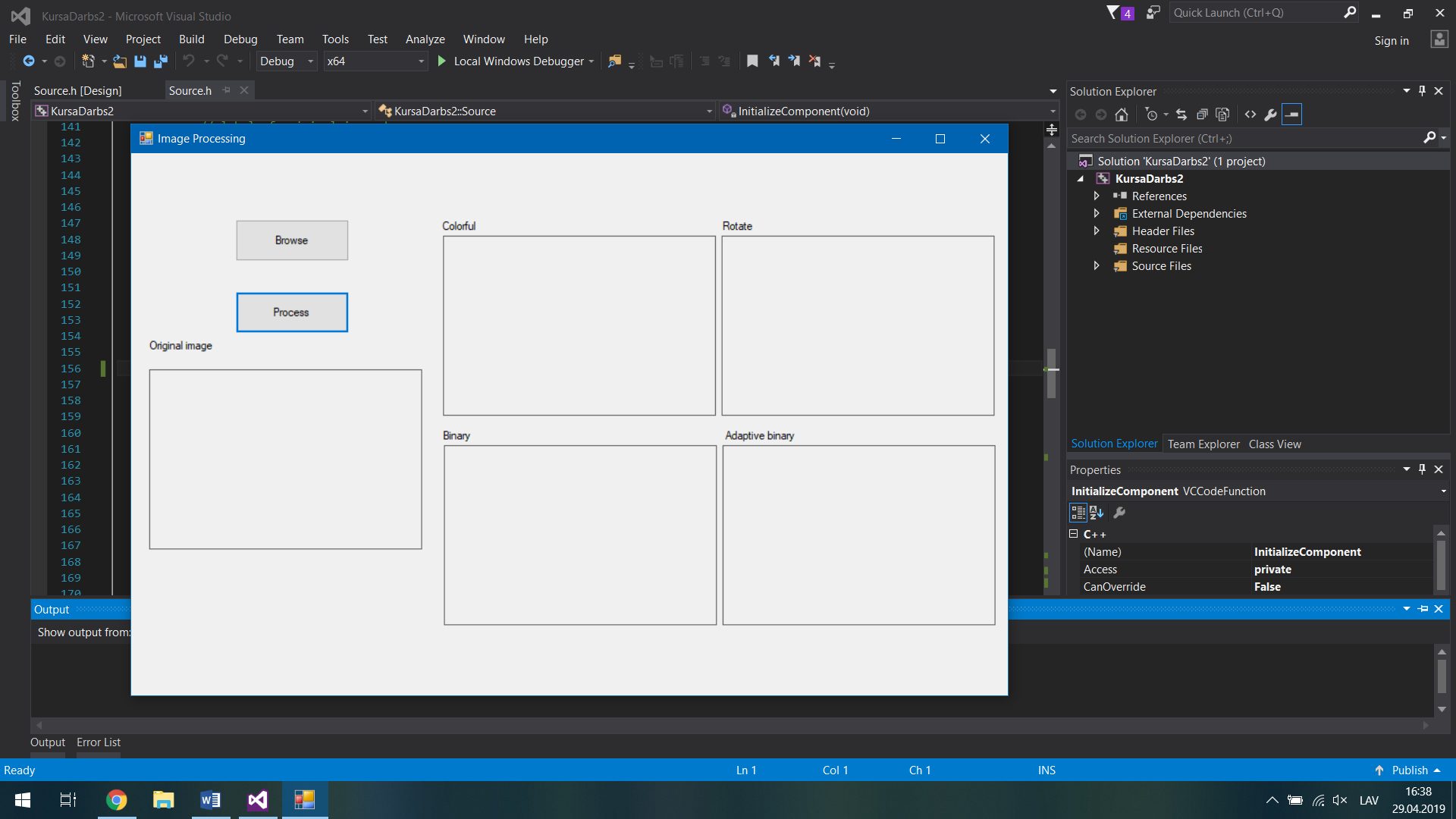
Piemērs attēla izvēlei un ceļa (path) pārveidošanai, izmantojot “Browse” opciju lietotāja saskarnē, “imread”, kas iekļauta “OPENCV” bibliotēkā:

img = imread(ConvertString2Char(dgOpen->FileName));

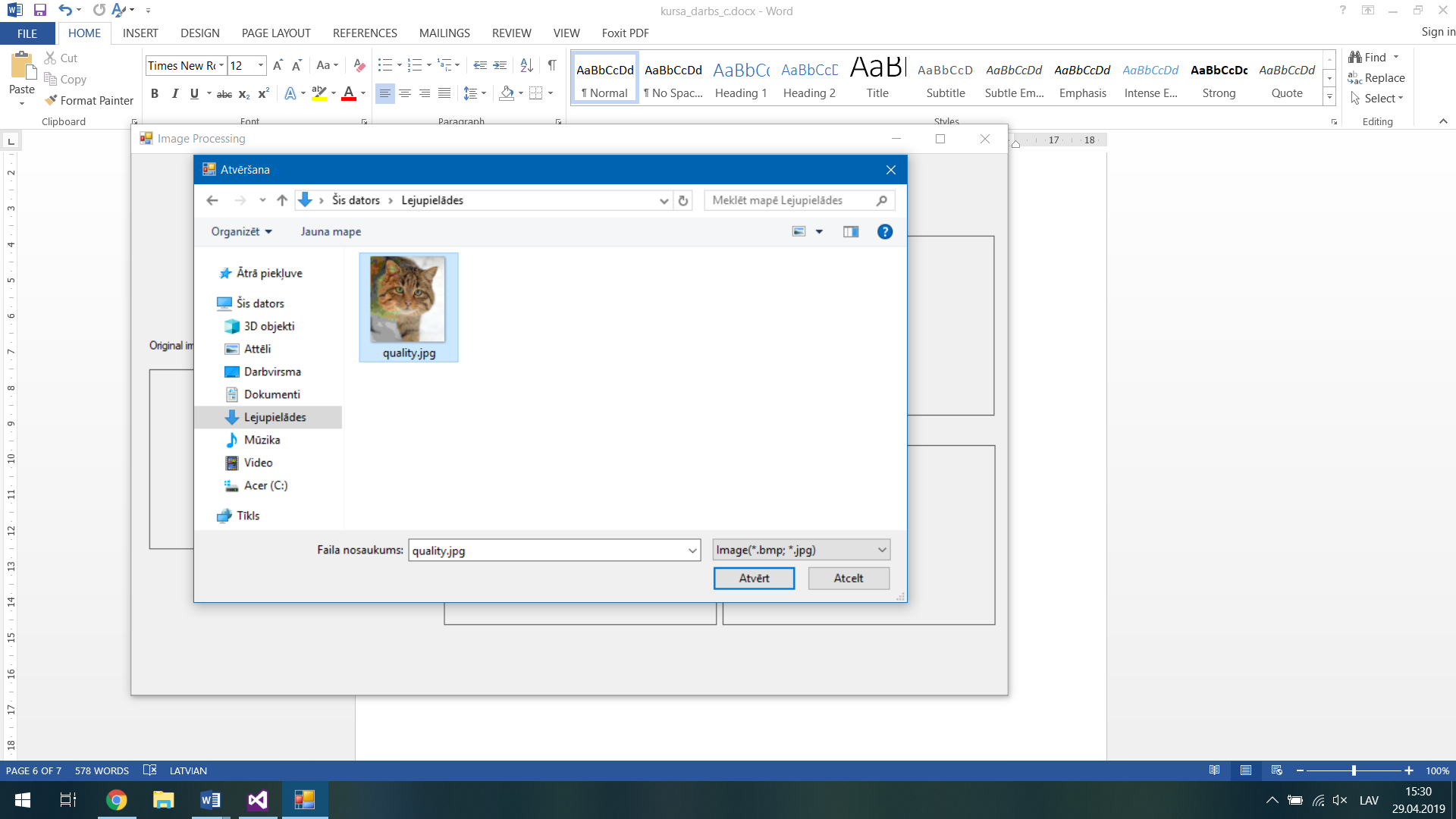
Programmā bibliotēka tiek izmantota, lai apstrādātu attēlus, uzstādot tiem efektus. Piemēram, ar “CV\_RGB2GRAY” palīdzību iespējams uzstādīt attēlam efektu.

# Programmas darbības demonstrācija

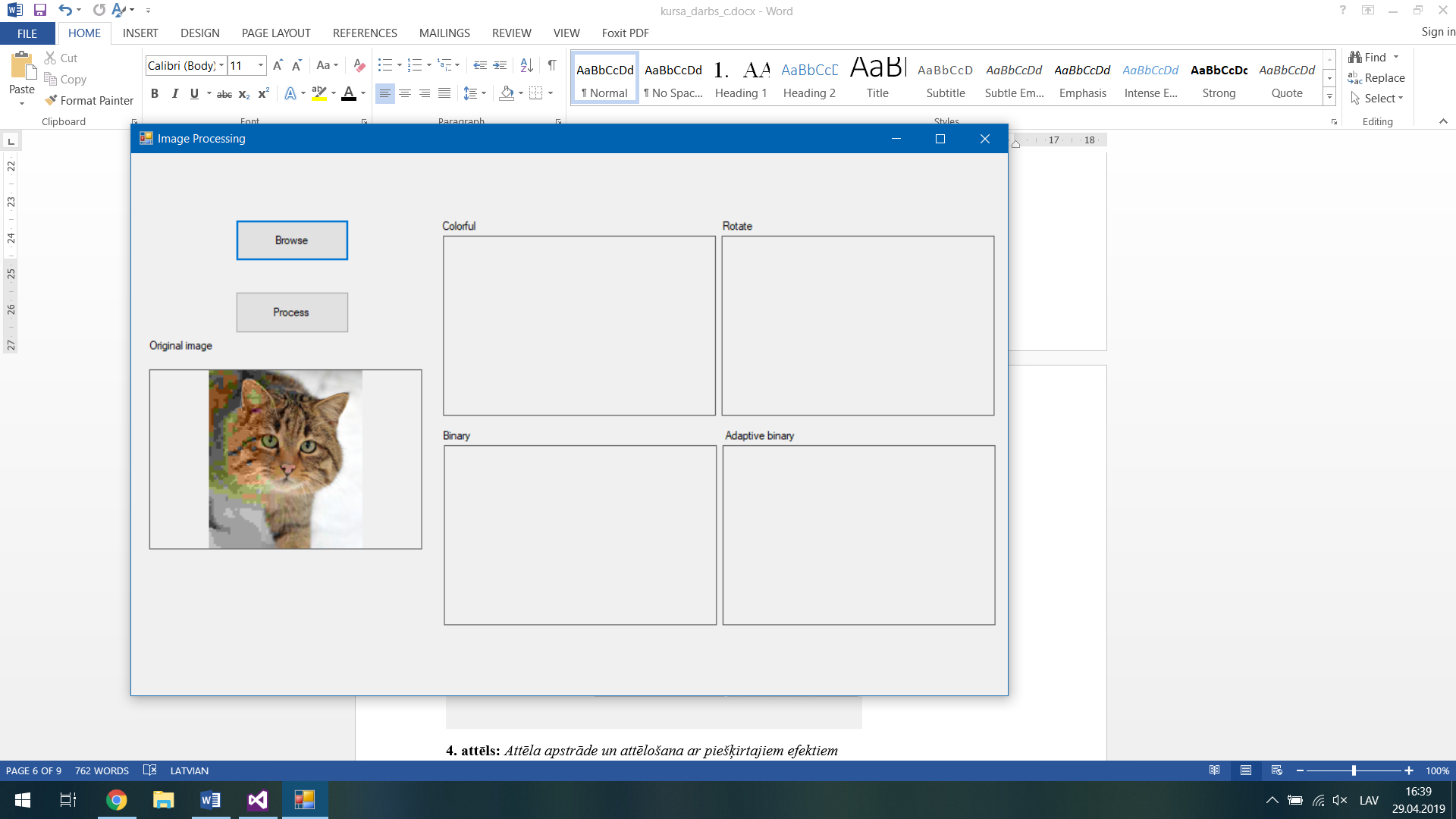
Programma tiek atvērta (skatīt 1. attēlu), pēc tam tiek izvēlēts attēls, izmantojot “Browse” opciju (skatīt 2. attēlu). Kreisajā pusē tiek attēlots izvēlētais attēls (skatīt 3. attēlu), kas tiek apstrādāts, piešķirot tam efektus, izmantojot “Process” opciju (skatīt 4. attēlu). Šīs darbības iespējams atkārot. Ja lietotājs izvēlas “Process” opciju pirms “Browse”, programma nereaģē uz šo darbību, jo nav izvēlēts attēls, ko apstrādāt, piešķirot tam efektus.



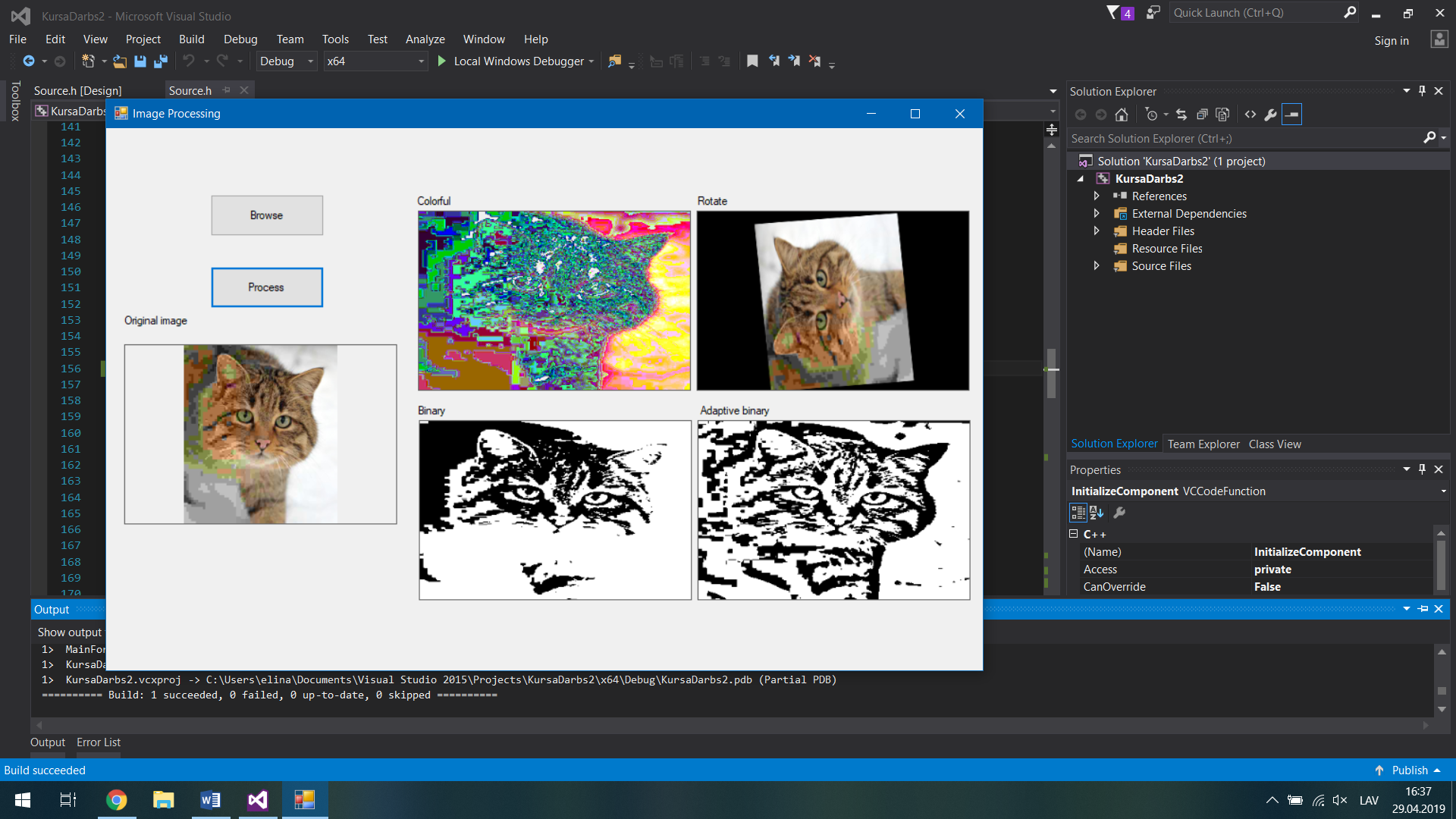
**1. attēls:** *Programmas logs*



**2. attēls:** *Attēla izvēle*



**3. attēls:** *Oriģināla attēlošana*



**4. attēls:** *Attēla apstrāde un attēlošana ar piešķirtajiem efektiem*

# Komentēts programmas teksts

#include <opencv2/core/core.hpp>

#include <opencv2/highgui/highgui.hpp>

#include <opencv2/imgcodecs/imgcodecs.hpp>

#include <opencv2/imgproc/imgproc.hpp>

namespace KursaDarbs2 {

using namespace System::Windows::Forms; //form of application

using namespace System::Drawing; //to set point, size etc. of components

using namespace System::Runtime::InteropServices; //for Marshal

using namespace cv; //for Mat, treshold, point2f etc.

//Mat type for original image

Mat img;

public ref class Source : public System::Windows::Forms::Form{

public:

//constructor

Source(void){

//components into form

InitializeComponent();

}

//destructor

~Source(){

if (components){

delete components;

}

}

private:

//buttons

System::Windows::Forms::Button^ buttonBrowse;

System::Windows::Forms::Button^ buttonProcess;

//boxes for images

System::Windows::Forms::PictureBox^ picOriginal;

System::Windows::Forms::PictureBox^ firstBox;

System::Windows::Forms::PictureBox^ secondBox;

System::Windows::Forms::PictureBox^ thirdBox;

System::Windows::Forms::PictureBox^ fourthBox;

//labels for picture boxes

System::Windows::Forms::Label^ labelOrigImage;

System::Windows::Forms::Label^ labelEffect1;

System::Windows::Forms::Label^ labelEffect2;

System::Windows::Forms::Label^ labelEffect3;

System::Windows::Forms::Label^ labelEffect4;

private:

//required variable for design, encapsulates components

System::ComponentModel::Container ^components;

void InitializeComponent(void){

//initilize instances of the classes with gcnew

this->buttonBrowse = (gcnew System::Windows::Forms::Button());

this->buttonProcess = (gcnew System::Windows::Forms::Button());

this->picOriginal = (gcnew System::Windows::Forms::PictureBox());

this->firstBox = (gcnew System::Windows::Forms::PictureBox());

this->secondBox = (gcnew System::Windows::Forms::PictureBox());

this->thirdBox = (gcnew System::Windows::Forms::PictureBox());

this->fourthBox = (gcnew System::Windows::Forms::PictureBox());

this->labelOrigImage = (gcnew System::Windows::Forms::Label());

this->labelEffect1 = (gcnew System::Windows::Forms::Label());

this->labelEffect2 = (gcnew System::Windows::Forms::Label());

this->labelEffect3 = (gcnew System::Windows::Forms::Label());

this->labelEffect4 = (gcnew System::Windows::Forms::Label());

//button BROWSE

this->buttonBrowse->Location = System::Drawing::Point(110, 70);

this->buttonBrowse->Margin = System::Windows::Forms::Padding(2, 3, 2, 3);

this->buttonBrowse->Name = L"buttonBrowse";

this->buttonBrowse->Size = System::Drawing::Size(120, 44);

this->buttonBrowse->TabIndex = 5;

this->buttonBrowse->Text = L"Browse";

this->buttonBrowse->Click += gcnew System::EventHandler(this, &Source::btnBrowser\_Click);

//button PROCESS

this->buttonProcess->Location = System::Drawing::Point(110, 146);

this->buttonProcess->Margin = System::Windows::Forms::Padding(2, 3, 2, 3);

this->buttonProcess->Name = L"buttonProcess";

this->buttonProcess->Size = System::Drawing::Size(120, 44);

this->buttonProcess->TabIndex = 4;

this->buttonProcess->Text = L"Process";

this->buttonProcess->Click += gcnew System::EventHandler(this, &Source::buttonProcess\_Click);

//original image box

this->picOriginal->BorderStyle = System::Windows::Forms::BorderStyle::FixedSingle;

this->picOriginal->Location = System::Drawing::Point(19, 228);

this->picOriginal->Margin = System::Windows::Forms::Padding(3, 4, 3, 4);

this->picOriginal->Name = L"picOriginal";

this->picOriginal->Size = System::Drawing::Size(288, 190);

this->picOriginal->SizeMode = System::Windows::Forms::PictureBoxSizeMode::Zoom;

this->picOriginal->TabIndex = 2;

this->picOriginal->TabStop = false;

//first effect box

this->firstBox->BorderStyle = System::Windows::Forms::BorderStyle::FixedSingle;

this->firstBox->Location = System::Drawing::Point(329, 87);

this->firstBox->Margin = System::Windows::Forms::Padding(3, 4, 3, 4);

this->firstBox->Name = L"firstBox";

this->firstBox->Size = System::Drawing::Size(288, 190);

this->firstBox->SizeMode = System::Windows::Forms::PictureBoxSizeMode::Zoom;

this->firstBox->TabIndex = 2;

this->firstBox->TabStop = false;

//second effect box

this->secondBox->BorderStyle = System::Windows::Forms::BorderStyle::FixedSingle;

this->secondBox->Location = System::Drawing::Point(623, 87);

this->secondBox->Margin = System::Windows::Forms::Padding(3, 4, 3, 4);

this->secondBox->Name = L"secondBox";

this->secondBox->Size = System::Drawing::Size(288, 190);

this->secondBox->SizeMode = System::Windows::Forms::PictureBoxSizeMode::Zoom;

this->secondBox->TabIndex = 2;

this->secondBox->TabStop = false;

//third effect box

this->thirdBox->BorderStyle = System::Windows::Forms::BorderStyle::FixedSingle;

this->thirdBox->Location = System::Drawing::Point(330, 308);

this->thirdBox->Margin = System::Windows::Forms::Padding(3, 4, 3, 4);

this->thirdBox->Name = L"thirdBox";

this->thirdBox->Size = System::Drawing::Size(288, 190);

this->thirdBox->SizeMode = System::Windows::Forms::PictureBoxSizeMode::Zoom;

this->thirdBox->TabIndex = 2;

this->thirdBox->TabStop = false;

//fourth effect box

this->fourthBox->BorderStyle = System::Windows::Forms::BorderStyle::FixedSingle;

this->fourthBox->Location = System::Drawing::Point(624, 308);

this->fourthBox->Margin = System::Windows::Forms::Padding(3, 4, 3, 4);

this->fourthBox->Name = L"fourthBox";

this->fourthBox->Size = System::Drawing::Size(288, 190);

this->fourthBox->SizeMode = System::Windows::Forms::PictureBoxSizeMode::Zoom;

this->fourthBox->TabIndex = 2;

this->fourthBox->TabStop = false;

// label of original image box

this->labelOrigImage->AutoSize = true;

this->labelOrigImage->Location = System::Drawing::Point(17, 196);

this->labelOrigImage->Name = L"labelOrigImage";

this->labelOrigImage->Size = System::Drawing::Size(99, 17);

this->labelOrigImage->TabIndex = 3;

this->labelOrigImage->Text = L"Original image";

this->labelOrigImage->Click += gcnew System::EventHandler(this, &Source::labelOrigImage\_Click);

//label of image box effect1

this->labelEffect1->AutoSize = true;

this->labelEffect1->Location = System::Drawing::Point(326, 70);

this->labelEffect1->Name = L"labelEffect1";

this->labelEffect1->Size = System::Drawing::Size(39, 17);

this->labelEffect1->TabIndex = 3;

this->labelEffect1->Text = L"Colorful";

this->labelEffect1->Click += gcnew System::EventHandler(this, &Source::labelEffect1\_Click);

//label of image box effect2

this->labelEffect2->AutoSize = true;

this->labelEffect2->Location = System::Drawing::Point(621, 70);

this->labelEffect2->Name = L"labelEffect2";

this->labelEffect2->Size = System::Drawing::Size(50, 17);

this->labelEffect2->TabIndex = 3;

this->labelEffect2->Text = L"Rotate";

this->labelEffect2->Click += gcnew System::EventHandler(this, &Source::labelEffect2\_Click);

//label of image box effect3

this->labelEffect3->AutoSize = true;

this->labelEffect3->Location = System::Drawing::Point(326, 291);

this->labelEffect3->Name = L"labelEffect3";

this->labelEffect3->Size = System::Drawing::Size(48, 17);

this->labelEffect3->TabIndex = 3;

this->labelEffect3->Text = L"Binary";

this->labelEffect3->Click += gcnew System::EventHandler(this, &Source::labelEffect3\_Click);

//label of image box effect4

this->labelEffect4->AutoSize = true;

this->labelEffect4->Location = System::Drawing::Point(624, 291);

this->labelEffect4->Name = L"labelEffect4";

this->labelEffect4->Size = System::Drawing::Size(106, 17);

this->labelEffect4->TabIndex = 3;

this->labelEffect4->Text = L"Adaptive binary";

this->labelEffect4->Click += gcnew System::EventHandler(this, &Source::labelEffect4\_Click);

//Main window of application

//size of application window

this->AutoScaleDimensions = System::Drawing::SizeF(10, 18);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->BackColor = System::Drawing::SystemColors::Control;

this->ClientSize = System::Drawing::Size(925, 572);

//padding, title and method for application window

this->Margin = System::Windows::Forms::Padding(3, 4, 3, 4);

this->Text = L"Image Processing";

//add controls - buttons, labels, boxes - to application window

this->Controls->Add(this->buttonProcess);

this->Controls->Add(this->buttonBrowse);

this->Controls->Add(this->labelOrigImage);

this->Controls->Add(this->labelEffect1);

this->Controls->Add(this->labelEffect2);

this->Controls->Add(this->labelEffect3);

this->Controls->Add(this->labelEffect4);

this->Controls->Add(this->picOriginal);

this->Controls->Add(this->firstBox);

this->Controls->Add(this->secondBox);

this->Controls->Add(this->thirdBox);

this->Controls->Add(this->fourthBox);

}

//methods to process and convert images

private:

//convert image to colorful

Mat ToColorful(Mat img) {

Mat done;

cvtColor(img, done, CV\_RGB2GRAY);

//return converted image

return done;

}

//rotate image

Mat ToRotate(Mat img, double angle, double scale) {

Mat done;

//rotate with center point defined

Point2f center(img.cols / 2, img.rows / 2);

//create a matrix with center point, angle and scale as a defination

Mat mat\_rot = getRotationMatrix2D(center, angle, scale);

//wrapAffine converts source image using matrix

warpAffine(img, done, mat\_rot, img.size());

return done;

}

//convert image to binary image

Mat ToBinary(Mat img) {

Mat done;

//use threshold method from namespace cv

threshold(img, done, 100, 255, CV\_THRESH\_BINARY);

return done;

}

//convert image to adaptive binary image

Mat ToAdaptiveBinary(Mat img) {

Mat done;

//adaptiveThreshold coverts in beter quality performance

adaptiveThreshold(img, done, 255, CV\_ADAPTIVE\_THRESH\_MEAN\_C, CV\_THRESH\_BINARY, 35, 5);

return done;

}

//resize image

Mat ToResize(Mat img) {

Mat done;

//define new size

resize(img, done, cv::Size(320, 240), 0, 0, 1);

return done;

}

//Marshal method - convert to char

private: char\* ConvertString2Char(System::String^ str) {

char\* str2 = (char\*)(void\*)Marshal::StringToHGlobalAnsi(str);

return str2;

}

//button BROWSE action

private: System::Void btnBrowser\_Click(System::Object^ sender, System::EventArgs^ e){

//load image and insert in box

OpenFileDialog^ dgOpen = gcnew OpenFileDialog();

dgOpen->Filter = "Image(\*.bmp; \*.jpg)|\*.bmp;\*.jpg|All files (\*.\*)|\*.\*||";

if (dgOpen->ShowDialog() == System::Windows::Forms::DialogResult::Cancel){

return;

}

Bitmap^ bitmapImg = gcnew Bitmap(dgOpen->FileName);

picOriginal->Image = bitmapImg;

picOriginal->Refresh();

//get img name - converted to char by Marshal method

//need for converting image

img = imread(ConvertString2Char(dgOpen->FileName));

}

//button PROCESS action

private: System::Void buttonProcess\_Click(System::Object^ sender, System::EventArgs^ e){

Mat resizeimg;

Mat colorful;

Mat rotate;

Mat binary;

Mat adaptivebinary;

if (!img.empty()) {

//at first resize image to fix img problem

resizeimg = ToResize(img);

//convert

colorful = ToColorful(resizeimg);

//show in box

DrawCVImage(firstBox, colorful);

//angle 95 degree and zoom scale 0.7

rotate = ToRotate(resizeimg, 95, 0.7);

DrawCVImage(secondBox, rotate);

binary = ToBinary(colorful);

DrawCVImage(thirdBox, binary);

adaptivebinary = ToAdaptiveBinary(colorful);

DrawCVImage(fourthBox, adaptivebinary);

}

}

//to show Mat image in pictureBox (method code from OpenCV website)

void DrawCVImage(System::Windows::Forms::Control^ control, cv::Mat& colorImage){

System::Drawing::Graphics^ graphics = control->CreateGraphics();

System::IntPtr ptr(colorImage.ptr());

System::Drawing::Bitmap^ b;

switch (colorImage.type()){

case CV\_8UC3: //non-grayscale images are correctly displayed here

b = gcnew System::Drawing::Bitmap(colorImage.cols, colorImage.rows, colorImage.step,

System::Drawing::Imaging::PixelFormat::Format24bppRgb, ptr);

break;

case CV\_8UC1: //grayscale images are incorrectly displayed here

b = gcnew System::Drawing::Bitmap(colorImage.cols, colorImage.rows, colorImage.step,

System::Drawing::Imaging::PixelFormat::Format8bppIndexed, ptr);

break;

default: "Error occured!";

break;

}

System::Drawing::RectangleF rect(0, 0, (float)control->Width, (float)control->Height);

graphics->DrawImage(b, rect);

}

//design auto-generated methods

System::Void labelEffect1\_Click(System::Object^ sender, System::EventArgs^ e) {

}

System::Void labelOrigImage\_Click(System::Object^ sender, System::EventArgs^ e) {

}

System::Void labelEffect2\_Click(System::Object^ sender, System::EventArgs^ e) {

}

System::Void labelEffect3\_Click(System::Object^ sender, System::EventArgs^ e) {

}

System::Void labelEffect4\_Click(System::Object^ sender, System::EventArgs^ e) {

}

};

}

# Literatūra

1. Nichols Jessie-Lee. 10 Types of Image File Extensions and When to Use Them. <https://blog.hubspot.com/insiders/different-types-of-image-files>, 04/10/2018
2. OPENCV team. 2019. About OPENCV. <https://opencv.org/about/>