MULTIMEDIA PROGRAMMIRLEMESI

Begnarlyýewiç Serdar Orazdurdyýew

KOICA Korea International Cooperation Agency 부천대학교 BUCHEON UNIVERSITY

Mazmuny

1. Sanly şekilleri işlemek bilen tanyşlyk

1.1 Sanly şekilleri işlemek 7

1.1.1 Nusga almak we mukdaryny kesgitlemek 7

1.1.2 Analogdan sanly görnüşe öwürmek 7

1.1.3 Surat çeşmesi 8

1.1.4 Suratyň ýagtylygyny üýtgetmek 8

1.1.5 Suratyň ýagtylyk ädimleri 9

1.1.6 Suratyň ýagtylygynyň ölçegi 9

1.1.7 Ýagtylyk spektri 10

1.1.8 Ýagtylyk we pigment garyndylary 10

1.1.9 R, G, B suraty 11

1.1.10 Sanly şekiliň görnüşi 11

1.1.11 Sanly şekil 12

2. OpenCV bilen tanyşlyk

2.1 OpenCV? 13

2.1.1 OpenCV - açyk çeşmeli kompýuter kitaphanasy 13

2.2 OpenCV-ni gurnamak 13

2.2.1 C ++ ulanyp, OpenCV-ni programmirlemek üçin şertler 13

2.2.2 Visual Studio Community 2017-ni gurnamak 14

2.2.3 OpenCV-ni ýüklemek 14

2.2.4 Windows ulgamynyň daşky gurşaw ýoluna goşmak 15

2.3 Visual Studio 2017 açyk rezýumesini düzmek 16

2.4 OpenCV programmasynda mysal ýazmak 24

3. OpenCV toparlary

3.1 Point_topary 29

3.2 Size_ topary 30

3.3 Rect_ topary 31

3.4 Vec topary 32
3.5 Scalar_ topary 33
3.6 Mat topary 33
3.7 Vector topary 38
3.8 Range topary 40
3.9 Matrisa amal funksiýasy 40
3.10 saturate_cast < > 42

4. OpenCV ulanyjy interfeýsleri

4.1 Penjiräni dolandyrmak 43
4.2 Klawiatura hadysalaryny dolandyrmak 44
4.3 TrackBar hadysalaryny dolandyrmak 47
4.4 Çyzyk, gönüburçluk çyzgysy 49
4.5 Çyzgy teksti 51
4.6 Töweregiň çyzgysy 53
4.7 Surat faýlyny işläp taýýarlamak 55
4.8 Wideo-ny işläp taýýarlamak 59

5. Massiwlerde OpenCV amaly

5.1 Massiwleri işläp taýýarlamagyň esasy funksiýalary 65
5.2 Kanaly işläp taýýarlamak funksiýalary 66
5.3 Massiwleriň dört esasy hereketi 68
5.4 Massiw amalynyň köki, güýji, ululygy 70
5.5 Massiwlerde logiki bit amallar 70
5.6 Massiw amalynyň Maks., Min. absolýut derejesi 72
5.7 Massiwler bilen amallaryň statistikasy 74

6. OpenCV ulanyp, suratlary işläp taýýarlamak

6.1 Surat piksellerine girmek 77
6.2 Suratyň ýagtylyk bahasyny goşmak / aýyrmak 79
6.3 Gistogramma 81

6.4 Gistogrammany hasaplamak 83
6.5 Gistogramma çyzmak 85
6.6 Gistogrammany giňeltmek 86
6.7 Gistogrammany deňlemek 89

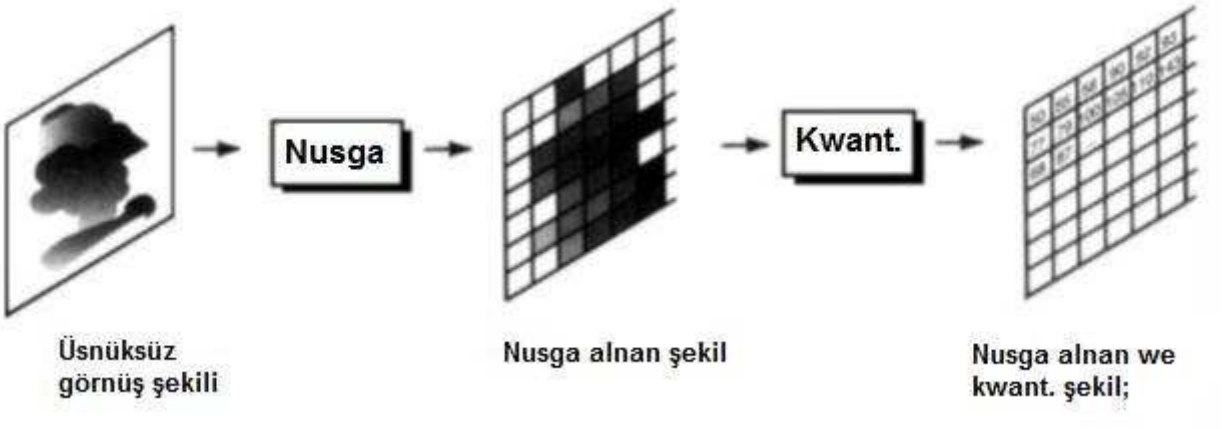
7. Açyk rezýume ulanyp dolamaklygy işläp taýýarlamak
7.1 Dolamaklygy işläp taýýarlamak 95
7.2 Näbellilik 96
7.3 Ýitileşdirmek 99
7.4 Gyralary kesgitlemek 102
7.5 Gyralary kesgitlemek (birmeňzeş / diferensial operator) 103
7.6 Gyralary kesgitlemek (Roberts / Sobel Mask Operatory) 107
7.7 Median filteri 111

8. Domen işleýişini üýtgetmek
8.1 Giňişlik ýygylygy 115
8.2 Fýurýeriň üýtgemegi 117
8.3 Diskret kosinus öwrülişigi (DCT) 118
8.4 Lenna 16x16 blogyň DCT koeffisiýentleri 119
8.5 Diskret kosinus öwrülişigi (DCT) 120

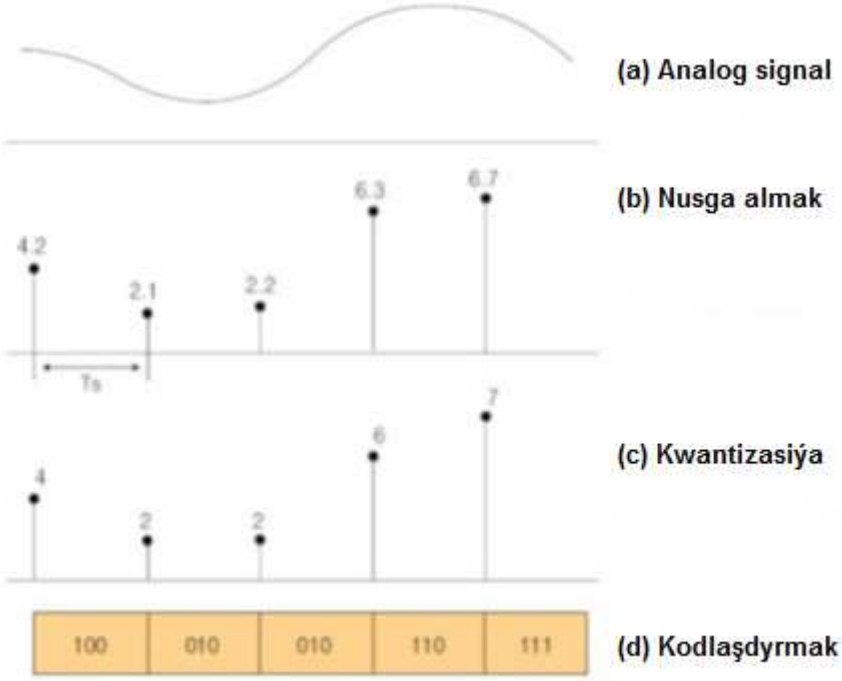
1. Sanly şekilleri işlemek bilen tanyşlyk

1.1 Sanly şekilleri işlemek

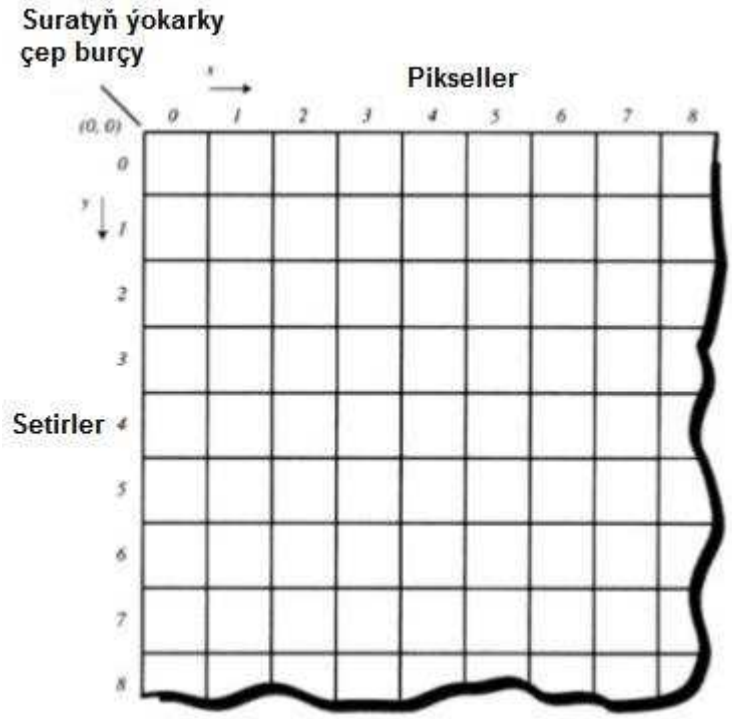
1.1.1 Nusga almak we mukdaryny kesgitlemek



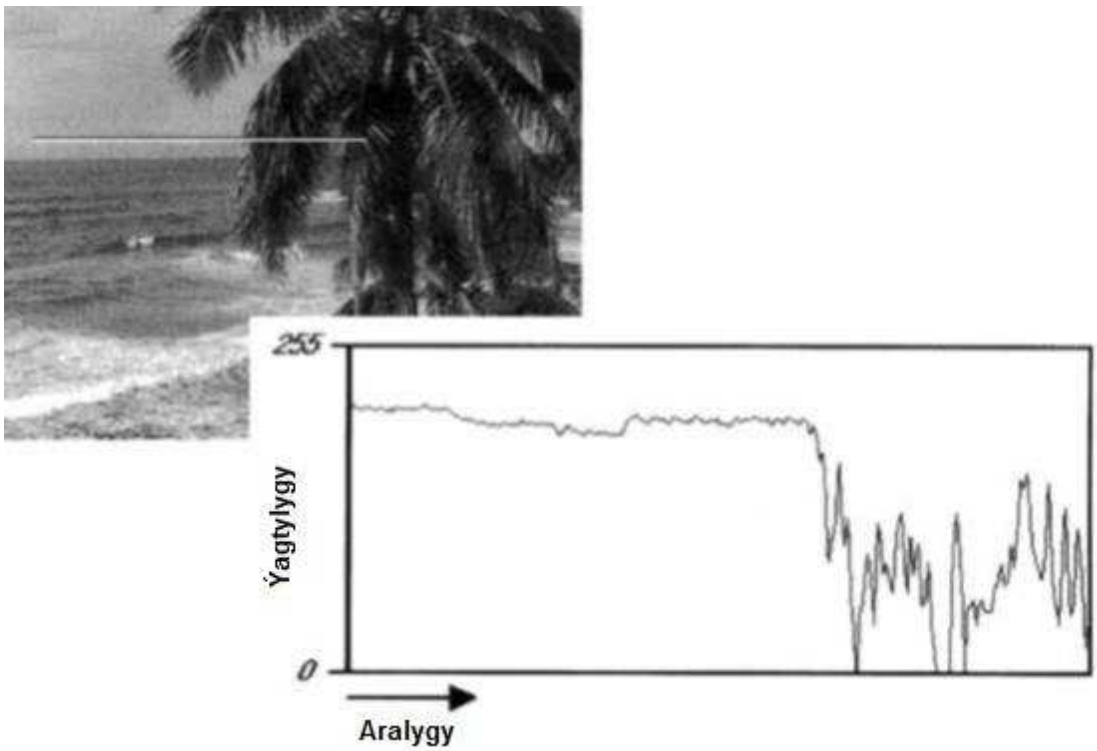
1.1.2 Analogdan sanly görnüşe öwürmek



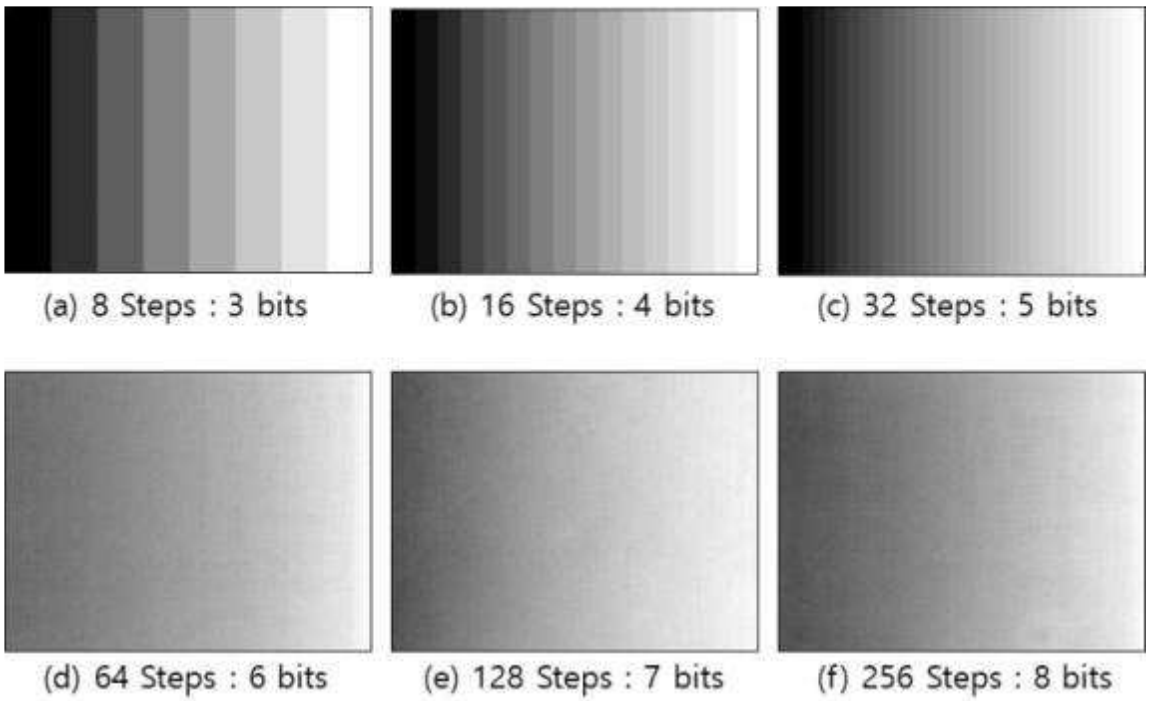
1.1.3 Surat çeşmesi



1.1.4 Suratyň ýagtylygyny üýtgetmek



1.1.5 Suratyň ýagtylyk ädimleri



1.1.6 Suratyň ýagtylygynyň ölçegi

1.1.7 Ýagtylyk spektri

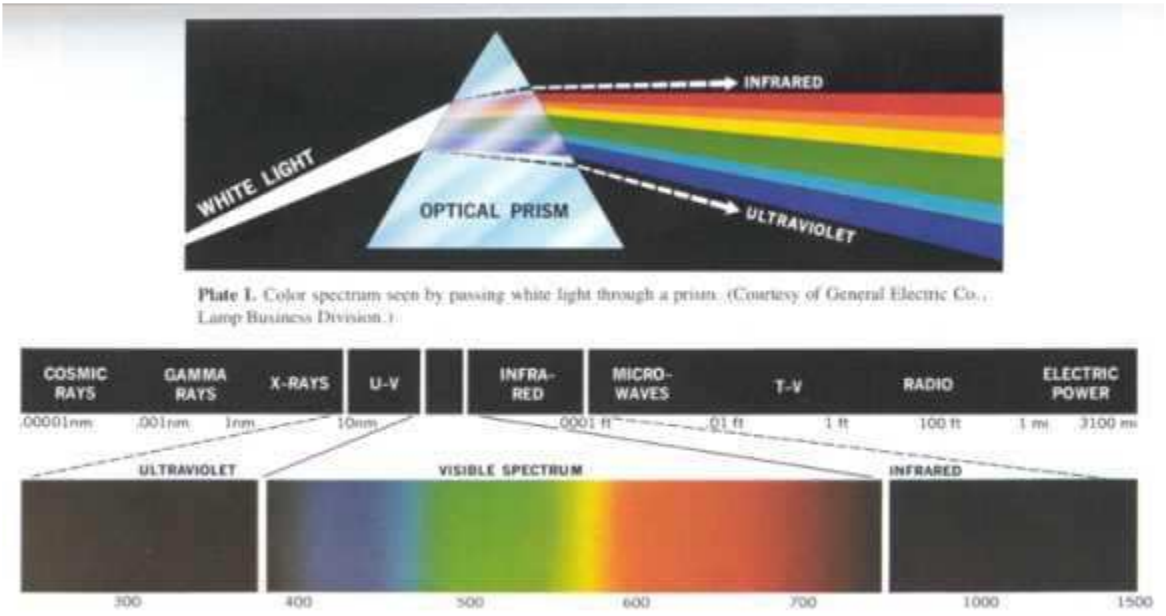
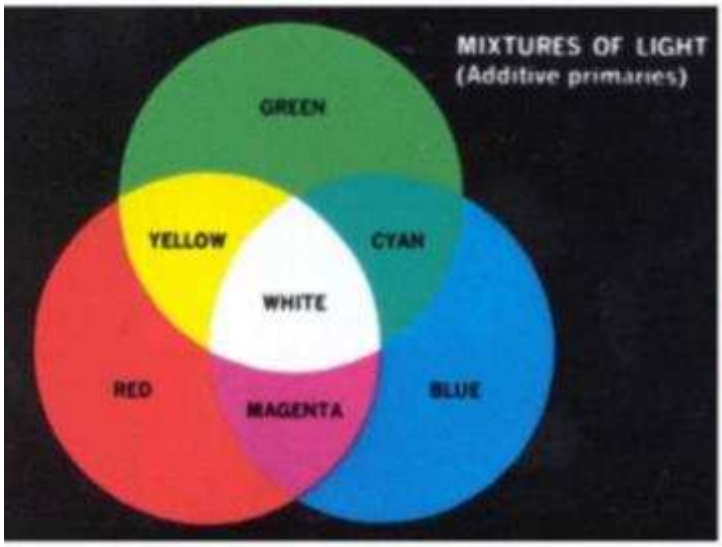


Plate I. Color spectrum seen by passing white light through a prism. (Courtesy of General Electric Co., Lamp Business Division.)

1.1.8 Ýagtylyk we pigment garyndylary



RGB reňk

CMY(K) reňk

1.1.9 R, G, B suraty

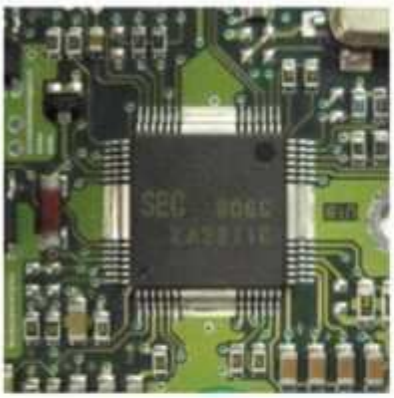
Asyl şekili

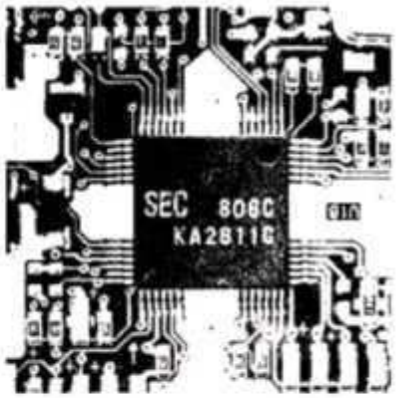
Gyzyl komponentli

Ýaşyl komponentli

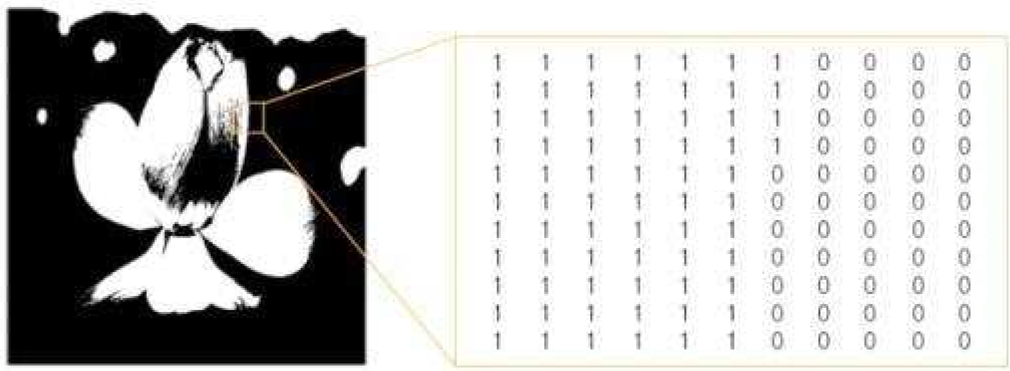
Gök komponentli

1.1.10 Sanly şekiliň görnüşi

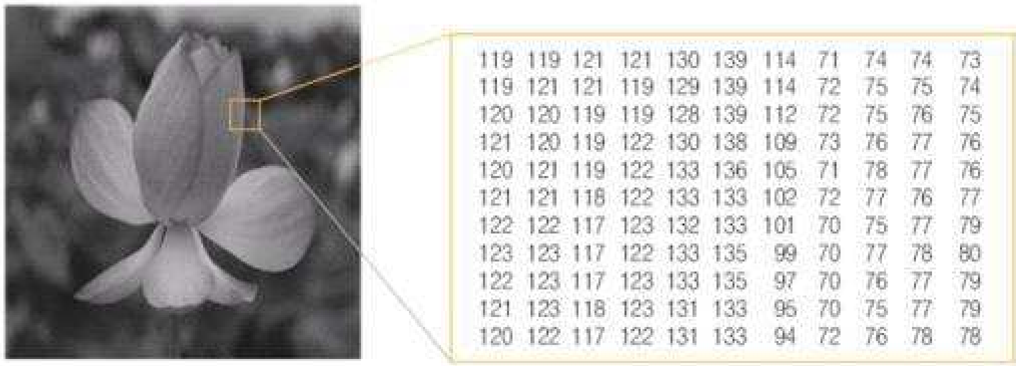




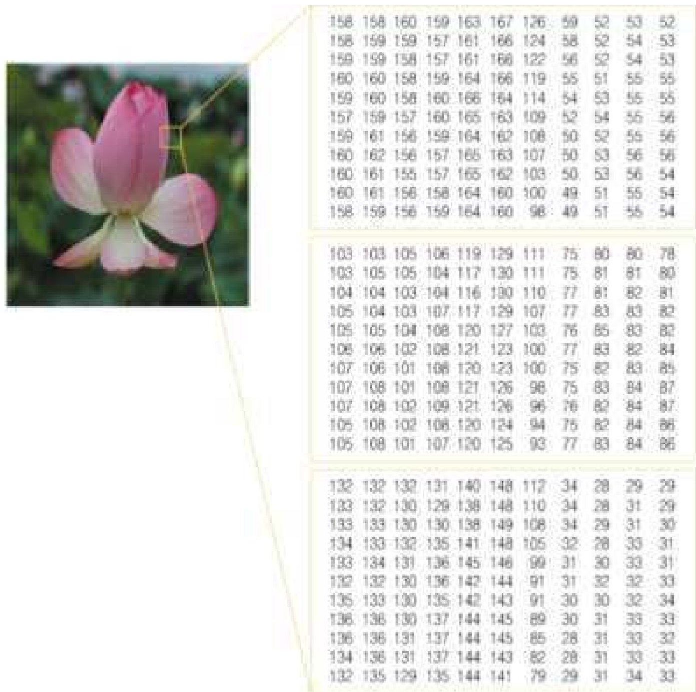
1.1.11 Sanly şekil



(a) Ikilik şekil



(b) Çal derejeli şekil



(c) Reňk şekili

2. OpenCV bilen tanyşlyk

2.1 OpenCV?

2.1.1 OpenCV - açyk çeşmeli kompýuter kitaphanasy

- Suraty işläp taýýarlamak we kompýuter görüşi üçin açyk çeşmeli kitaphana.
- 2500-den gowrak algoritmden ybarat.
- C, C ++, Python, Matlab üçin interfeýs goldawy.
- Windows, Linux, Android, Mac OS we ş.m. üçin operasion ulgamynyň goldawy.
- MX (MultiMedia eXtension) we SSE (Streaming SIMD Extensions) görkezmelerini ulanyp, çalt algoritm ýerine ýetirilişi.
- CUDA we OpenCL interfeýslerini işläp taýýarlamak.

2.2 OpenCV-ni gurnamak

2.2.1 C ++ ulanyp, OpenCV-ni programmirlemek üçin şertler

- Kompýuteriňizde 64 bitli Windows gurnalan bolmaly. (OpenCV diňe 64 bitli operasion ulgamyny goldaýar).
- Visual Studio 2017 C ++ programmirlemek üçin programma redaktory guraly hökmünde gurulmalydyr. (Visual Studio-nyň iň soňky wersiýasy 2019-dyr, emma häzirki wagtda diňe Visual Studio 2017 üçin OpenCV-ni goldaýar.)

2.2.2 Visual Studio Community 2017-ni gurnamak

- https://visualstudio.microsoft.com/ru/free-developer-offers/

Mugt iň gowy programmalary döretmek üçin zerur zatlaryň ählisi.

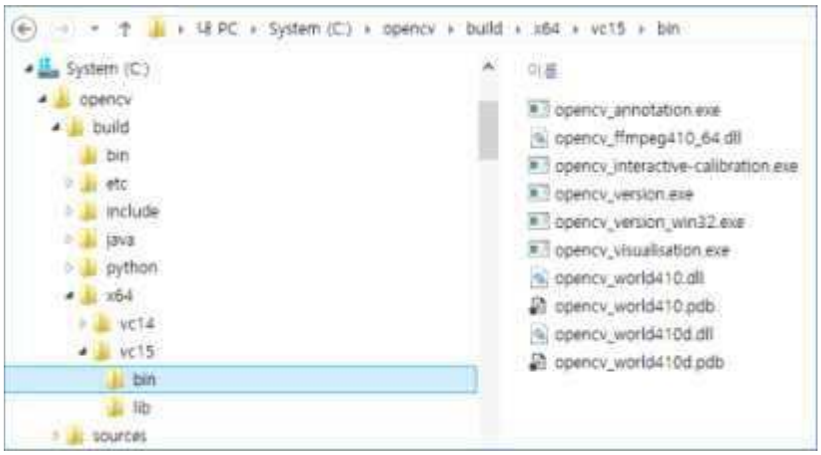
- Ýokardaky resmi web sahypasynda diňe iň soňky 2019 wersiýasyny ýükläp alyp bolýar, şol sebäpli internetden 2017 wersiýasyny gözleg we ýüklemek arkaly tapyp bilersiňiz.
- 30 günlük tanyşdyrylyş wersiýasy, soňra Microsoft-a agza bolup, Visual Studio-a giriň we mugt ulanmagy dowam etdiriň.

2.2.3 OpenCV-ni ýüklemek

- http://opencv.org>Releases
- Gurnama faýlyny ýükläniňizden soň, gurmak üçin exe faýlyna iki gezek basyň.

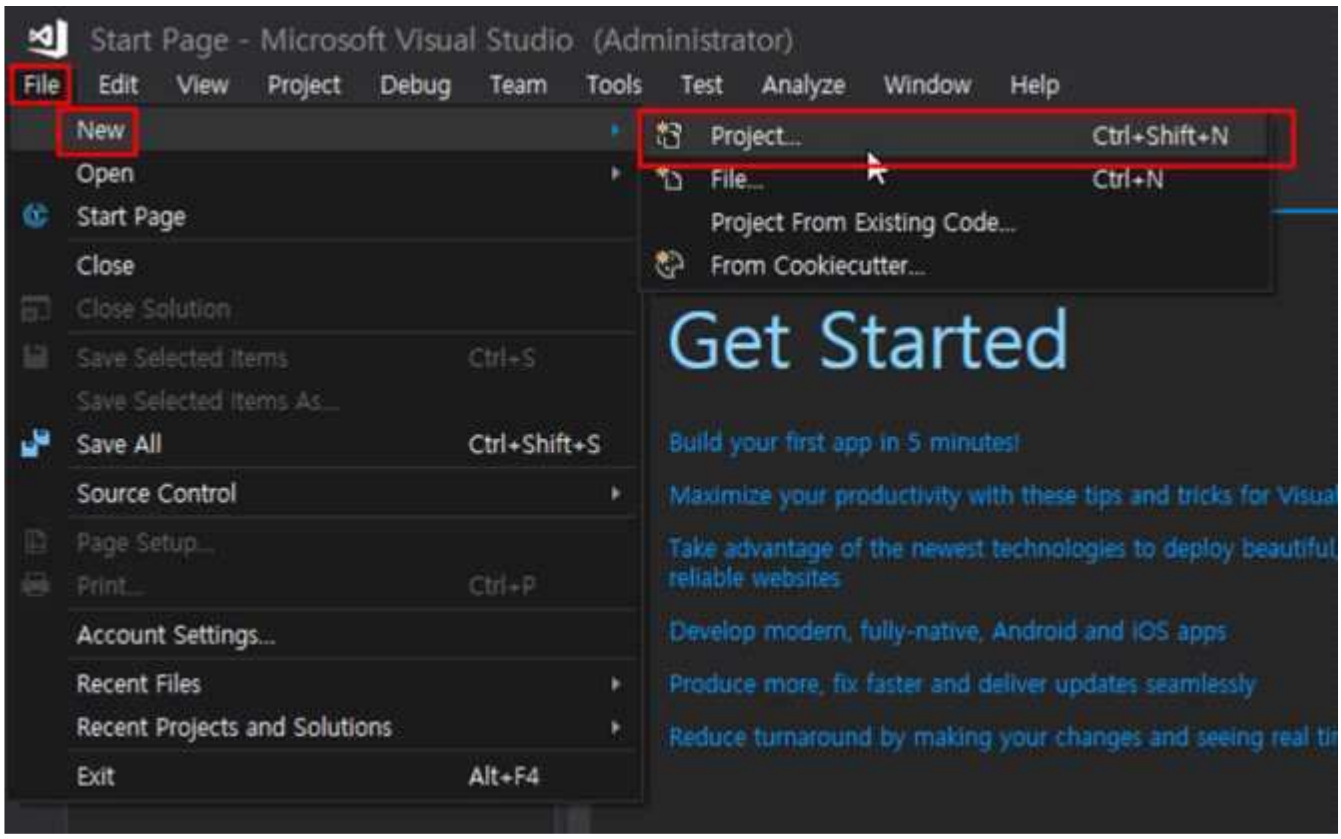
2.2.4 Windows ulgamynyň daşky gurşaw ýoluna goşmak

- OpenCV üçin gurnama ýerini windows ulgamynyň daşky gurşaw ýoluna ýazdyryň.
- C diskini aşakdaky ýaly gurnan bolsaňyz, aşakdakylary “Path”-a goşuň.
- C:\OpenCV\build\x64\vc15\bin

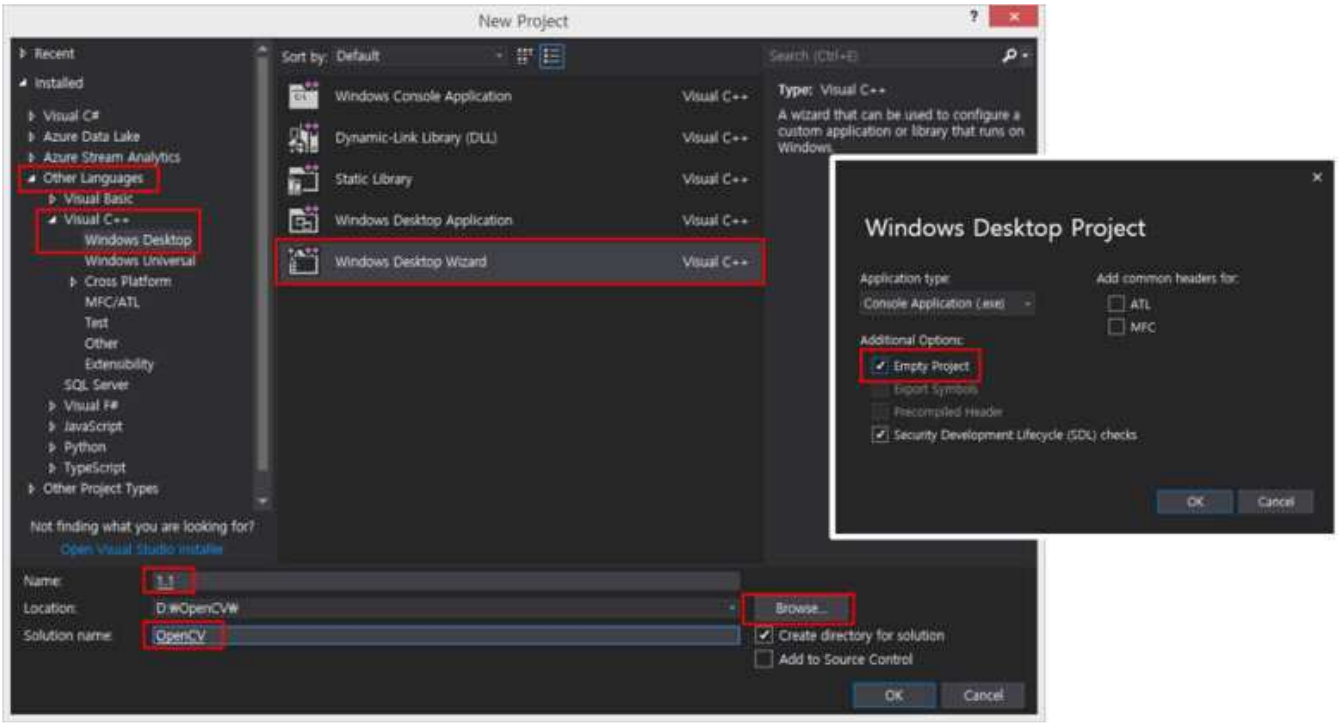


2.3 Visual Studio 2017 açyk rezýumesini düzmek

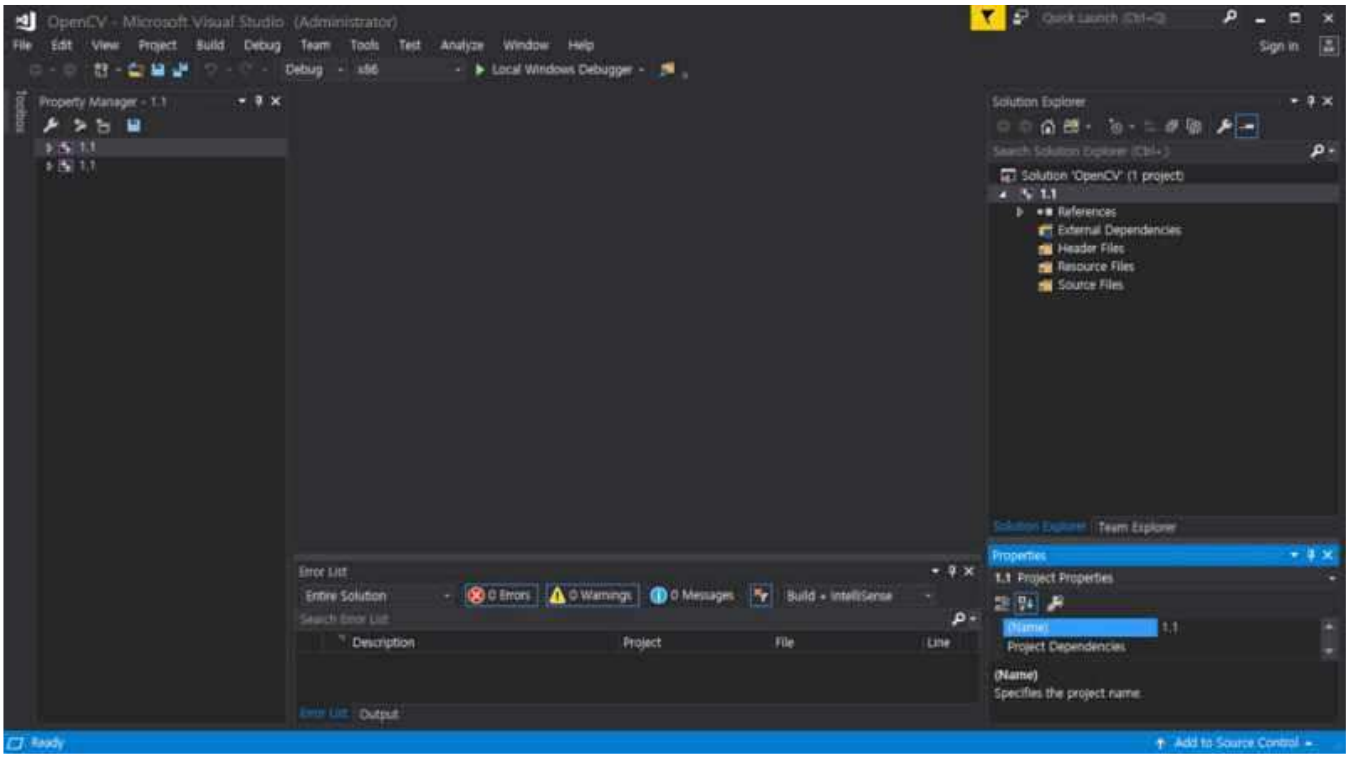
(1)



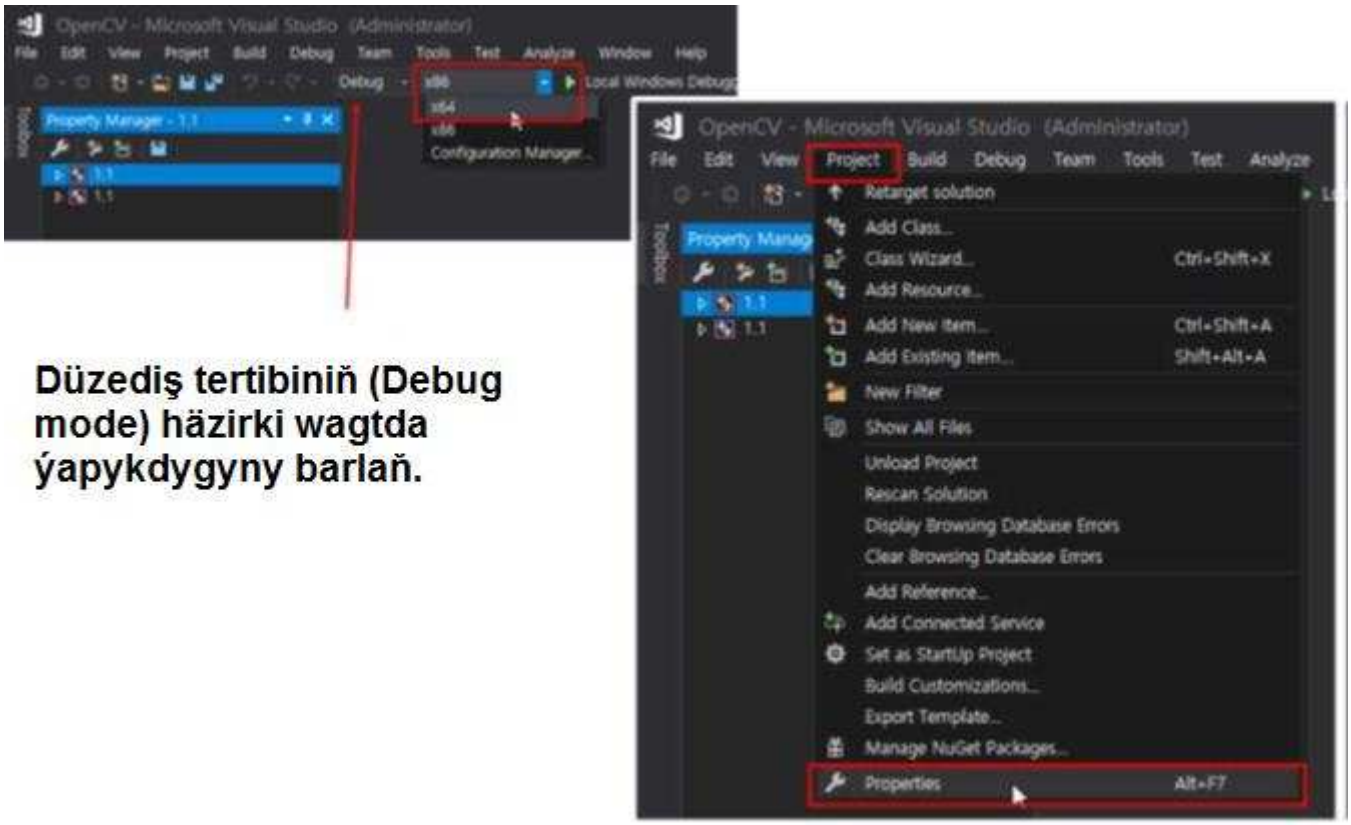
(2)



(3)

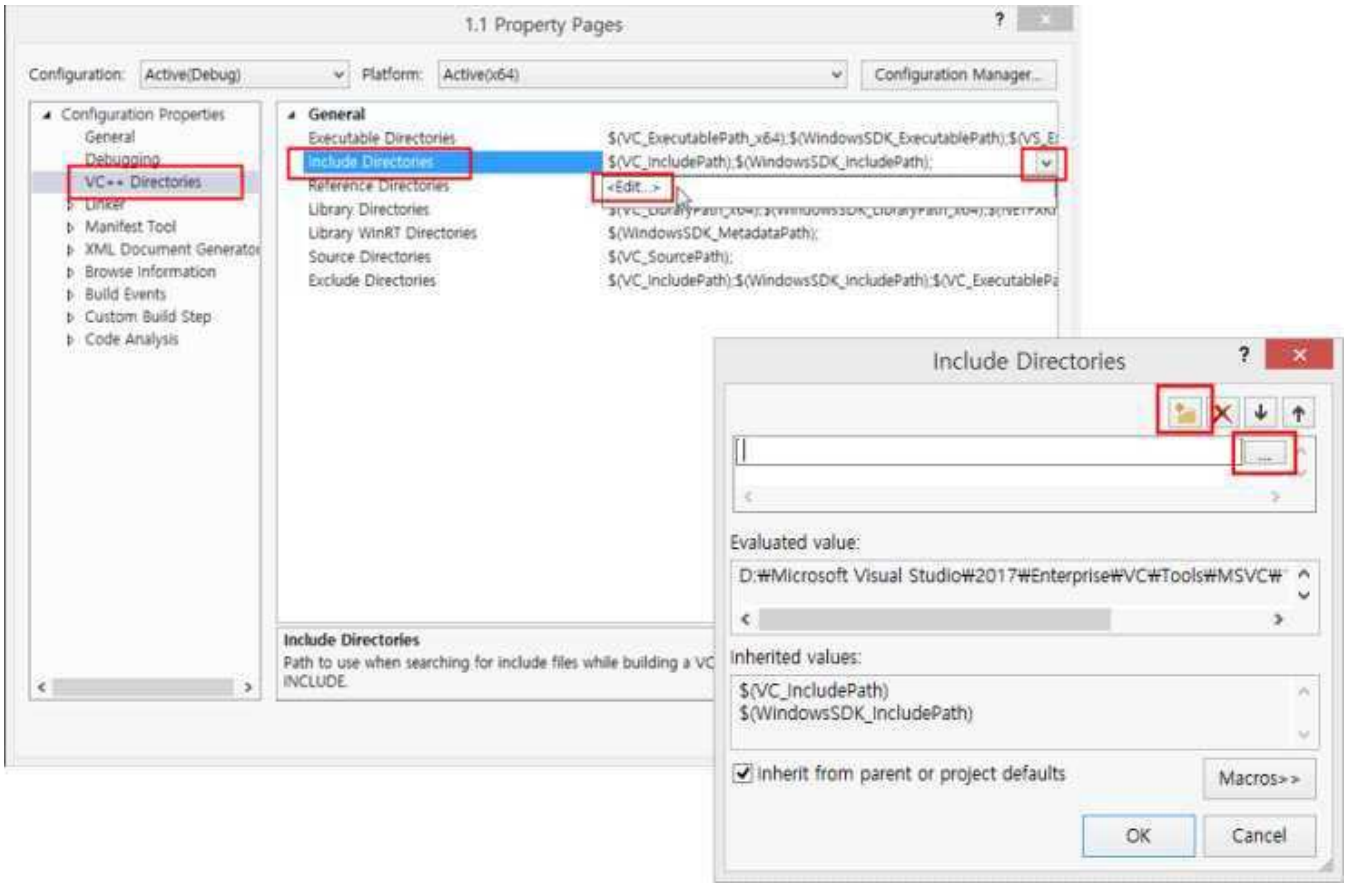


(4)

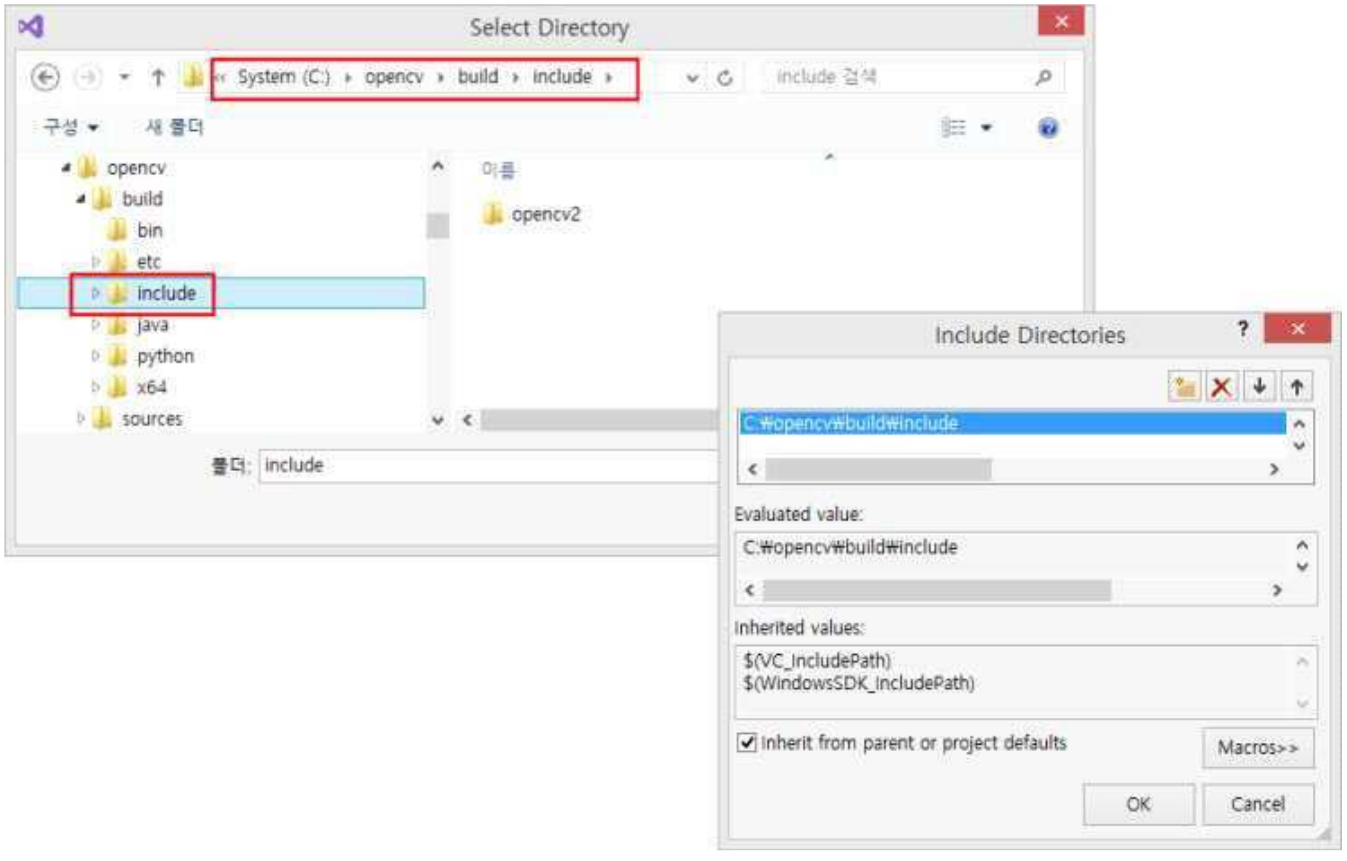


Düzediş tertibiniň (Debug mode) häzirki wagtda ýapykdygyny barlaň.

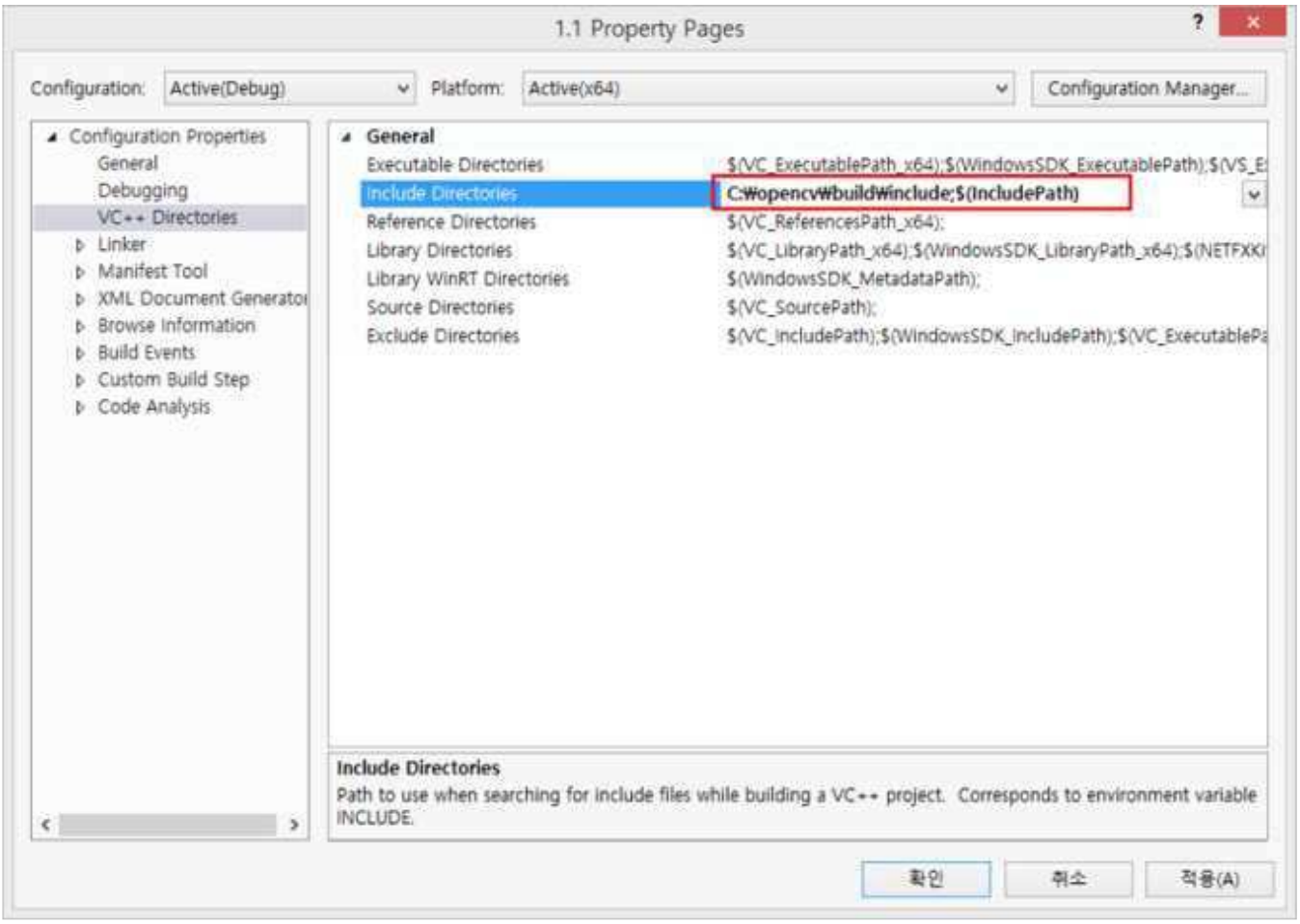
(5)


1.1 Property Pages
Configuration: Active(Debug)
Platform: Active(x64)
Configuration Manager...
Configuration Properties
General
Debugging
VC++ Directories
Linker
Manifest Tool
XML Document Generator
Browse Information
Build Events
Custom Build Step
Code Analysis
General
Executable Directories
Include Directories
Reference Directories
Library Directories
Library WinRT Directories
Source Directories
Exclude Directories
$(VC_ExecutablePath_x64);$(WindowsSDK_ExecutablePath);
$(VC_IncludePath);$(WindowsSDK_IncludePath);
<Edit...>
$(WindowsSDK_MetadataPath);
$(VC_SourcePath);
$(VC_IncludePath);$(WindowsSDK_IncludePath);
Include Directories
Path to use when searching for include files while building a VC
INCLUDE.
Include Directories
Evaluated value:
D:#Microsoft Visual Studio#2017#Enterprise#VC#Tools#MSVC#
Inherited values:
$(VC_IncludePath)
$(WindowsSDK_IncludePath)
Inherit from parent or project defaults
Macros>>
OK
Cancel

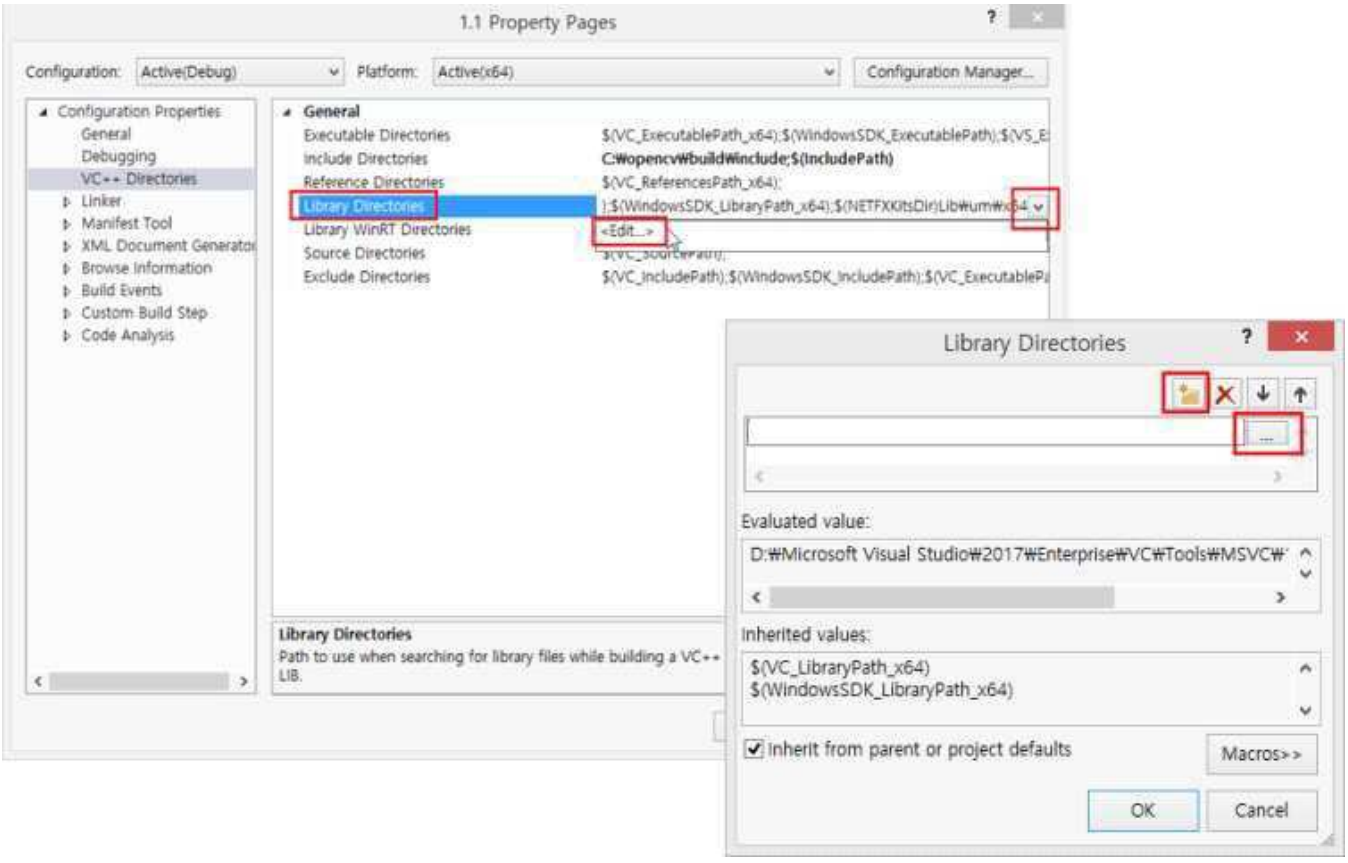
(6)


Select Directory
System (C:) ▸ opencv ▸ build ▸ include ▸
include 검색
구성
새 폴더
opencv
build
bin
etc
include
java
python
x64
sources
이름
opencv2
폴더: include
Include Directories
C:#opencv#build#include
Evaluated value:
C:#opencv#build#include
Inherited values:
$(VC_IncludePath)
$(WindowsSDK_IncludePath)
Inherit from parent or project defaults
Macros>>
OK
Cancel

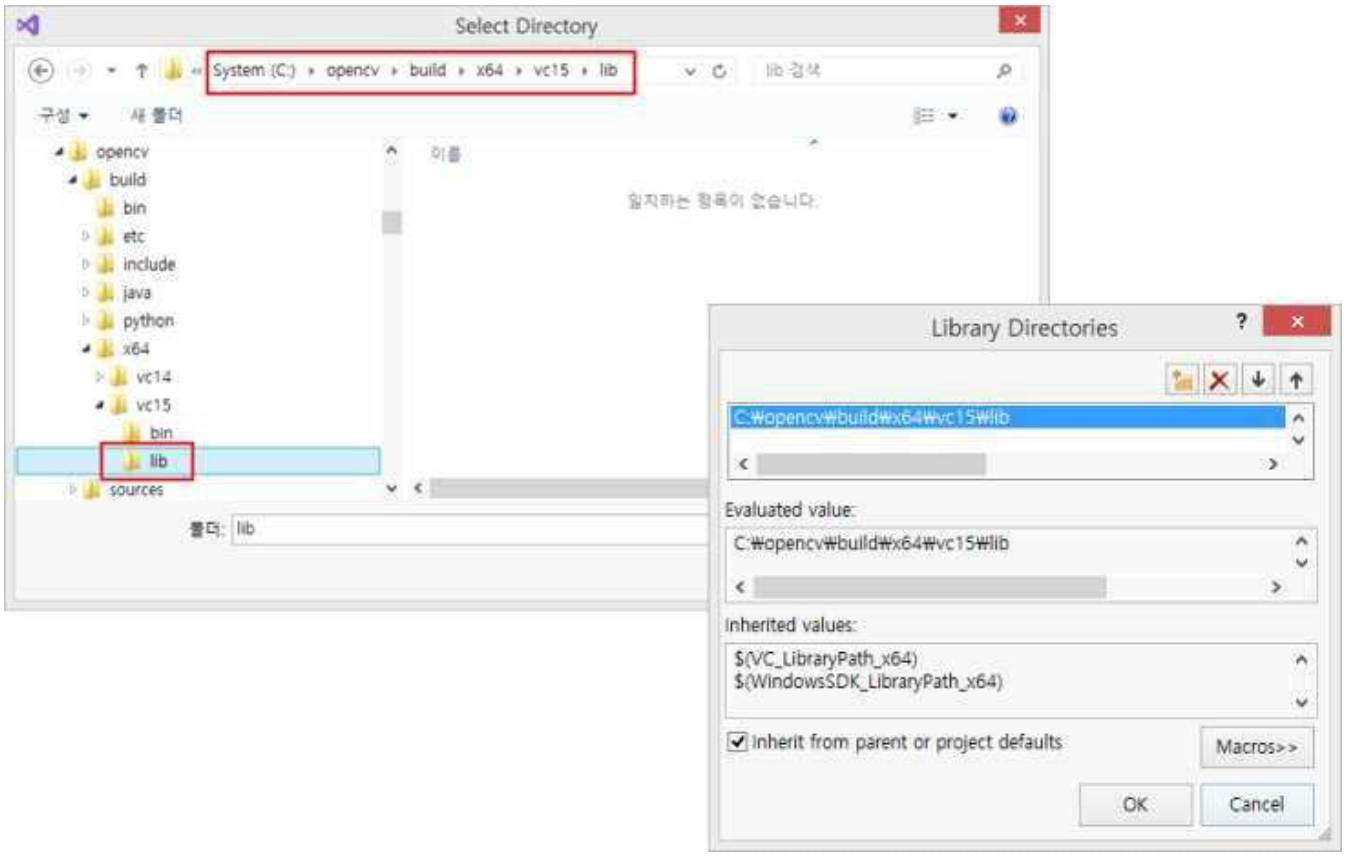
(7)


1.1 Property Pages
Configuration: Active(Debug)
Platform: Active(x64)
Configuration Manager...
Configuration Properties
General
Debugging
VC++ Directories
Linker
Manifest Tool
XML Document Generator
Browse Information
Build Events
Custom Build Step
Code Analysis
General
Executable Directories
Include Directories
C:₩opencv₩build₩include;$(IncludePath)
Reference Directories
Library Directories
Library WinRT Directories
Source Directories
Exclude Directories
Include Directories
Path to use when searching for include files while building a VC++ project. Corresponds to environment variable INCLUDE.
확인
취소
적용(A)

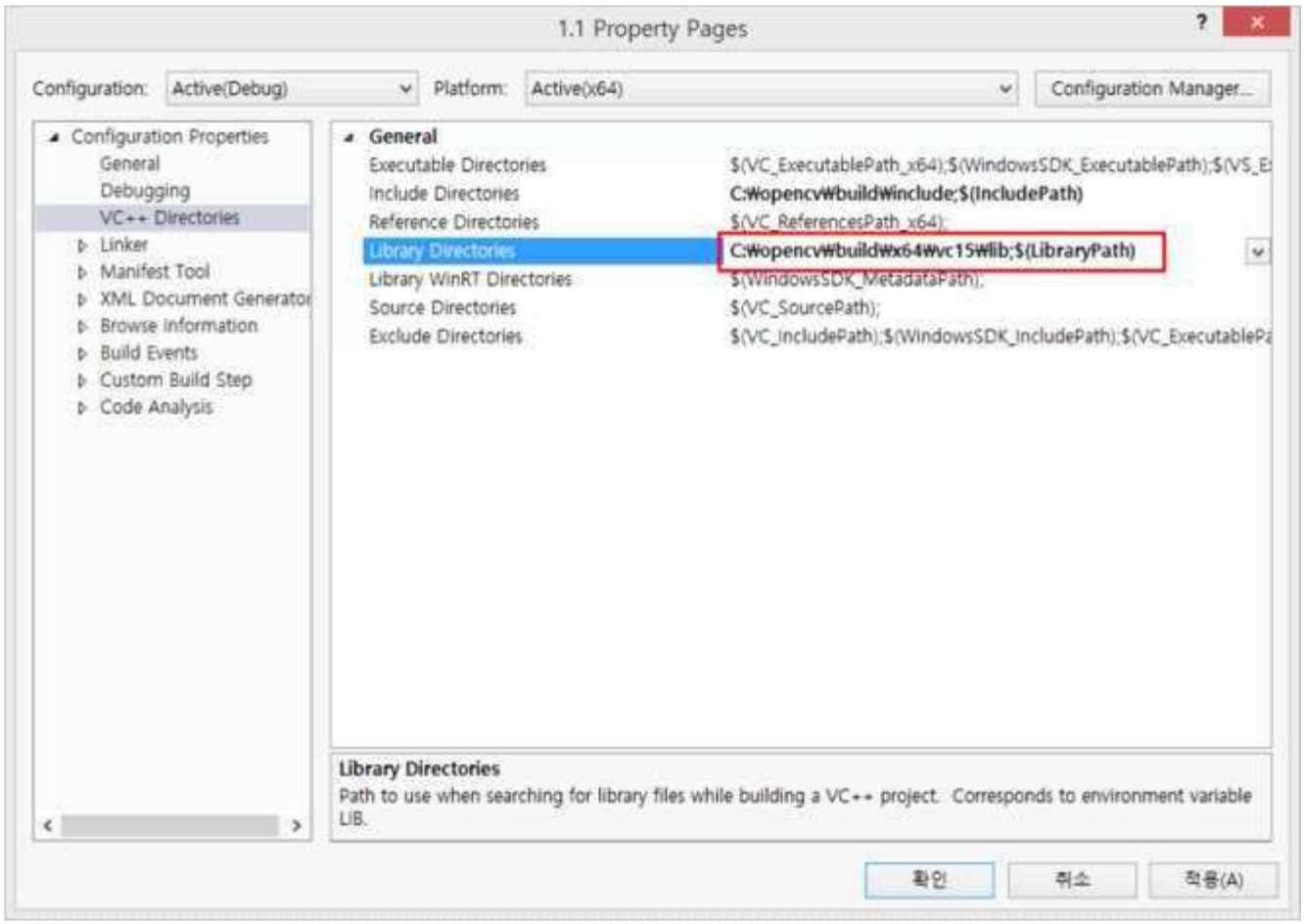
(8)


1.1 Property Pages
Configuration: Active(Debug)
Platform: Active(x64)
Configuration Manager...
Configuration Properties
General
Debugging
VC++ Directories
Linker
Manifest Tool
XML Document Generator
Browse Information
Build Events
Custom Build Step
Code Analysis
General
Executable Directories
Include Directories
C:₩opencv₩build₩include;$(IncludePath)
Reference Directories
Library Directories
Library WinRT Directories
<Edit...>
Source Directories
Exclude Directories
Library Directories
Path to use when searching for library files while building a VC++ LIB.
Library Directories
Evaluated value:
Inherited values:
$(VC_LibraryPath_x64)
$(WindowsSDK_LibraryPath_x64)
Inherit from parent or project defaults
Macros>>
OK
Cancel

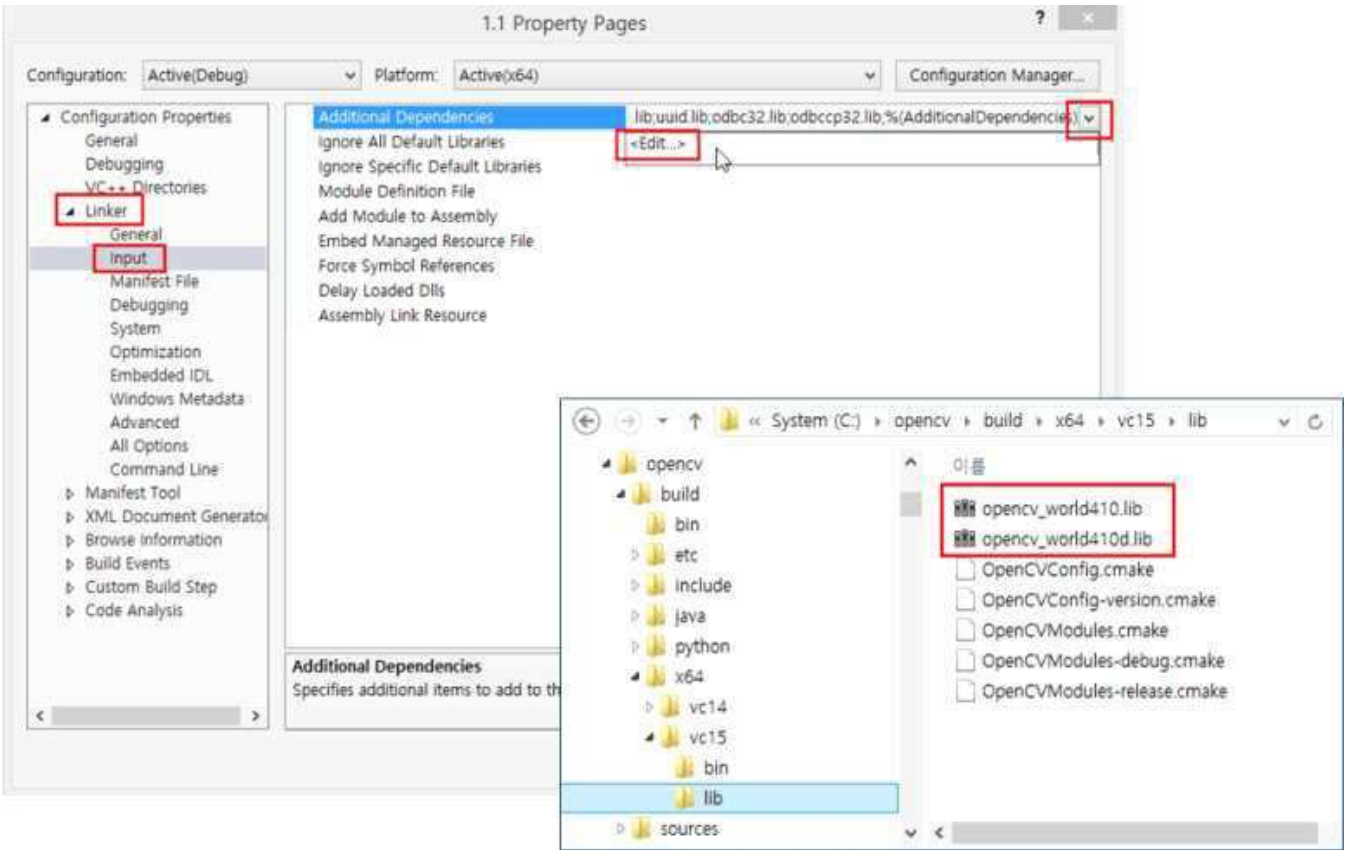
(9)


Select Directory
System (C:) ▸ opencv ▸ build ▸ x64 ▸ vc15 ▸ lib
opencv
build
bin
etc
include
java
python
x64
vc14
vc15
bin
lib
sources
폴더: lib
Library Directories
C:₩opencv₩build₩x64₩vc15₩lib
Evaluated value:
C:₩opencv₩build₩x64₩vc15₩lib
Inherited values:
$(VC_LibraryPath_x64)
$(WindowsSDK_LibraryPath_x64)
Inherit from parent or project defaults
Macros>>
OK
Cancel

(10)

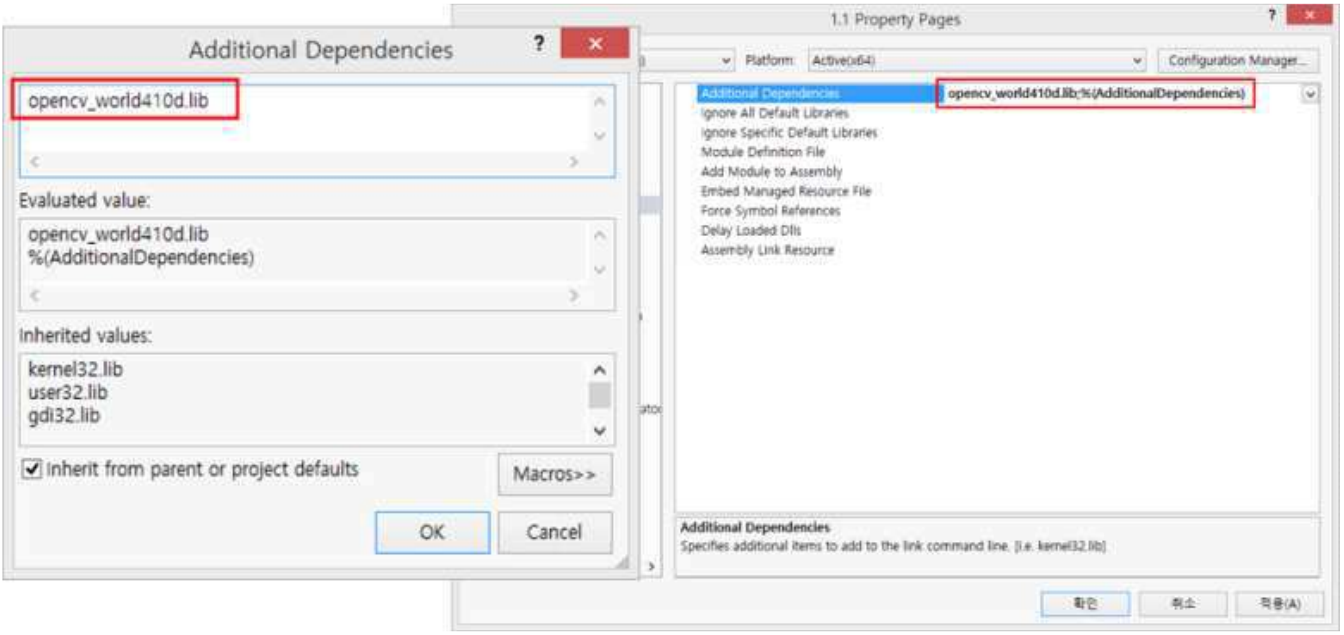

1.1 Property Pages
Configuration: Active(Debug)
Platform: Active(x64)
Configuration Manager...
Configuration Properties
General
Debugging
VC++ Directories
Linker
Manifest Tool
XML Document Generator
Browse Information
Build Events
Custom Build Step
Code Analysis
General
Executable Directories
Include Directories
C:₩opencv₩build₩include;$(IncludePath)
Reference Directories
Library Directories
C:₩opencv₩build₩x64₩vc15₩lib;$(LibraryPath)
Library WinRT Directories
Source Directories
Exclude Directories
Library Directories
Path to use when searching for library files while building a VC++ project. Corresponds to environment variable LIB.
확인
취소
적용(A)

(11)



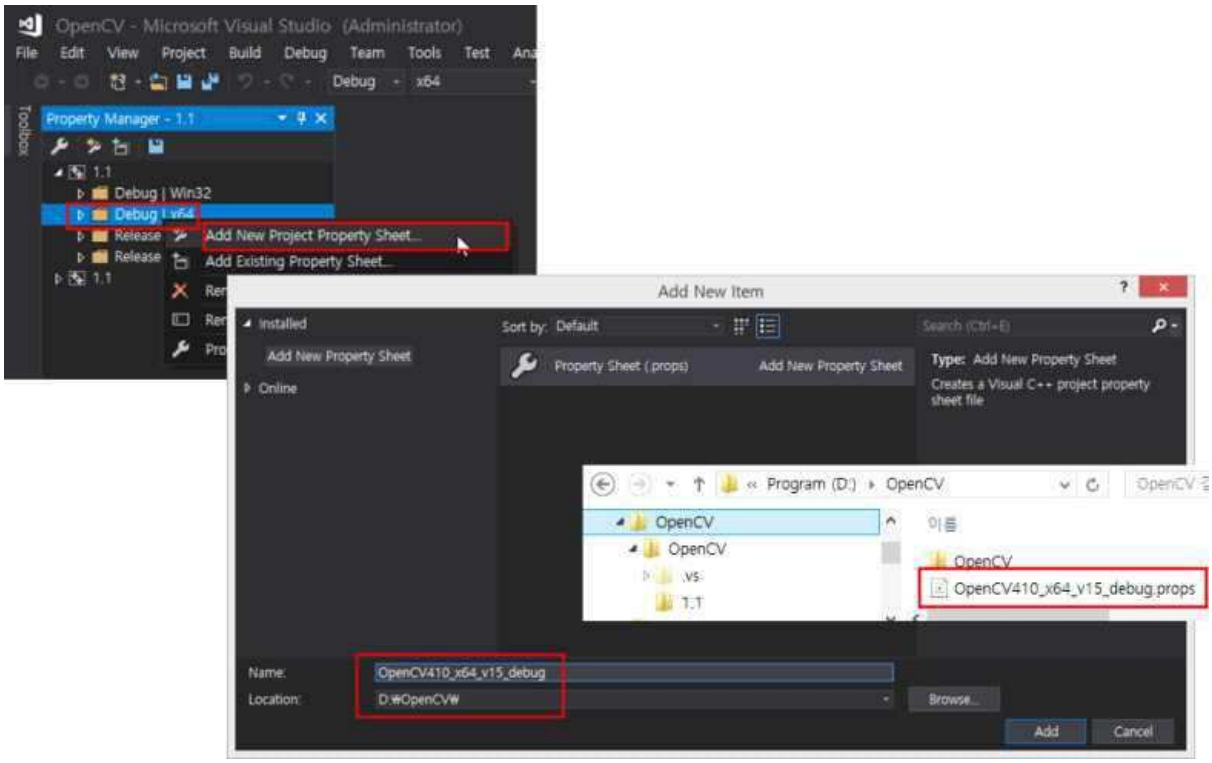
- Düzediş tertibi → pencv_worldxxxd.lib (Çykyş tertibi → opencv_worldxxx.lib)

(12)



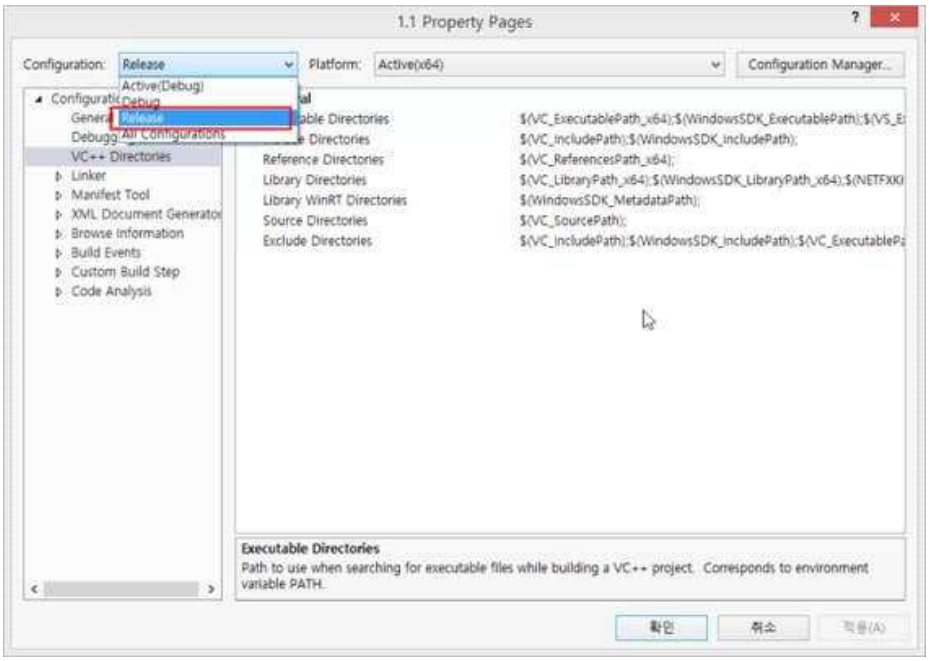
- Ady → OpenCV410_x64_v15_debug
- Ýerleşişi → OpenCV programma saklaýyş bukjasy (ýatda saklaň!!!)

(13)



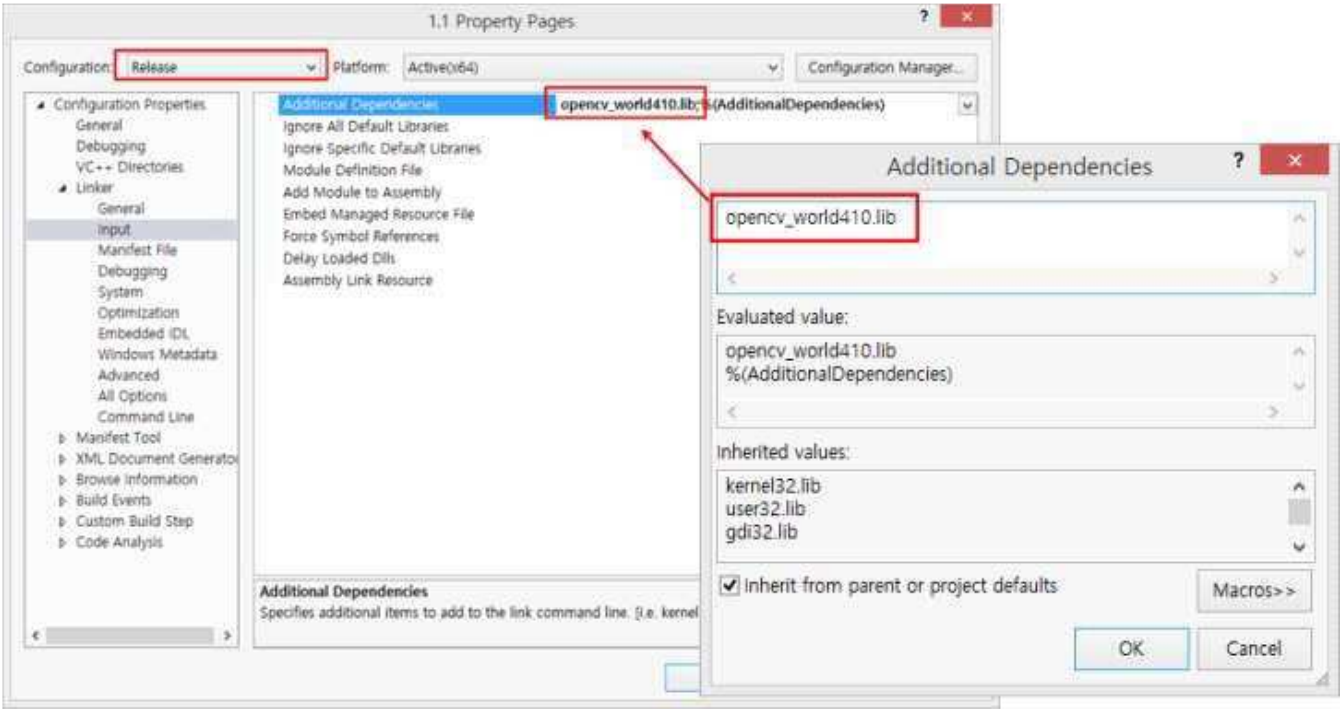
- Visual Studio-ny çykaryş tertibinde ulanylanda (çykaryş tertibi)
- Taslamany işlediň> Esasy menýudaky aýratynlyklar

(14)

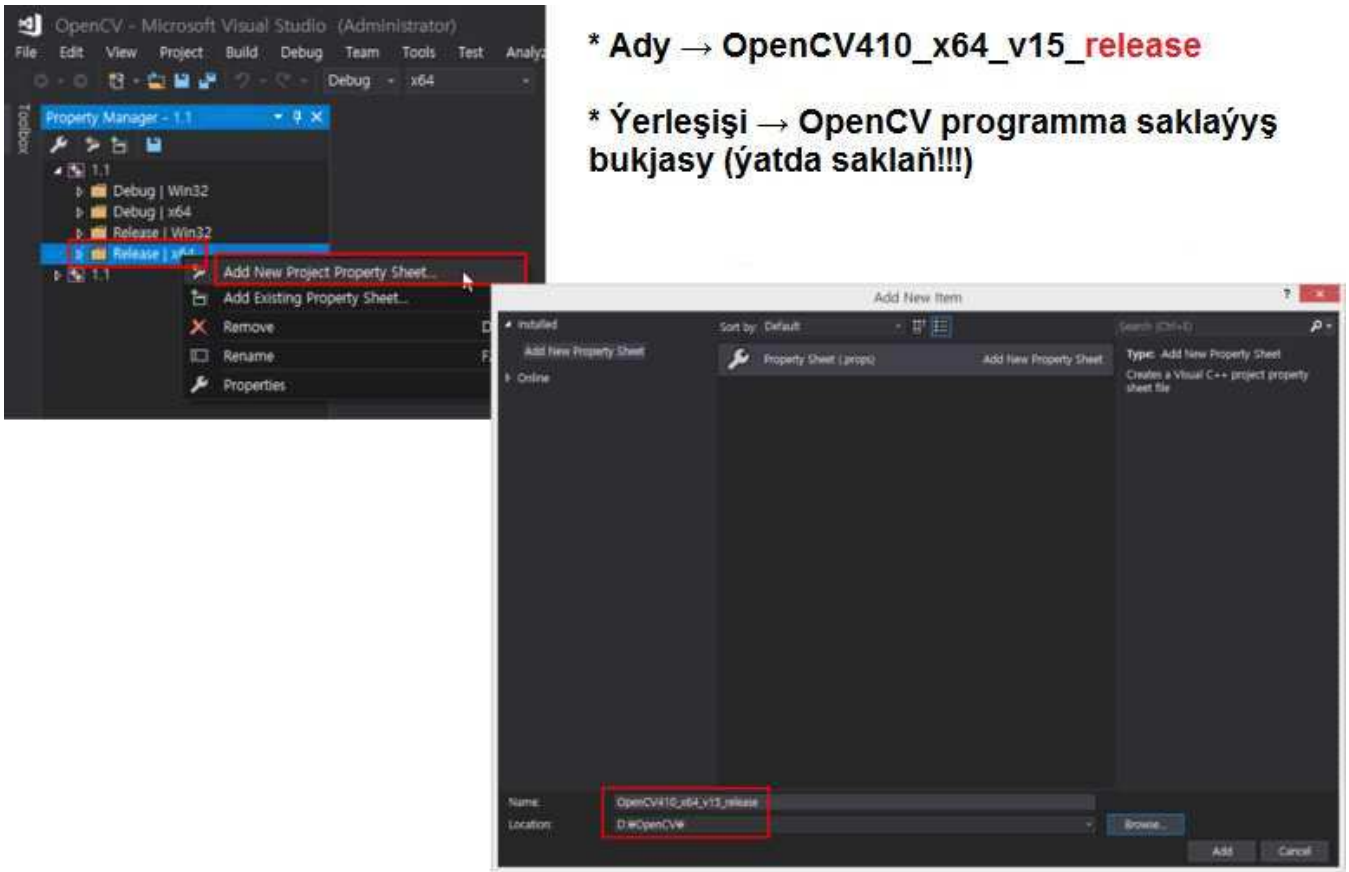


- 24-30-nji sahypalardaky şol bir mazmuny yzarlaň
- Çykyş tertibi → opencv_worldxxx.lib

(15)

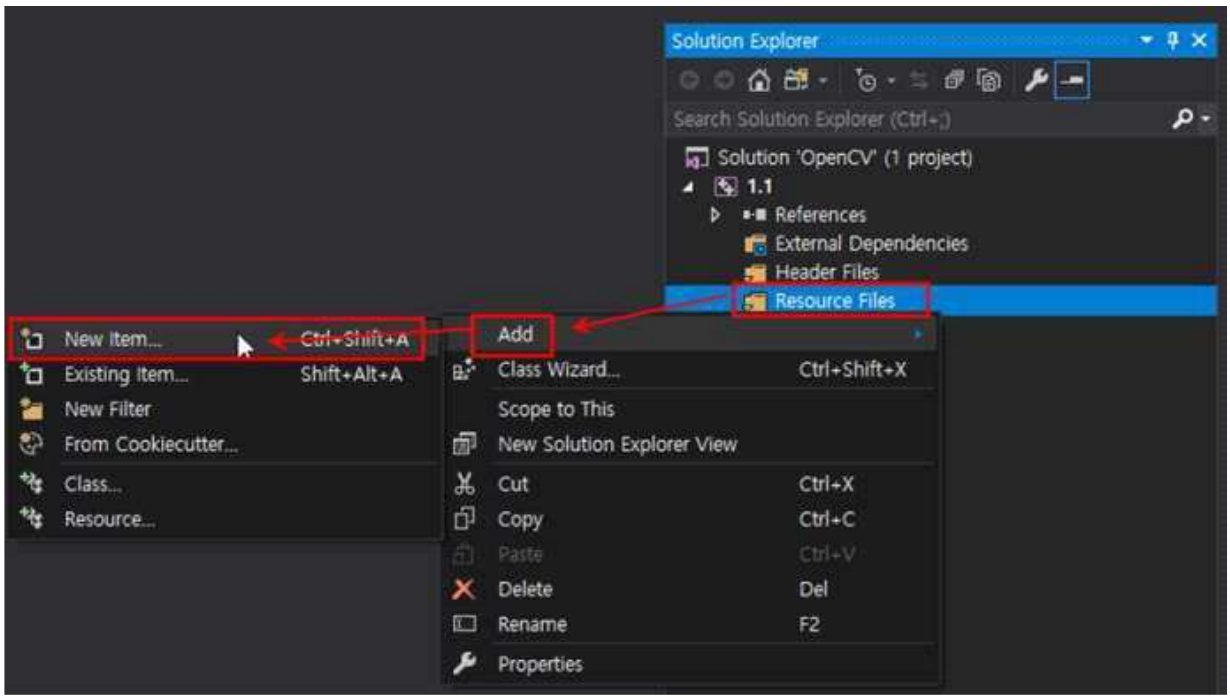


(16)



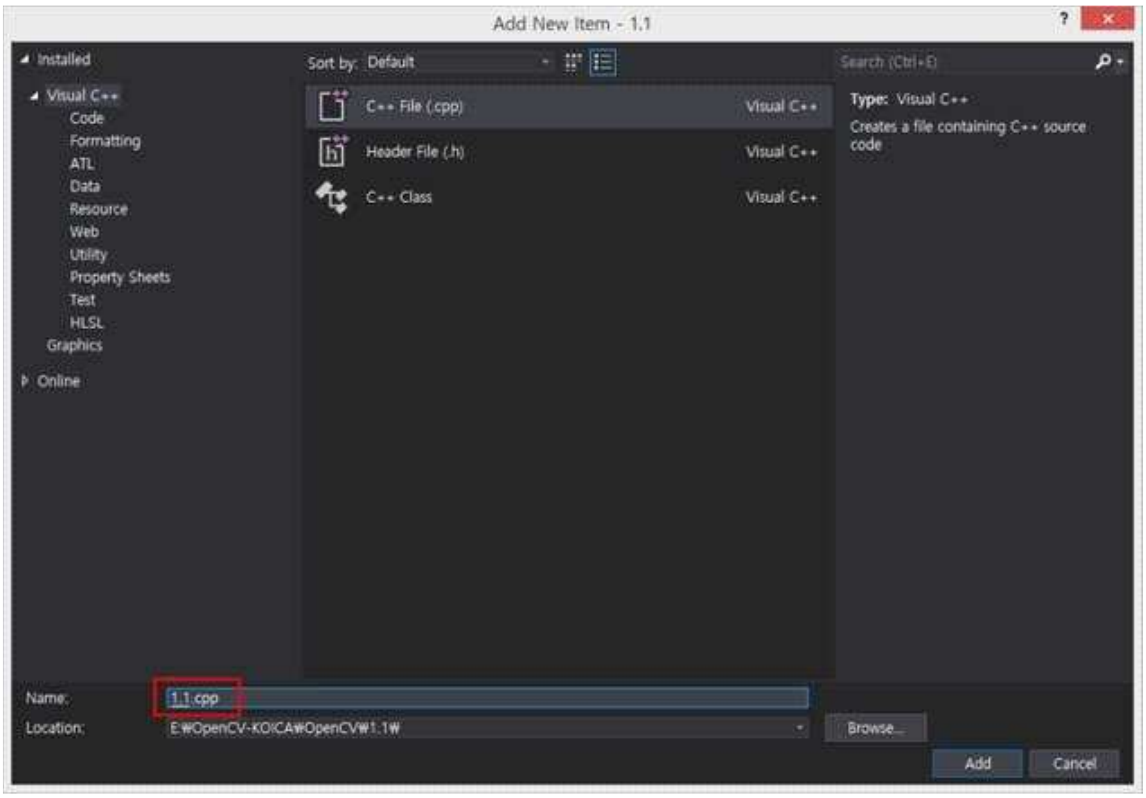
2.4 OpenCV programmasynda mysal ýazmak

(1)

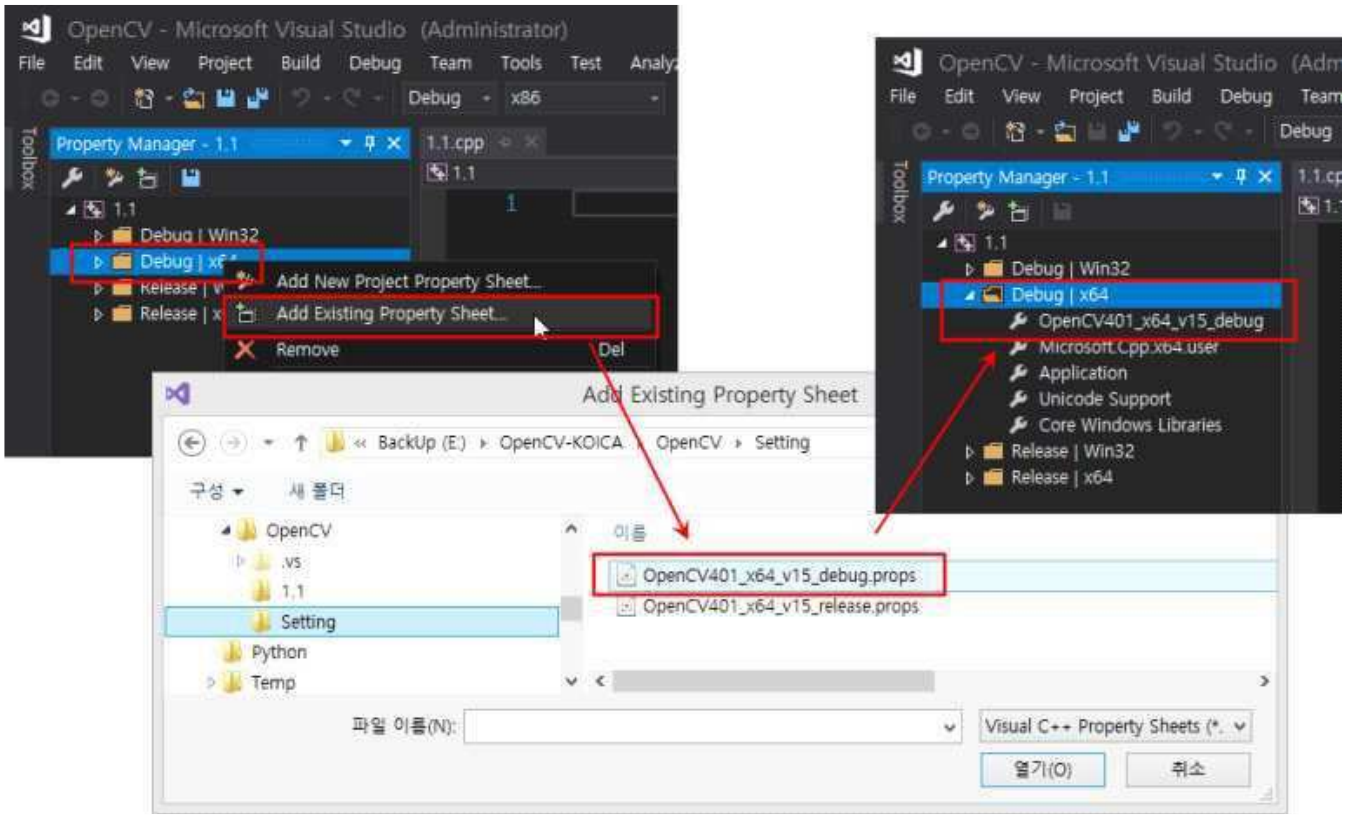


- **C ++ programmanyň ady ýazylýar ... → ???. Cpp**

(2)

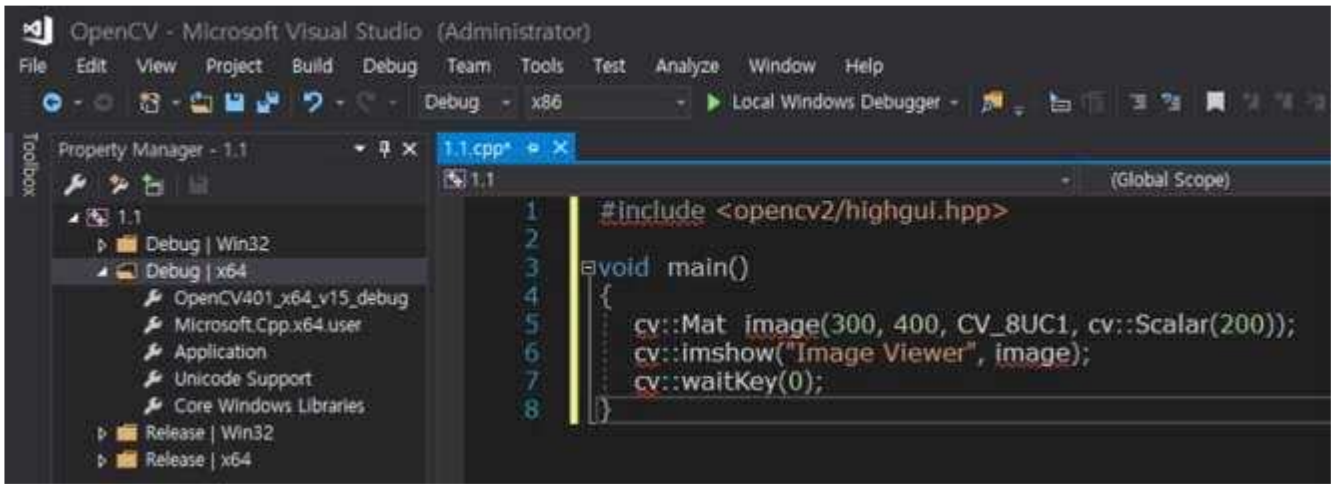


(3)



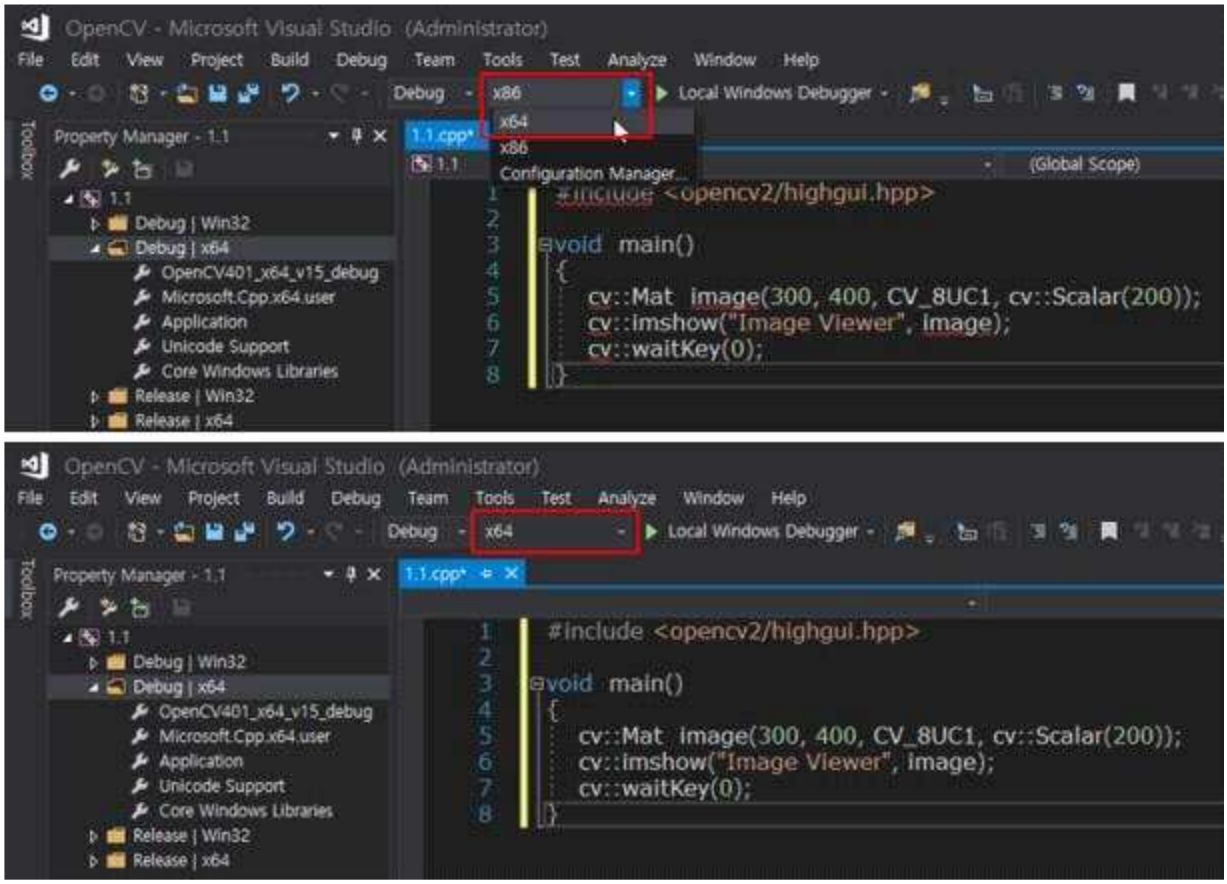

```
#include <opencv2/highgui.hpp>
void main()
{
	cv::Mat image(300, 400, CV_8UC1, cv::Scalar(200));
	cv::imshow("Image Viewer", image);
	cv::waitKey(0);
}
```

(4)



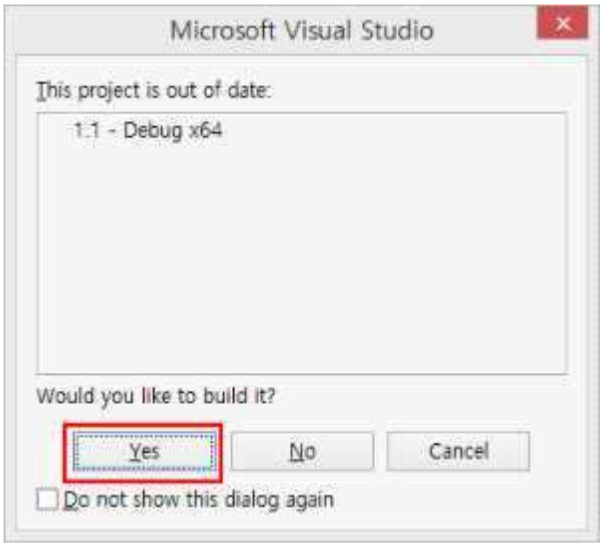
- Ýalňyşlyk → x64 gurnamasy!!

(5)



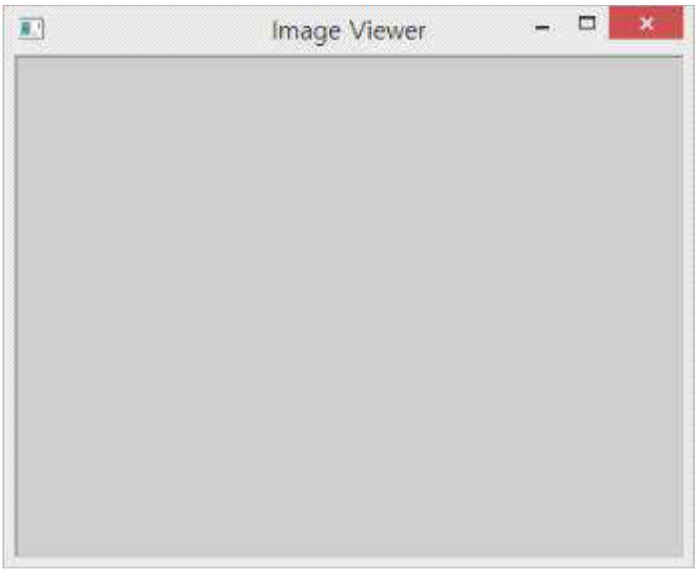
- Ctrl + F5: Ýalňyşlygy ýüze çykaryşsyz başlaň

(6)



- **Çykarma (netije)**

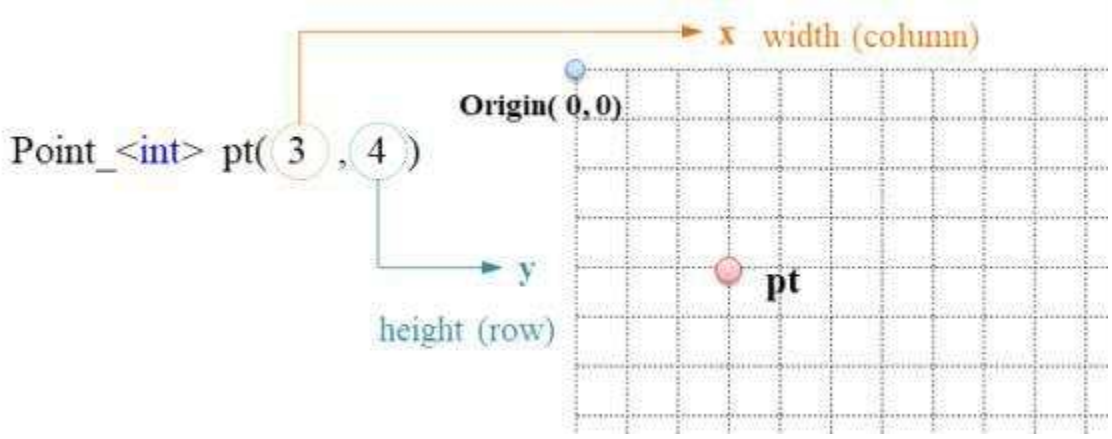
(7)



3. OpenCV toparlary

3.1 Point_topary

2D koordinatasyndaky ini we beýiklik pozisiýalaryny görkezýän şablon topary.



- **Point_ toparynyň beýany**

Point_<int> <==> Point2i <==> Point;

Point_<float> <==> Point2f;

Point_<double> <==> Point2d;

- **Mysal**

Point_<int> pt1(100, 200);

Point_<float> pt2(92.3f, 125.23f);

Point_<double> pt3(100.2, 300.9);

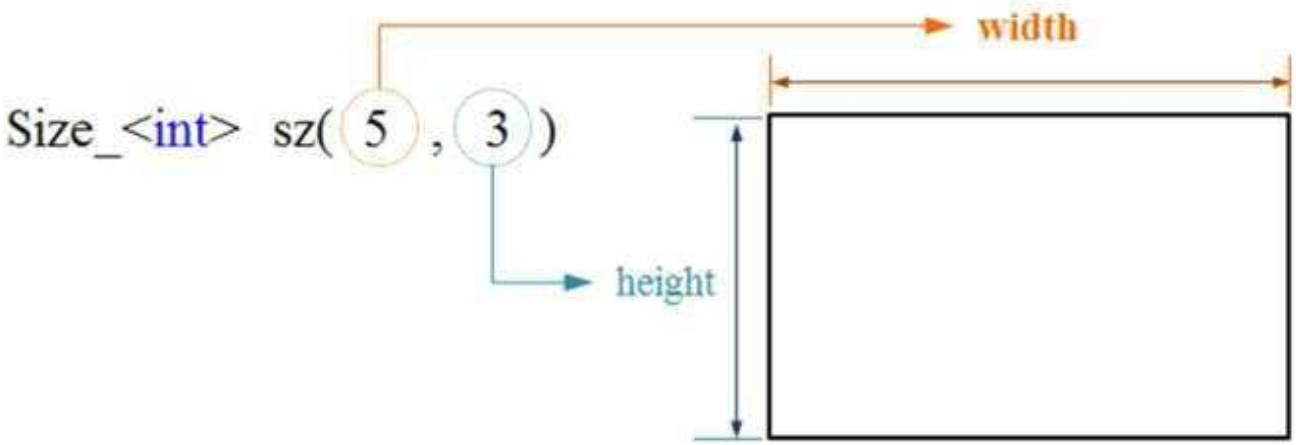
Point pt4(120, 69);

Point2f pt5(0.3f, 0.f), pt6(0.f, 0.4f);

Point2d pt7(0.25, 0.6);

3.2 Size_ topary

- Suratyň ýa-da gönüburçlugyň ululygyny kesgitleýän şablon topary.



- **Size_ toparynyň beýany**

Size_<int> <==> Size2i <==> Size;

Size_<float> <==> Size2f;

Size_<double> <==> Size2d;

- **Mysal**

Size_<int> sz1(100, 200);

Size_<float> sz2(192.3f, 25.3f);

Size_<double> sz3(100.2, 30.9);

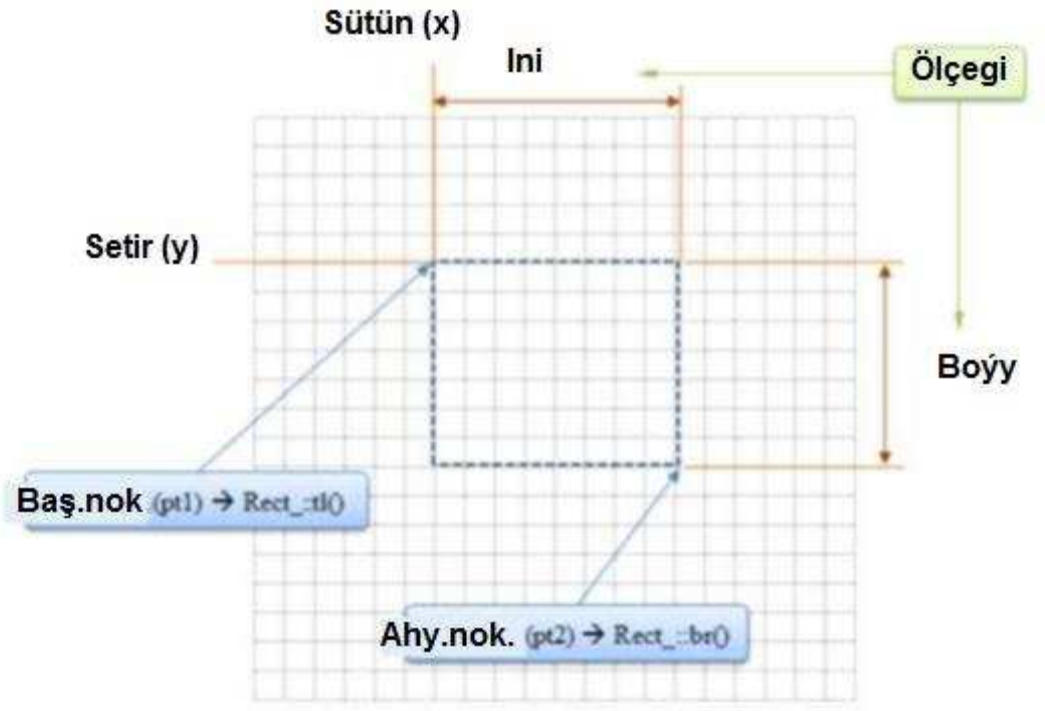
Size sz4(120, 69);

Size2f sz5(0.3f, 0.f);

Size2d sz6(0.25, 0.6);

3.3 Rect_ topary

- Gönüburçlugy görkezmek üçin şablon topary
- ((Başlangyç nokady_x, Başlangyç nokady_y), (Ahyrky nokady_x, Ahyrky nokady_y))
- ((Başlangyç nokady_x, Başlangyç nokady_y), ini, beýikligi)



- **Rect_ toparynyň beýany**

Size_<int> <==> Size2i <==> Size;

Size_<float> <==> Size2f;

Size_<double> <==> Size2d;

- **Mysal**

Size2d sz(100.5, 60.6);

Point2f pt1(20.f, 30.f), pt2(100.f, 200.f);

Rect_<int> rect1(10, 10, 30, 50); // column, row, width, height

Rect_<float> rect2(pt1, pt2);

Rect_<double> rect3(Point2d(20.5, 10), sz);

3.4 Vec topary

- Birnäçe elementli wektor belgisi üçin şablon topary.

- Maglumatlaryň görnüşini we <and> arasyndaky elementleriň sanyny görkeziň.

Vec<uchar, 2> <==> Vec2b

Vec<int, 3> <==> Vec3i

Vec<float, 4> <==> Vec4f

Vec<double, 5> <==> Vec5d

- **Mysal**

```
Vec<int, 2> v1(5, 12);
Vec<double, 3> v2(40, 130.7, 125.6);
Vec2b v3(10, 10);
Vec6f v4(40.f, 230.25f, 525.6f);
Vec3i v5(200, 230, 250);
```

3.5 Scalar_ topary

Bir pikseliň ýagtylyk derejesini kesgitlemek üçin dört maglumat görnüşiniň derejesini kesgitläň.
Dört derejäni Gök, ýaşyl, gyzyl, alfa (dury) ýaly saklaň.
Başladylan mahaly hiç hili dereje görkezilmedik bolsa 0-a düzüň. Scalar_ <double> <==> Skaler

- Mysal

Scalar_<uchar> red(0, 0, 255);
Scalar_<int> blue(255, 0, 0);
Scalar_<double> color1(500);
Scalar_<float> color2(100.f, 200.f, 125.9f);

3.6 Mat topary

Suraty kesgitlemek üçin ulanylýan topar.

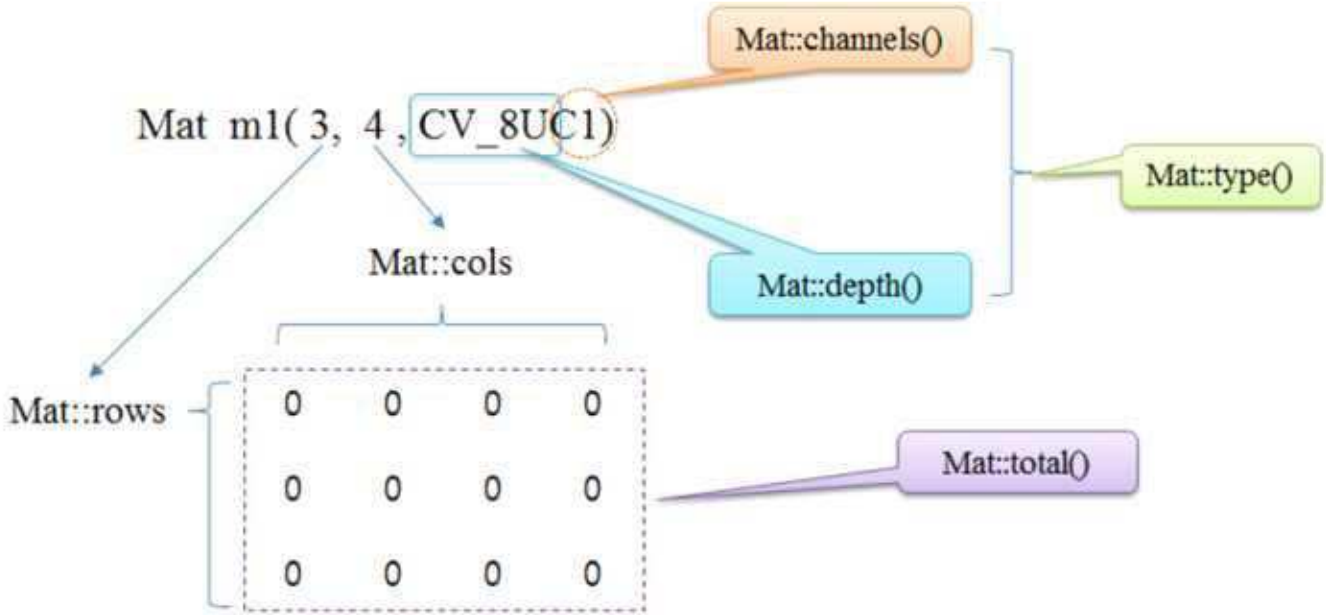
- Mat (rows, cols, type, Scalar)

rows: setiriň ululygy
cols: sütüniň ululygy
type: maglumat görnüşi
Scalar: matrisanyň bahasy

Maglumat görnüşi	Düşündirilişi	Çuňlugy
CV_8U	uchar(unsigned char)	0
CV_8S	signed char	1
CV_16U	unsigned short int	2
CV_16S	signed short int	3
CV_32S	int	4
CV_32F	float	5
CV_64F	double	6

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;
int main()
{
	float data[] = {
		1.2f, 2.3f, 3.2f,
		4.5f, 5.f, 6.5f,
	};
	Mat m1(2, 3, CV_8U);
	Mat m2(2, 3, CV_8U, Scalar(300));
	Mat m3(2, 3, CV_32F, data);
	Size sz(2, 3);
	Mat m4(Size(2, 3), CV_64F);
	Mat m5(sz, CV_32F, data);
	cout << "[m1] =" << endl << m1 << endl;
	cout << "[m2] =" << endl << m2 << endl;
	cout << "[m3] =" << endl << m3 << endl << endl;
	cout << "[m4] =" << endl << m4 << endl;
	cout << "[m5] =" << endl << m5 << endl;
	return 0;
}
```

Agza üýtgeýjileri	Mat::dims	Ölçegleriň sany
	Mat:: rows	Setirleriň sany
	Mat::cols	Sütünleriň sany
Agza usullary	Mat::channels()	Matrisadaky kanallaryň sanyny görkezýär.
	Mat::depth()	Matrisada maglumat görnüşiniň derejesiniň yzyna gaýtarylmasy
	Mat::empty()	Yzyna gaýtarylma (Return) boş matrisa elementidir
	Mat::size()	Matrisanyň ululygynyň "Ölçeg" görnüşi hökmünde yzyna gaýtarylmasy
	Mat::total()	Jemi yzyna gaýtarylan matrisa elementleriniň sany
	Mat::resize(sz, s) sz: üýtgemeli setriň sany s: goşmak üçin setir elementiniň derejesi	Bar bolan matrisa setirine görä üýtgediň. Sz bar bolan matrisadaky setirleriň sanyndan az bolsa, aşaky setiri aýyryň we has köp bolsa, bar bolan matrisanyň aşagyna setir goşuň.
	Mat::reshape(cn, rows) cn: üýtgemeli kanallaryň sany rows: üýtgemeli setirleriň sany	Jemi elementleriň sanyny üýtgetmän matrisany üýtgediň. Asyl matrisanyň we üýtgedilen matrisanyň jemi element sany deň gelmeýän bolsa ýalňyşlyk ýüze çykýar.



```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	Mat m1(4, 3, CV_32FC3);

	cout << “Dimension = " << m1.dims << endl;
	cout << “Rows = " << m1.rows << endl;
	cout << “Columns = " << m1.cols << endl << endl;

	cout << “Channels = " << m1.channels() << endl;
	cout << “Data Type = " << m1.depth() << endl;
	cout << “Matrix Size = " << m1.size() << endl << endl;
	cout << “Total Data Number = " << m1.total() << endl;
	return 0;
}
```

Mysal	**Düşündirilişi**
m1=100	Sag tarapdaky ähli matrisa derejelerini “=” -e üýtgediň.
m1=m2	m1, m1-e göçürilmeýär, emma m1 matrisa m2 matrisany bölýär.
m1=m2+m3	Sag tarapdaky martisa goşulmasynyň netijesi m1 matrisasyna göçürilýär.

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	Mat m1(2, 3, CV_8U, 2);
	Mat m2(2, 3, CV_8U, Scalar(10));

	Mat m3 = m1 + m2;
	Mat m4 = m2 - 6;
	Mat m5 = m1;

	cout << "[m2] =" << endl << m2 << endl;
	cout << "[m3] =" << endl << m3 << endl;
	cout << "[m4] =" << endl << m4 << endl << endl;

	cout << "[m1] =" << endl << m1 << endl;
	cout << "[m5] =" << endl << m5 << endl << endl;
	m5 = 100;
	cout << "[m1] =" << endl << m1 << endl;
	cout << "[m5] =" << endl << m5 << endl;
	return 0;
}
```

- Asyl matrisany başga bir matrisa göçüriň
- Mat clone()
- void copyTo (maksat matrisasy, maska matrisasy)
- mask matrix: Diňe nol däl elementleri göçüriň
- void convertTo (obýektiw matrisasy, maglumat görnüşi)
- data type: üýtgetmek isleýän maglumat görnüşiňiz

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	double data[] = {
		1.1, 2.2, 3.3, 4.4,
		5.5, 6.6, 7.7, 8.9,
		9.9, 10, 11, 12
	};
	Mat m1(3, 4, CV_64F, data);
	Mat m2 = m1.clone();		// m1-den m2-e göçürmek

	Mat m3, m4;
	m1.copyTo(m3);			// m1-den me-e göçürmek
	m1.convertTo(m4, CV_8U); // m1-den m4-e nusgasyny uchar-a öwürýär

	cout << "[m1] =\n" << m1 << endl;
	cout << "[m2] =\n" << m2 << endl;
	cout << "[m3] =\n" << m3 << endl;
	cout << "[m4] =\n" << m4 << endl;
	return 0;
}
```

3.7 Vector topary

Yzygiderlilik konteýneri C ++ STL (Standart şablon kitaphanasy)

- Wektoryň elementine girmegi: inteks operatoryny [] massiw hökmünde ulanyň

- vector(): konstruktor
- void pushback(): wektoryň soňuna bir element goşýar
- void pop_back(): soňky elementi aýyrýar

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{

	vector<Point> v1;
	v1.push_back(Point(10, 20));
	v1.push_back(Point(20, 30));
	v1.push_back(Point(50, 60));

	vector<float> v2(3, 9.25);
	Size arr_size[] = { Size(2, 2), Size(3, 3), Size(4, 4) };
	int arr_int[] = { 10, 20, 30, 40, 50 };

	vector<Size> v3(arr_size, arr_size + sizeof(arr_size) / sizeof(Size));
	vector<int> v4(arr_int + 2, arr_int + sizeof(arr_int) / sizeof(int));

	cout << "[v1] " << ((Mat)v1) << endl << endl;
	cout << "[v2] " << ((Mat)v2) << endl << endl;
	cout << "[v2] " << ((Mat)v2).reshape(1, 1) << endl;
	cout << "[v3] " << ((Mat)v3).reshape(1, 1) << endl;
	cout << "[v4] " << ((Mat)v4).reshape(1, 1) << endl;
	return 0;
}
```

3.8 Range topary

Ilki bilen Mat toparynda setirleriň we sütünleriň yzygiderliligini kesgitlemek üçin ulanylýar.

- Range (int start, int end)
- Başlangyç (Start) aralygynda, ahyrky (End) aralykda däl

3.9 Matrisa amal funksiýasy

- Matexp inv (usuly): ters matrisa hasaplama usuly

DECOMP.LU	Optimal pivot elementini saýlap, Gaussyň ýok edilmegi.
DECOMP.SVD	Ýeke-täk baha bölüniş (SVD) usuly: ulgam aşa kesgitlenip bilner we/ýa-da matrisa srcl ýeke bolup biler
DECOMP.CHOLESKY	Cholesky faktorlaşdyrma: srcl matrisa simmetrik bolmaly we pozitiw görnüşde kesgitlenmeli

Matexp mul(input matrix): Iki matrisany element boýunça köpeldiň
Matexp t(): Transpoz matrisasyny hasaplaň. Bir wagtda deňleme.

$$\begin{aligned} 1x_1 + \quad & \quad 2x_3 = 6 \\ -3x_1 + 2x_2 + & \quad 6x_3 = 30 \\ -1x_1 - 2x_2 + & \quad 3x_3 = 8 \end{aligned}$$

$$\begin{bmatrix} 1 & 0 & 2 \\ -3 & 2 & 6 \\ -1 & -2 & 3 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ 30 \\ 8 \end{bmatrix} \rightarrow \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 2 \\ -3 & 2 & 6 \\ -1 & -2 & 3 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 6 \\ 30 \\ 8 \end{bmatrix}$$

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	float data[] = {
		1, 0, 2,
		-3, 2, 6,
		-1, -2, 3
};
	float ans[] = {6, 30, 8 };

	Mat m1(3, 3, CV_32F, data);
	Mat m2(1, 3, CV_32F, ans);
	Mat m2_t = m2.t();

	Mat m1_inv = m1.inv(DECOMP_LU);
	Mat x = m1_inv * m2_t;

	cout << "[m1] = " << endl << m1 << endl;
	cout << "[m1_inv] = " << endl << m1_inv << endl << endl;
	cout << "[m2(transposed)] = " << endl << m2_t << endl <<
	endl;
	cout << “solution x1, x2, x3 = " << x.t() << endl;
}
```

3.10 saturate_cast < >

- Şekil maglumatlary (image data): esasan maglumatlary her kanalda 8 bitde kodlaýar.
- Diňe 8 bit ulanýandygy sebäpli, çäkli piksel derejeleri aralygyna eýe (0 ~ 255).
- saturate_cast () şablon usuly: Bahasy 8 bit görnüşinde saklananda, 8 bitlik aralykdan geçse, 0 ýa-da 255 görnüşinde saklanýar

- **Mysal**

```
Mat m1(2, 2, CV_8U);
m1(0, 0) = -50; // -> 206
m1(0, 1) = 300; // -> 44
m1(1, 0) = saturate_cast<uchar>(-50);
m1(1, 1) = saturate_cast<uchar>(300);
```

4. OpenCV ulanyjy interfeýsleri

4.1 Penjiräni dolandyrmak

- namedWindow(winname, flags): Penjiräniň adyny düzýär we şol at bilen penjire döredýär
- flags: penjiräniň ölçegini üýtgetýär

Görnüşi	Derejesi	Düşündirilişi
WINDOW.NORMAL	0	Penjiräniň ölçegi üýtgedilýär
WINDOW_AUTOSIZE	1	Matrisanyň ölçegine görä awtomatiki görnüşde sazlanýar. Ölçegini üýtgetmek

- imshow (): “mat” matrisasyny winname penjiresinde, bir penjire hökmünde görkezýär
- destroyWindow (): görkezilen penjiräni ekrandan aýyrýar
- destroyAllWindows (): ähli görünýän penjireleri aýyrýar
- moveWindow (x, y): winname penjiresini görkezilen ýere geçirýär (x (sütün, y (setir))

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	Mat image1(300, 400, CV_8U, Scalar(255));
	Mat image2(300, 400, CV_8U, Scalar(100));
	string title1 = "white window control";
	string title2 = "gray window control";

	namedWindow(title1, WINDOW_AUTOSIZE);
	namedWindow(title2, WINDOW_NORMAL);
	moveWindow(title1, 100, 200);
	moveWindow(title2, 300, 200);

	imshow(title1, image1);
	imshow(title2, image2);
	waitKey();
	destroyAllWindows();
	return 0;
}
```

4.2 Klawiatura hadysalaryny dolandyrmak

- waitKey (gijikdirme): gijikdirme wagtynda düwme girişine garaşýar, klawiatura hasydasy ýüze çykanda klawiatura derejesini yzyna gaýtarýar

- delay: gijikdirme wagty. ms.
- delay <= 0: açar bir hadysa ýüze çykýança tükeniksiz garaşýar

- delay > 0: gijikdirilen wagtyň içinde açaryň girizilmegine garaşyň. Gijikdirme döwründe hiç hili düwme girişi ýok bolsa -1 yzyna gaýtarýar

- Düwmäniň kömegi bilen klawiatura giriş üçin waitKeyEx () ulanyň
- Hadysa diňe penjire aktiw bolanda ýüze çykýar.

Belgileýji	
EVENT_MOUSEMOVE Python cv EVENT_MOUSEMOVE	Syçanyň görkezijisiniň penjiräniň üstünden geçendigini görkezýär
EVENT_LBUTT ONDOWN Python cv EVENT_LBUTTONDOWN	Syçanyň çep düwmesiniň basylandygyny görkezýär
EVENT_RBUTT ONDOWN Python cv EVENT_RBUTTONDOWN	Syçanyň sag düwmesiniň basylandygyny görkezýär
EVENT_MBUTTONDOWN Python cv EVENT_MBUTTONDOWN	Syçanyň orta düwmesiniň basylandygyny görkezýär
EVENT_LBUTTONUP Python cv EVENT_LBUTTONUP	Syçanyň çep düwmesiniň goýberilendigini görkezýär
EVENT_RBUTTONUP Python cv EVENT_RBUTTONUP	Syçanyň sag düwmesiniň goýberilendigini görkezýär
EVENT_MBUTTONUP Python cv EVENT_MBUTTONUP	Syçanyň orta düwmesiniň goýberilendigini görkezýär
EVENT_LBUTTONDBLCLK Python cv EVENT_LBUTTONDBLCLK	Syçanyň çep düwmesiniň iki gezek basylandygyny görkezýär
EVENT_RBUTT ONDBLCLK Python cv EVENT_RBUTTONDBLCLK	Syçanyň sag düwmesiniň iki gezek basylandygyny görkezýär
EVENT_MBUTT ONDBLCLK Python cv EVENT_MBUTTONDBLCLK	Syçanyň orta düwmesine iki gezek basylandygyny görkezýär
EVENT_MOUSEWHEEL Python cv EVENT_MOUSEWHEEL	Položitel we otrisatel derejeleri degişlilikde öňe we yza hereket etdirmegi aňladýar
EVENT_MOUSEHWHEEL Python CV EVENT_MOUSEHWHEEL	Položitel we otrisatel derejeleri degişlilikde saga we çepe hereket etdirmegi aňladýar

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void onMouse(int, int, int, int, void *);
int main()
{
	Mat image(200, 300, CV_8U);
	image.setTo(255);
	imshow(“mouse event 1", image);
	imshow(“mouse event 2", image);

	setMouseCallback(“mouse event 1", onMouse, 0);
	waitKey(0);
	return 0;
}
// indiki sahypada dowam et ....
```

```
void onMouse(int event, int x, int y, int flags, void *params)
{
	switch (event)
	{
		case EVENT_LBUTTONDOWN:
			cout << “Left mouse button press" << endl;
			break;
		case EVENT_RBUTTONDOWN:
			cout << "Right mouse button press" << endl;
			break;
		case EVENT_RBUTTONUP:
			cout << "Right mouse button release" << endl;
			break;
		case EVENT_LBUTTONDBLCLK:
			cout << "Left mouse button double click" << endl;
				break;
	}
}
```

4.3 TrackBar hadysalaryny dolandyrmak

Belli bir aralykda belli bir bahany saýlamak üçin ulanylýan çyzgyç lineýkasy ýa-da slaýder çyzgyç;

TrackBar döredиň (tn, pw, sv, max, callback, data)
- tn: TrackBar-yň ady
- pw: esasy penjiräniň ady
- sv: slaýderiň bahasy
- max: slaýderiň iň ýokary bahasy (iň pes bahasy = 0)
- callback: yzyna çagyryş funksiýasy (onChange)

- data: ulanyjy maglumatlary yzyna çagyryş funksiýasyna geçirildi

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

string title = “Trackbar Event";
Mat image;

void onChange(int value, void* userdata)
{
	int add_value = value - 128;
	cout << “added pixel value " << add_value << endl;
	Mat tmp = image + add_value;
	imshow(title, tmp);
}
// dowam et ....
int main()
{
	int value = 128;
	image = Mat(300, 400, CV_8UC1, Scalar(128));

	namedWindow(title, WINDOW_AUTOSIZE);
	createTrackbar(“Brightness", title, &value, 255, onChange);

	imshow(title, image);
	waitKey(0);
	return 0;
}
```

4.4 Çyzyk, gönüburçluk çyzgysy

- çyzyk (img, pt1, pt2, color, thickness, linetype, sift)
- gönüburçluk (img, pt1, pt2, color, thickness, linetype, sift)

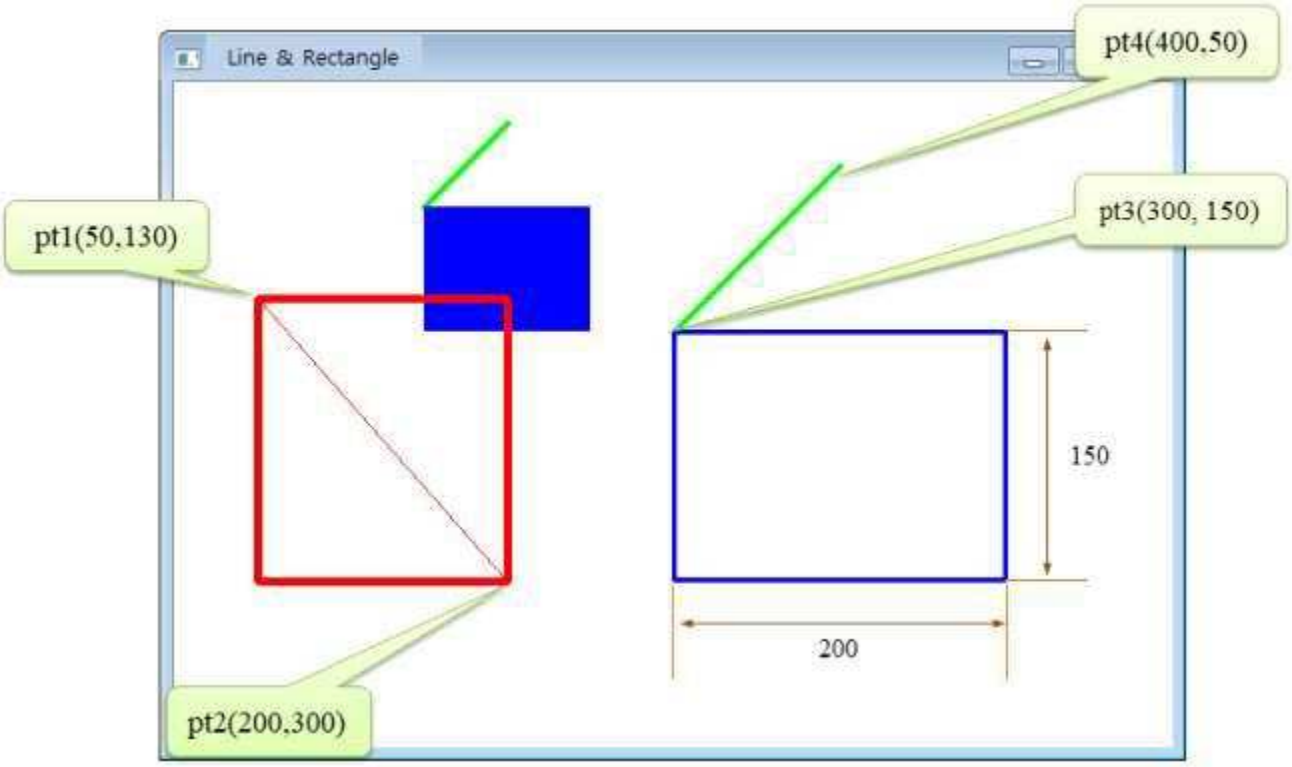
- img: matrisa (surat)
- pt1, pt2: ýyldyz nokady, ahyrky nokat
- reňk: çyzyk ýa-da gönüburçly reňk
- galyňlygy: çyzygyň galyňlygy (-1)
- linetype

Görnüşi	Derejesi	Düşündirilişi
LINE_4	4	4 birikdirilen çyzyk
LINE_8	8	8 birikdirilen çyzyk
LINE_AA	16	Garşylyga garşy çyzyk

- sift: saga biraz süýşmek

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	Scalar blue(255, 0, 0), red(0, 0, 255), green = Scalar(0, 255, 0);
	Scalar white(255, 255, 255);
	Scalar yellow(0, 255, 255);

	Mat image(400, 600, CV_8UC3, white);
	Point pt1(50, 130), pt2(200, 300), pt3(300, 150), pt4(400, 50);
	Rect rect(pt3, Size(200, 150));

	line(image, pt1, pt2, red);
	line(image, pt3, pt4, green, 2, LINE_AA);
	line(image, pt3, pt4, green, 3, LINE_8, 1);

	rectangle(image, rect, blue, 2);
	rectangle(image, rect, blue, FILLED, LINE_4, 1);
	rectangle(image, pt1, pt2, red, 3);

	imshow(“Line & Rectangle", image);
	waitKey(0);
	return 0;
}
```

- Çykarylan netije



4.5 Çyzgy teksti

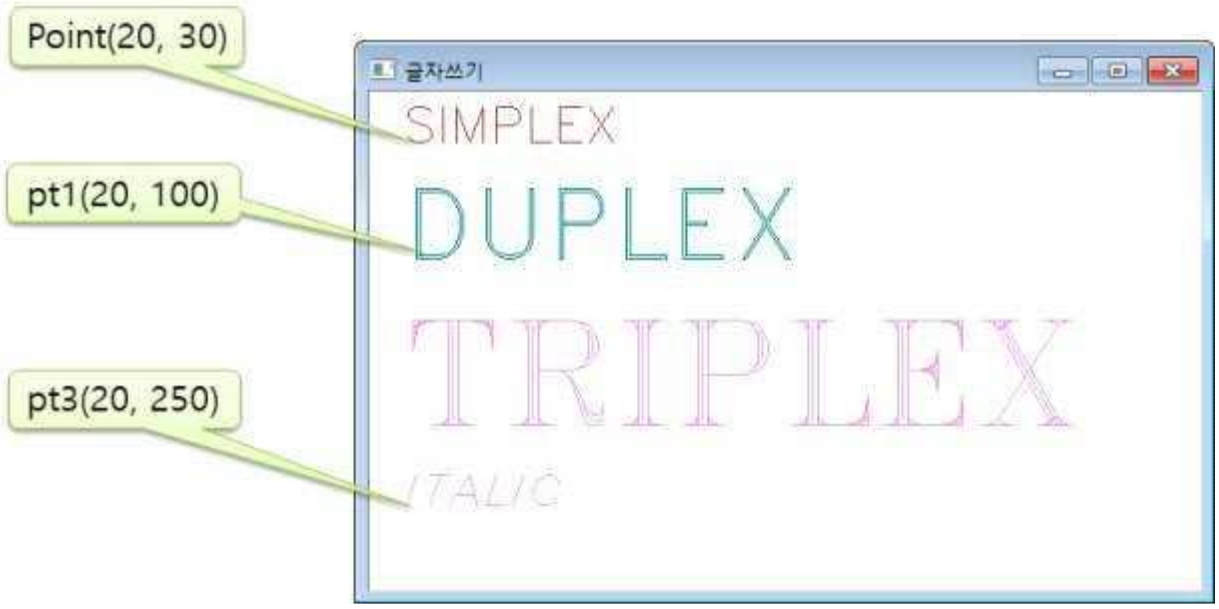
putText (img, text, org, fontFace, fontScale, color, thickness, linetype)

- img: Matrisa tekst ýazmak (surat)
- text: hat ýazmak
- org: tekstiň koordinatlaryny başlatmak
- fontFace: tekstiň ýazylyş görnüşi (şrifti)
- fontScale: Şriftiň ululygyny ýokarlandyrmak faktory
- color: tekstiň reňki
- thickness: tekstiň galyňlygy
- linetype: tekst setiriniň görnüşi (default = 8)
  - Ekranyň çyzygynyň başlangyç koordinatlary çep aşaky tarapda
  - fontFace: tekstiň şrifti

Belgileýji	
FONT_HERSHEY_SIMPLEX Python cv FONT_HERSHEY_SIMPLEX	adaty ölçegli sans-serif şrifti
FONT_HERSHEY_PLAIN Python cv FONT_HERSHEY_PLAIN	kiçi ölçegli sans-serif şrifti
FONT_HERSHEY_DUPLEX Python cv FONT_HERSHEY_DUPLEX	adaty ölçegli sans-senf şrifti (FONT_HERSHEY_SIMPLEX-den has çylşyrymly)
FONT_HERSHEY_COMPLEX Python cv FONT_HERSHEY_COMPLEX	adaty ölçegli serif şrifti
FONT_HERSHEY_TRIPLEX Python cv FONT_HERSHEY_TRIPLEX	adaty ölçegli senf şrifti (FONT_HERSHEY_COMPLEX-den has çylşyrymly)
FONT_HERSHEY_COMPLEX_SMALL Python CV FONT_HERSHEY_COMPLEX_SMALL	FONT_HERSHEY_COMPLEX-iň kiçi görnüşi
FONT_HERSHEY_SCRIPT_SIMPLEX Python cv FONT_HERSHEY_SCRIPT_SIMPLEX	El bilen ýazylýan görnüş şrifti
FONT_HERSHEY_SCRIPT_COMPLEX Python cv FONT_HERSHEY_SCRIPT_COMPLEX	FONT_HERSHEY_SCRIPT_SIMPLEX-iň has çylşyrymly görnüşi
FONT_ITALIC Python cv FONT_ITALIC	Italik (ýasy) şrift üçin baýdak

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	Scalar olive(128, 128, 0), violet(221, 160, 221), brown(42, 42, 165);
	Point pt1(20, 100), pt2(20, 200), pt3(20, 250);

	Mat image(300, 500, CV_8UC3, Scalar(255, 255, 255));
	putText(image, "SIMPLEX", Point(20, 30), FONT_HERSHEY_SIMPLEX, 1,
	brown);
	putText(image, "DUPLEX", pt1, FONT_HERSHEY_DUPLEX, 2, olive);
	putText(image, "TRIPLEX", pt2, FONT_HERSHEY_TRIPLEX, 3, violet);
	putText(image, "ITALIC", pt3, FONT_HERSHEY_PLAIN | FONT_ITALIC,
	2, violet);
	imshow(“Text Drawing", image);
	waitKey(0);
	return 0;
}
```

- Çykarylan netije



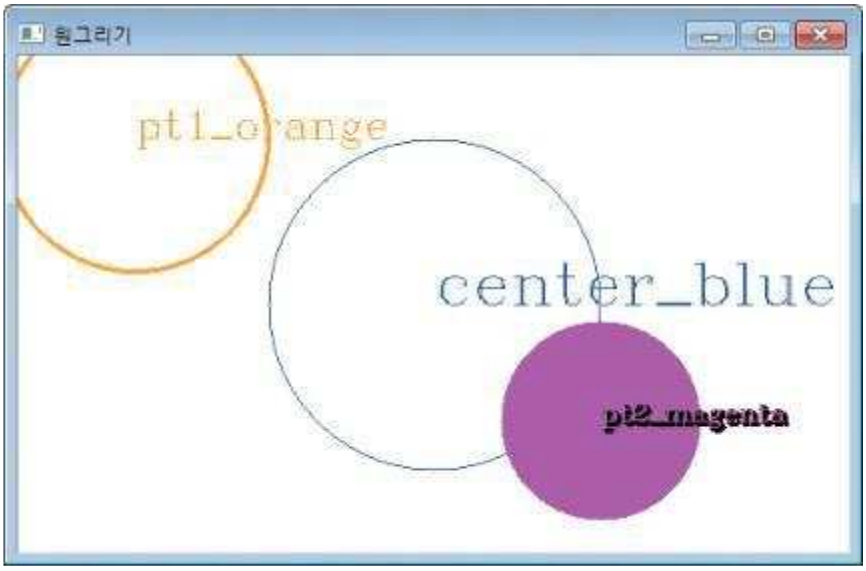
4.6 Töweregiň çyzgysy

- circle(img, center, radius, color, thickness, linetype)

- img: töwerek çyzmak üçin matrisa (surat)
- center: töweregiň merkezi koordinatlary
- radius: töweregiň radiusy
- color: töweregiň reňki
- thickness: töweregiň galyňlygy
- linetype: töwerek çyzgysynyň görnüşi (default = 8)

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	Scalar orange(0, 165, 255), blue(255, 0, 0), magenta(255, 0, 255);
	Mat image(300, 500, CV_8UC3, Scalar(255, 255, 255));

	Point center = image.size() / 2;
	Point pt1(70, 50), pt2(350, 220);

	circle(image, center, 100, blue);
	circle(image, pt1, 80, orange, 2);
	circle(image, pt2, 60, magenta, -1);

	int font = FONT_HERSHEY_COMPLEX;
	putText(image, "center_blue", center, font, 1.2, blue);
	putText(image, "pt1_orange", pt1, font, 0.8, orange);
	putText(image, "pt2_magenta", pt2 + Point(2, 2), font, 0.5, Scalar(0,
	0, 0), 2);
	putText(image, "pt2_magenta", pt2, font, 0.5, magenta, 1);

	imshow(“Circle Drawing", image);
	waitKey(0);
	return 0;
}
```

- Çykarylan netije



4.7 Surat faýlyny işläp taýýarlamak

- imread (file name, flags)

- file name: okaljak surat faýlynyň ady

- flags: reňkiň görnüşi

Belgileýji	
IMREAD_UNCHANGED Python cv IMREAD_UNCHANGED	Gurnalan bolsa, ýüklenen suraty bir görnüşde yzyna gaýtarýar (alfa kanaly bilen bolmasa 4 sanysy kesiler)
IMREAD_GRAYSCALE Python cv IMREAD_GRAYSCALE	Gurnalan bolsa, suraty hemişe ýeke kanal çal reňkli şekile öwürýär (kodek içerki öwrülişigi)
IMREAD_COLOR Python cv IMREAD_COLOR	Gurnalan bolsa, suraty hemişe 3 kanally BGR reňkli şekile öwürýär
IMREAD_ANYDEPTH Python cv IMREAD_ANYDEPTH	Girişe garşylyk gelýän çuňluga eýe bolanda 16 bit / 32 bit suraty yzyna gaýtarsa, ýogsam 4-di 8 bite öwüriň
IMREAD_ANYCOLOR Python cv IMREAD_ANYCOLOR	Gurnalan bolsa, surat islendik reňk görnüşinde okalýar
IMREAD_LOAO_GDAL Python cv IMREAD_LOAO_GDAL	Suraty ýüklemek üçin gdal draýwerini ulanyň

IMREAD_REDUCED_GRAYSCALE_2 Python cv IMREAD_REDUCED_GRAYSCALE_2	Gurnalan bolsa, hemişe suraty ýeke kanally çal reňkli surata öwürýär we suratyň ölçegi 1/2-e çenli azalýar
IMREAD_REDUCED_COLOR_2 Python cv lMREAD_REDOCED_COLOR_2	Gurnalan bolsa, surat hemişe 3 kanally BGR reňkli surata öwürýär we suratyň ölçegi 1/2-e çenli azalýar
IMREAD_REDUCED_GRAYSCALE_4 Python cv IMREAD_REDUCED_GRAYSCALE_4	Gurnalan bolsa, hemişe suraty ýeke kanally çal reňkli surata öwürýär we suratyň ölçegi 1/4-e çenli azalýar
IMREAD_REDUCEO_COLOR_4 Python cv IMREAO_REDUCED_COLOR_4	Gurnalan bolsa, surat hemişe 3 kanally BGR reňkli surata öwürýär we suratyň ölçegi 1/4-e çenli azalýar
IMREAD_REDUCED_GRAYSCALE_8 Python cv IMREAD_REDUCED_GRAYSCALE_8	Gurnalan bolsa, hemişe suraty ýeke kanally çal reňkli surata öwürýär we suratyň ölçegi 1/8-e çenli azalýar
IMREAD_REDUCED_COLOR_8 Python cv IMREAD_REDUCED_COLOR_8	Gurnalan bolsa, surat hemişe 3 kanally BGR reňkli surata öwürýär we suratyň ölçegi 1/8-e çenli azalýar
IMREAD_IGNORE_ORIENTATION Python cv IMREAD_IGNORE_ORIENTATION	Gurnalan bolsa, suraty EXIP-iň ugrukdyryjy baýdagyna görä öwürmäň.

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void print_matInfo(string name, Mat img)
{
	string str;
	int depth = img.depth();
	if (depth == CV_8U) str = "CV_8U";
	else if (depth == CV_8S) str = "CV_8S";
	else if (depth == CV_16U) str = "CV_16U";
	else if (depth == CV_16S) str = "CV_16S";
	else if (depth == CV_32S) str = "CV_32S";
	else if (depth == CV_32F) str = "CV_32F";
	else if (depth == CV_64F) str = "CV_64F";
	cout << name;
	cout << format(": depth(%d) channels(%d) -> data type: ", depth,
	img.channels());
	cout << str << "C" << img.channels() << endl;
}
// indiki sahypda dowam et ....
```

```
int main()
{
	string filename = "../image/read_color.jpg";
	Mat color2gray = imread(filename, IMREAD_GRAYSCALE);
	Mat color2color = imread(filename, IMREAD_COLOR);
	CV_Assert(color2gray.data && color2color.data);

	Rect roi(100, 100, 1, 1);
	cout << “Matrix (100,100) pixel value " << endl;
	cout << "color2gray " << color2gray(roi) << endl;
	cout << "color2color " << color2color(roi) << endl;

	print_matInfo("color2gray", color2gray);
	print_matInfo("color2color", color2color);
	imshow("color2gray", color2gray);
	imshow("color2color", color2color);
	waitKey(0);
	return 0;
}
```

- imwrite (file name, image, parameter vector)

- file name: faýlyň adyny ýatda saklaýar
- image: matrisa (surat) ýazdyrylar
- parameter vector: parametr wektor jübütine baglylykda - gysyş usuly

Belgileýji	
IMWRITE_JPEG_QUALITY Python cv IMWRITE_JPEG_QUALITY	JPEG üçin 0 bilen 100 arasynda hil bolup biler (näçe ýokary bolsa şonça gowy) Bellenen (default) derejesi 95
IMWRITE_PNG_COMPRESSION Python cv IMWRITE_PNG_COMPRESSION	PNG üçin gysyş derejesi 0 bilen 9 aralygynda bolup biler. Has ýokary dereje has kiçi ölçegi we has uzyn gysyş wagtyny aňladýar, strategiýa IMWRITE_PNG_STRATEGY_DEFAULT (Z_DEFAULT_STRATEGY) görnüşinde üýtgedilýär. Bellenen (default) derejesi 1 (iň gowy tizlik sazlamasy).

- Giňeltme ady bilen suratyň faýlyny ýatda saklamak aňsat.
- Surat faýl formatlary: JPG, BMP, PNG, TIF, PPM we ş.m.

```
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;
int main()
{
	Mat img8 = imread("../image/read_color.jpg", IMREAD_COLOR);
	CV_Assert(img8.data);
	vector<int> params_jpg, params_png;
	params_jpg.push_back(IMWRITE_JPEG_QUALITY);
	params_jpg.push_back(50);
	params_png.push_back(IMWRITE_PNG_COMPRESSION);
	params_png.push_back(9);
	imwrite("../image/write_test1.jpg", img8);
	imwrite("../image/write_test2.jpg", img8, params_jpg);
	imwrite("../image/write_test.png", img8, params_png);
	imwrite("../image/write_test.bmp", img8);
	return 0;
}
```

4.8 Wideo-ny işläp taýýarlamak

- Wideo faýly, format boýunça kodek bilen gysylýar we ýazga alnan wideo faýly açylýar.
- VideoCapture topary: Kameradan ýa-da wideo faýlyndan kadry (kaframe) okaýar.

- VideoCapture (faýlyň ady, enjam)
- device: Surata alýan enjamyň şahsyýetnamasy (kamera bar bolsa 0)
- open (): wideo faýly ýa-da kamerany açmak
- isOpened (): Surata alýan enjamyň aýlanmagyny birikdirýär
- get (): aýratynlyk kesgitleýjiniň derejesini öwürýär
- set ():aýratynlyk kesgitleýjini ulanyp wideo aýratynlyklaryny sazlaýar
- read (): wideo kadrynyň okaýar, soňra surat matrisasyna geçirýär

- **Aýratynlyk kesgitleýji**

Belgileýji	
CAP_PROP_POS_MSEC Python cv CAP_PROP_POS_MSEC	Wideo faýlyň millisekuntda häzirki ýerleşişi
CAP_PROP_POS_FRAMES Python cv CAP_PROP_POS_FRAMES	Soňra kody çözüljek / alynjak kadryň 0 esasly görkezijisi
CAP_PROP_POS_AVl_RATIO Python cv CAP_PROP_POS_AVl_RATIO	Wideo faýlyň otnositel ýagdaýy 0 = filmiň başlangyjy, 1 = filmiň soňy
CAP_PROP_FRAME_WIDTH Python cv CAP_PROP_FRAME_WIDTH	Wideonyň akymyndaky kadrlaryň giňligi
CAP_ PROP_FRAME_HEIGHT Python cv CAP_ PROP_FRAME_HEIGHT	Wideonyň akymyndaky kadrlaryň beýikligi
CAP_PROP_FPS Python cv CAP_PROP_FPS	Kadryň tizligi
CAP_PROP_FOURCC Python cv CAP_PROP_FPS	4 simwolly kody VideoWriter :: fource serediň
CAP_PROP_FRAME_COUNT Python cv CAP_PROP_FRAME_COUNT	Wideo faýldaky kadrlaryň sany
CAP_PROP_FORMAT Python cv CAP_PROP_FORMAT	VideoCapture::retrieve() tarapyndan yzyna gaýtarylan Mat obýektleriniň formaty
CAP_PROP_MODE Python cv CAP_PROP_MODE	Häzirki surata düşüriş tertibini görkezýän arka derejesi

CAP_PROP_BRIGHTNESS Python cv CAP_PROP_BRIGHTNESS	Suratyň ýagtylygy (diňe goldaýan kameralar üçin)
CAP_PROP_CONTRAST Python cv CAP_PROP_CONTRAST	Surat kontrasty (diňe goldaýan kameralar üçin)
CAP_PROP_SATURATION Python cv CAP_PROP_SATURATION	Suratyň doýgunlygy (diňe goldaýan kameralar üçin)
CAP_PROP_HUE Python cv CAP_PROP_HUE	Suratyň öwüşgini (diňe goldaýan kameralar üçin)
CAP_PROP_GAIN Python cv CAP_PROP_HUE	Suraty almak (diňe goldaýan kameralar üçin)
CAP_PROP_EXPOSURE Python cv CAP_PROP_EXPOSURE	Ekspozisiýa (diňe goldaýan kameralar üçin)
CAP_PROP_AUTOFOCUS Python cv CAP_PROP_AUTOFOCUS	

- VideoWriter topary: surat matrisasyny wideo faýlyna ýazdyrmak

- VideoWriter (file name, fourcc, fps, framesize, isColor)
- fourcc: 4 simwolly kodek kody
- fps: sekuntda kadrlar
- framesize: wideo kadyrynyň ölçegi (sütün x setir)
- isColor: dogry (ture) - reňkli echo, ýalnyş (false) - çal kadryň kodlanmagy
- open (): wideo faýlyny açmak
- isOpened (): wideo ýazmak faýlyny öwürülişigini açmak
- write (): surat matrisasyndan wideo kadryny ýazmak
- 4 simwolly kodek kody (www.fourcc.org)

4 simwolly kod	**Düşündirilişi**
CV.FOU RCC.PROM PT	Kodek saýlamasy
VideoWriter::fourcc('D', T, 'V, ’4')	DivX MPEG-4
VideoWriter::fourcc('D', T, 'V, '5')	Div5
VideoWriter::fourcc('D', '1', 'V, ’X')	DivX MPEG-4
VideoWriter::fourcc(D', 'X', '5, ’O')	DivX MPEG-4
VideoWriter::fourcc('F’, 'M', 'R '4')	Ffmpeg
VideoWriter::fourcc(T, 'Y', ’U, 'V')	İYUV

VideoWriter::fourcc('M', 'J', 'R 'G')	Hareketli JPEG kodek
VideoWriter::fourcc('M', 'P', '4, '2')	MPEG v2
VideoWriter::fourcc('M', 'P', 'E, 'G')	MPEG kodekleri
VideoWriter::fourcc('X'; 'V', '1, 'D')	XVID kodekleri
VideoWriter::fourcc('X', '2', '6, '4')	H.264 / AVC kodekleri

```
// Wideo kadryny okaýan programma
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	VideoCapture capture(0);
	if (!capture.isOpened())
	{
	cout << “camera not connected!!" << endl;
	exit(1);
	}
	cout<<“Width "<<capture.get(CAP_PROP_FRAME_WIDTH) << endl;
	cout<<“Height"<<capture.get(CAP_PROP_FRAME_HEIGHT) << endl;
	cout<<“Exposure "<<capture.get(CAP_PROP_EXPOSURE) << endl;
	cout<<“Brightness "<<capture.get(CAP_PROP_BRIGHTNESS) << endl;

	Point shade = Point(10, 40) + Point(2, 2);
	int font = FONT_HERSHEY_SIMPLEX;

	string text = "EXPOS: " + to_string((int)capture.get(CAP_PROP_EXPOSURE));
	Mat frame;
```

```
        for (;;) {
        capture.read(frame);
        putText(frame, text, shade, font, 0.7, Scalar(0, 0, 0), 2);
        putText(frame, text, Point(10, 40), font, 0.7, Scalar(120, 200, 90), 2);
        imshow("Camera Viewer", frame);
        if (waitKey(30) >= 0)
                break;
        }
        return 0;
}
```

```
// Wideo kadryny ýazýan programma
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;
int main()
{
	VideoCapture capture(0);
	CV_Assert(capture.isOpened());
	double fps = 29.97;
	int delay = cvRound(1000.0 / fps);
	Size size(640, 480);
	int fourcc = VideoWriter::fourcc('D', 'X', '5', '0');
	capture.set(CAP_PROP_FRAME_WIDTH, size.width);
	capture.set(CAP_PROP_FRAME_HEIGHT, size.height);
	cout << "width x height : " << size << endl;
	cout << "VideoWriterfourcc : " << fourcc << endl;
	cout << "delay : " << delay << endl;
	cout << "fps : " << fps << endl;
	VideoWriter writer;
	writer.open("../image/video_file.avi", fourcc, fps, size);
	CV_Assert(writer.isOpened());
	Mat frame;
	for (;;) {
		capture >> frame;
		writer << frame;
		imshow(“Camera Viewer", frame);
		if (waitKey(delay) >= 0)
			break;
	}
	return 0;
}
```

```
// Wideo faýlyny okaýan programma
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	VideoCapture capture;
	capture.open("../image/video_file.avi");
	CV_Assert(capture.isOpened());

	Mat frame;
	double frame_rate = capture.get(CAP_PROP_FPS);
	int delay = 1000 / frame_rate;
	int frame_cnt = 0, font = FONT_HERSHEY_PLAIN;
	Point pt(25, 55);
	Point shade = pt + Point(2, 2);

	while (capture.read(frame))
{
		if (waitKey(delay) >= 0) break;

		string text = "Frame Count : ";
		text += to_string(frame_cnt++);
		putText(frame, text, shade, font, 1.8, Scalar(0, 0, 0), 2);
		putText(frame, text, pt, font, 1.8, Scalar(120, 200, 90), 2);
		imshow(“Video File Reading", frame);
	}
	return 0;
}
```

5. Massiwlerde OpenCV amaly

5.1 Massiwleri işläp taýýarlamagyň esasy funksiýalary

- flip (src, dst, flipCode): dik, kese, iki taraply hem süýşme

- src: input Array - dst: out Array
- flipCode: 0 - keseligine öwürmek
- 1 - dikligine öwürmek
-1 - keseligine we dikligine öwürmek

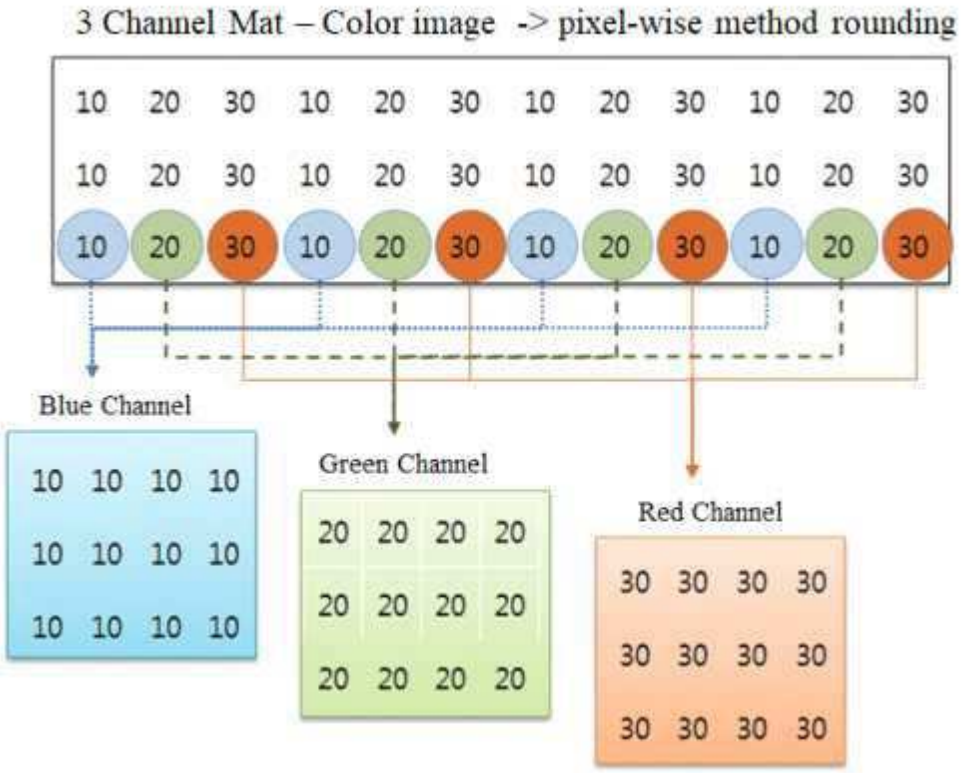
- transpose (): giriş matrisasynyň transpose matrisasyny yzyna gaýtarýar

```
// Suratlary öwürmek programmasy
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	Mat image = imread("../image/flip_test.jpg", IMREAD_COLOR);
	CV_Assert(image.data);

	Mat x_axis, y_axis, xy_axis, trans_img;
	flip(image, x_axis, 0);
	flip(image, y_axis, 1);
	flip(image, xy_axis, -1);

	transpose(image, trans_img);

	imshow("image", image);
	imshow("x_axis", x_axis);
```

```
    imshow("y_axis", y_axis);
    imshow("xy_axis", xy_axis);
    imshow("trans_img", trans_img);

    waitKey();
    return 0;
}
```

5.2 Kanaly işläp taýýarlamak funksiýalary

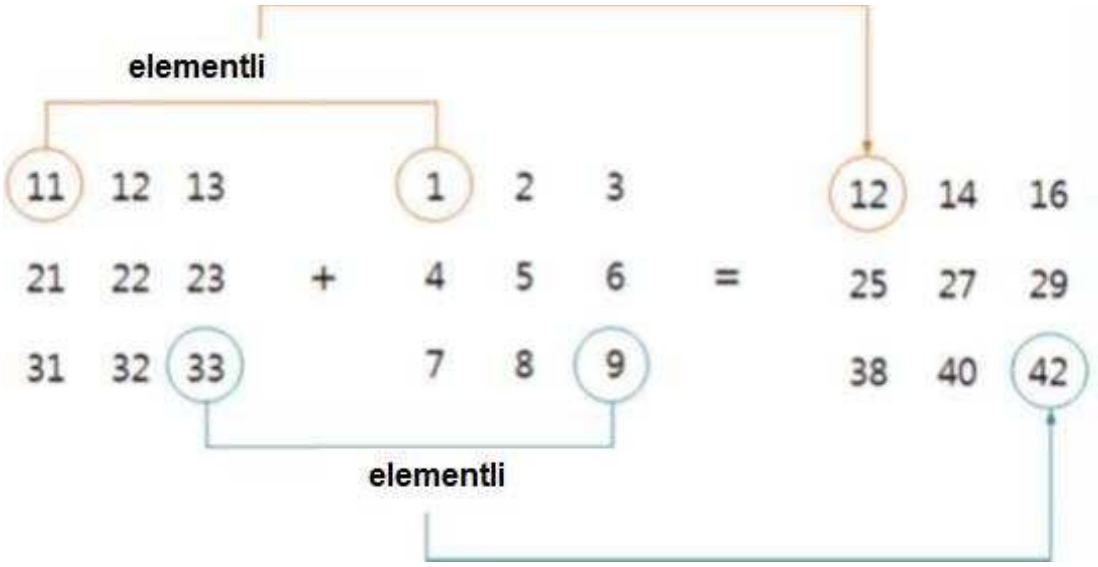
- 3 kanally matrisanyň (reňkli) elementlerini ýatda saklamagyň usuly



```
// Surat kanalyny bölmek programmasy
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	Mat image = imread("../image/color.jpg", IMREAD_COLOR);
	CV_Assert(image.data);

	Mat bgr[3];
	split(image, bgr);
	imshow("image", image);
	imshow(“Blue Channel", bgr[0]);
	imshow(“Green Channel", bgr[1]);
	imshow(“Red Channel", bgr[2]);
	waitKey(0);
	return 0;
}
```

5.3 Massiwleriň dört esasy hereketi

- Element esasly amaly ýerine ýetirýär
- add/goşmak (), subtract/aýyrmak (), multiply/köpeltmek (), divide/bölmek ()



```
// Massiwleriň dört esasy amaly
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	Mat m1(3, 6, CV_8UC1, Scalar(10));
	Mat m2(3, 6, CV_8UC1, Scalar(50));
	Mat m_add1, m_add2, m_sub, m_div1, m_div2;
	Mat mask(m1.size(), CV_8UC1, Scalar(0));

	Rect rect(Point(3, 0), Size(3, 3));
	mask(rect).setTo(1);

	add(m1, m2, m_add1);
	add(m1, m2, m_add2, mask);
	divide(m1, m2, m_div1);
	m1.convertTo(m1, CV_32F);
	m2.convertTo(m2, CV_32F);
	divide(m1, m2, m_div2);
	cout << "[m1] = " << endl << m1 << endl;
	cout << "[m2] = " << endl << m2 << endl;
	cout << "[mask] = " << endl << mask << endl << endl;
	cout << "[m_add1] = " << endl << m_add1 << endl;
	cout << "[m_add2] = " << endl << m_add2 << endl;
	cout << "[m_div1] = " << endl << m_div1 << endl;
	cout << "[m_div2] = " << endl << m_div2 << endl;
	return 0;
}
```

5.4 Massiw amalynyň köki, güýji, ululygy

- sqrt (input, output): ähli massiw elementleriniň kwadrat kök hasaplamasy
- pow (input, power, output): ähli massiw elementleriniň güýç hasaplamasy
- magnitude (x, y, output): x we y wektorlarynyň ululyk hasaplamasy

$$magnitude(i) = \sqrt{x(i)^2 + y(i)^2}$$

- exp (input, output): ähli massiw elementleriniň ekspensial hasaplamasy
- log (input, output): ähli massiw elementleriniň tebigy logaritm hasaplamasy

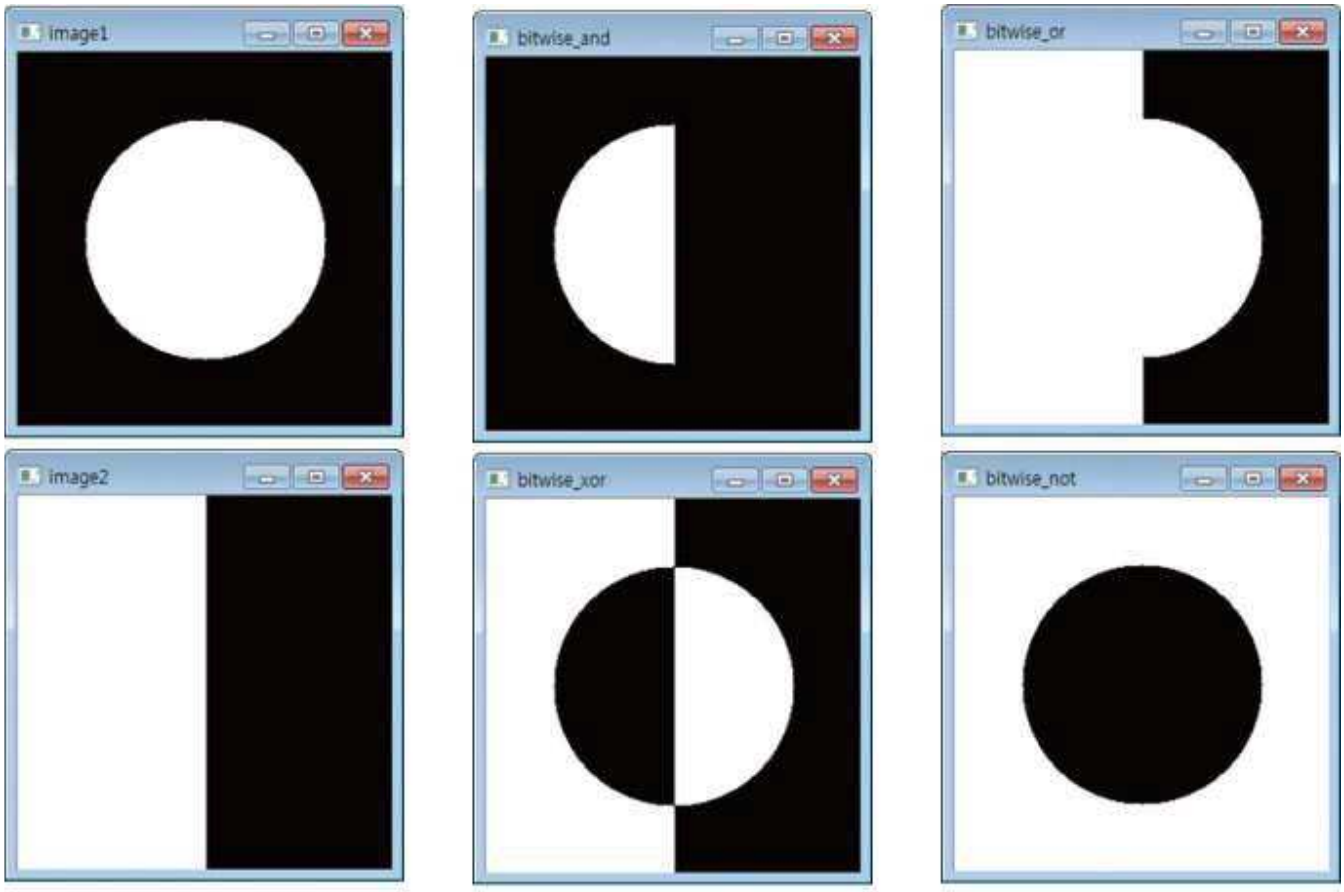
5.5 Massiwlerde logiki bit amallar

- Bit esasly massiw elementlerinde logiki amallary ýerine ýetiriň

- bitwise_and (input1, input2, output, mask)
- mask: Hasaplamalary, diňe nol derejesi bolmadyk pozisiýalar üçin ýerine ýetiriň
- bitwise_or(input1, input2, output, mask)
- bitwise_xor(input1, input2, output, mask)
- bitwise_not(input, output, mask)

```
// Massiwleriň logiki bit amallary
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	Mat image1(250, 250, CV_8U, Scalar(0));
	Mat image2(250, 250, CV_8U, Scalar(0));
	Mat image3, image4, image5, image6;

	Point center = image1.size() / 2;
	circle(image1, center, 80, Scalar(255), -1);
	rectangle(image2, Point(0, 0), Point(125, 250), Scalar(255), -1);

	bitwise_or(image1, image2, image3);
	bitwise_and(image1, image2, image4);
	bitwise_xor(image1, image2, image5);
	bitwise_not(image1, image6);

	imshow("image1", image1);
	imshow("image2", image2);
	imshow("bitwise_or", image3);
	imshow("bitwise_and", image4);
	imshow("bitwise_xor", image5);
	imshow("bitwise_not", image6);
	waitKey();
	return 0;
}
```

- Çykarylan netije



5.6 Massiw amalynyň Maks., Min. absolýut derejesi

abs (): ähli massiw elementleriniň absolýut derejesini hasaplaýar

absdiff (): Iki massiw element boýunça aýrylandan soň absolýut derejäni hasaplaýar

max (src1, src2, dst): src1 we src2-i element esasynda deňeşdirýär we dst matrisada uly derejäni yzyna gaýtarýar

min (src1, src2, dst): src1 we src2 elementleri deňeşdirilende dst matrisada pes derejeleri yzyna gaýtarýar

```
// Massiw amalynyň Maks., Min. absolýut derejesi
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;
int main()
{
	Mat image1 = imread("../image/Semiconduct.tif", 0);
	Mat image2 = imread("../image/Semiconduct2.tif", 0);
	CV_Assert(image1.data && image2.data);
	Mat dif_img, abs_dif1, abs_dif2;

	image1.convertTo(image1, CV_16S);
	image2.convertTo(image2, CV_16S);
	subtract(image1, image2, dif_img);
	abs_dif1 = abs(dif_img);

	image1.convertTo(image1, CV_8U);
	image2.convertTo(image2, CV_8U);
	dif_img.convertTo(dif_img, CV_8U);
	abs_dif1.convertTo(abs_dif1, CV_8U);

	absdiff(image1, image2, abs_dif2);
	imshow("image1", image1), imshow("image2", image2);
	imshow("dif_img", dif_img);
	imshow("abs_dif1", abs_dif1), imshow("abs_dif2", abs_dif2);

	Mat image_max, image_min;
	image1 = imread("../image/abs_test1.jpg", 0);
	image2 = imread("../image/abs_test2.jpg", 0);
	CV_Assert(image1.data && image2.data);
```

```
    max(image1, 120, image_max);
    min(image1, image2, image_min);
    image_max.convertTo(image_max, CV_8U);
    image_min.convertTo(image_min, CV_8U);

    imshow("image_max", image_max);
    imshow("image_min", image_min);
    waitKey();
    return 0;
}
```

5.7 Massiwler bilen amallaryň statistikasy

sum(input): bir massiwdaky her kanalyň elementleriniň jemini hasaplaýar

mean(input, mask): Massiwdaky her bir kanal üçin elementleriň ortaça mukdaryny hasaplaýar

mask: Hasaplamalary diňe nol bolmadyk derejelere eýe bolan pozisiýalar üçin ýerine ýetirýär

meanStdDev(input, mean, stddev, mask): Massiw elementleriniň ortaça we standart gyşarmagyny hasaplaýar

countNonZero (): massiwdäki nol bolmadyk elementlerip sanyny yzyna gaýtarýar

```
// Massiwler bilen amallaryň statistikasy
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	Mat image = imread("../image/sum_test.jpg", 1);
	CV_Assert(image.data);

	Mat mask(image.size(), CV_8U, Scalar(0));
	mask(Rect(20, 40, 70, 70)).setTo(255);

	Scalar sum_value = sum(image);
	Scalar mean_value1 = mean(image);
	Scalar mean_value2 = mean(image, mask);
	cout << "[sum_value] = " << sum_value << endl;
	cout << "[mean_value1] = " << mean_value1 << endl;
	cout << "[mean_value2] = " << mean_value2 << endl << endl;
	Mat mean, stddev;
	meanStdDev(image, mean, stddev);
	cout << "[mean] = " << mean << endl;
	cout << "[stddev] = " << stddev << endl << endl;

	meanStdDev(image, mean, stddev, mask);
	cout << "[mean] = " << mean << endl;
	cout << "[stddev] = " << stddev << endl;
	imshow("image", image), imshow("mask", mask);
	waitKey();
	return 0;
}
```

6. OpenCV ulanyp, suratlary işläp taýýarlamak

6.1 Surat piksellerine girmek

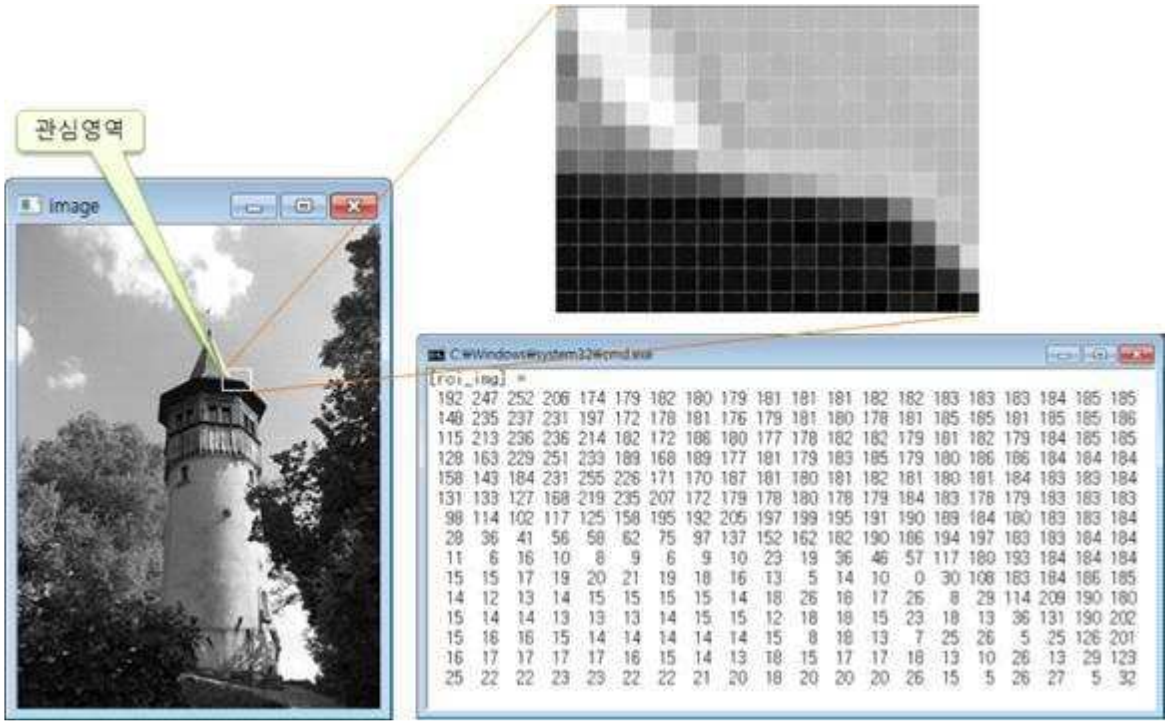
- Mat :: at (): Matrisanyň görkezilen elementine (piksel) giryän şablon funksiýasy
- Mat :: at () funksiýasynyň maglumatlary yzyna gaýtarmak görnüşi, massiw elementiniň maglumat görnüşine laýyk gelmelidir

- mat1.at <uchar> (10, 20);
- mat2.at <int> (i, j);
- mat3.at <iki esse> (y, x);
- mat4.at <Vec3d> (y, x) [0];

```
// Suratyň piksel ekrany
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	Mat image = imread("../image/pixel_test.jpg", IMREAD_GRAYSCALE);
	if (image.empty()) {
		cout << “can’t open Image!!!" << endl;
		exit(1);
	}

	Rect roi(135, 95, 20, 15);
	Mat roi_img = image(roi);
```

```
	Mat image_roi(Size(roi_img.cols * 10, roi_img.rows * 10), CV_8U,
	Scalar(0));

	for (int i = 0; i < roi_img.rows; i++) {
		for (int j = 0; j < roi_img.cols; j++) {
		for (int k = 0; k < 10; k++) {
			for (int m = 0; m < 10; m++)
			image_roi.at<uchar>(i*10+k, j*10+m) =
	roi_img.at<uchar>(i, j);
		}
		}
	}
	imshow("image_roi", image_roi);

	cout << "[roi_img] =" << endl;
	cout << roi_img << endl;

	rectangle(image, roi, Scalar(255), 1);
	imshow("image", image);
	waitKey();
	return 0;
}
```

- Çykarylan netije



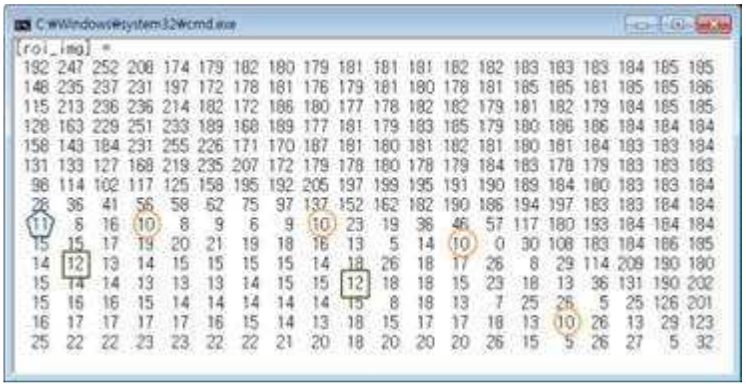
6.2 Suratyň ýagtylyk bahasyny goşmak / aýyrmak

```
// Suratyň ýagtylygynyň derejesini goşmak / aýyrmak
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
	Mat image = imread("../image/bright.jpg", IMREAD_GRAYSCALE);
	CV_Assert(!image.empty());

	Mat dst1 = image + 100; // Automatic saturate_cast
	Mat dst2 = image - 100; // Automatic saturate_cast
	Mat dst3 = 255 - image; // Image negative transform
```

```
        Mat dst4(image.size(), image.type());
        Mat dst5(image.size(), image.type());

        for (int i = 0; i < image.rows; i++) {
                for (int j = 0; j < image.cols; j++) {
                        dst4.at<uchar>(i, j) = image.at<uchar>(i, j) + 100;
                        dst5.at<uchar>(i, j) = 255 - image.at<uchar>(i, j);
                }
        }
        imshow(“Original Image", image);
        imshow("dst1 - Brighter", dst1);
        imshow("dst2 - Darker", dst2);
        imshow("dst3 - Inversion", dst3);
        imshow("dst4 - Brighter", dst4);
        imshow("dst5 - Darker", dst5);
        waitKey();
        return 0;
}
//dst4.at<uchar>(i, j)=saturate_cast<uchar>(image.at<uchar>(i, j) + 100);
```

6.3 Gistogramma

- Suratdaky her pikseliň ýagtylyk derejeleriniň sanyny görkezýän grafika
- Kese (gorizontal) ok - pikseliň ýagtylyk derejesi
- Dik ok, her pikseliň ýagtylygynyň ýygylygydyr



- OpenCV gistogramma hasaplaýyş funksiýasy bilen üpjün edilýär

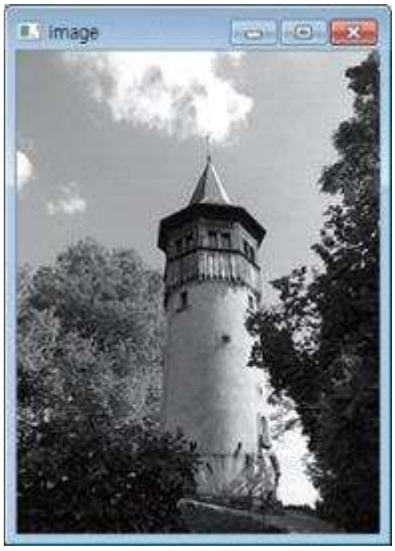
➢ calcHist (img, nimg, chn, mask, hist, dims, histsize, range)

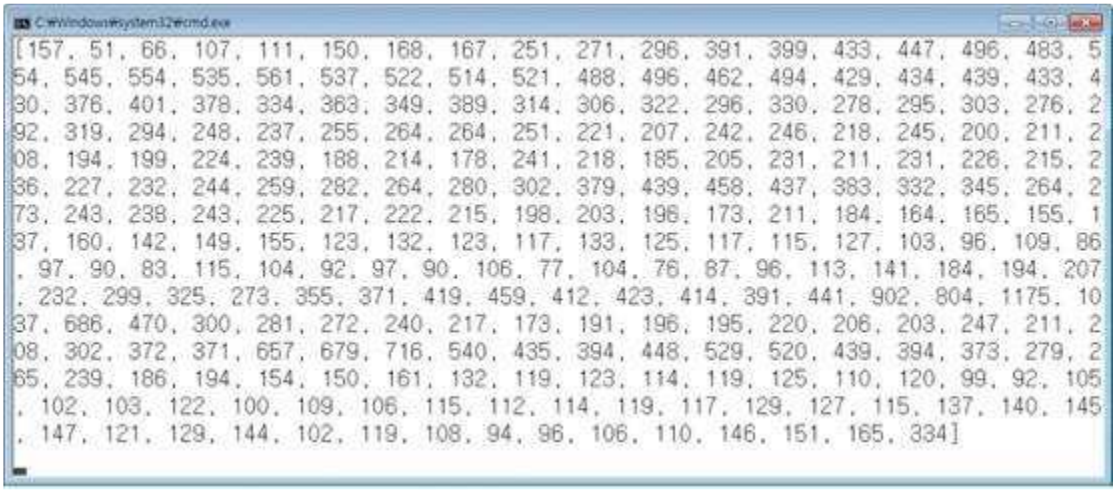
- img: asyl surat massiwi (CV_8U ýa-da CV_32F maglumat görnüşi)
- nimg: asyl surat belgileri
- chn: kanal sanawy
- mask: maska massiwi (Diňe nol bolmadyk derejä eýe bolan pozisiýalar üçin hasaplamalary ýerine ýetirýär)
- hist: gistogramma hasaplamagyň netijeleriniň yzygiderliligini saklaýar
- dims: gistogrammanyň ölçeginiň belgisi
- histsize: her ölçegiň gistogramma massiwiniň ululygy
- range: gistogrammanyň piksel derejesiniň diapazony

6.4 Gistogrammany hasaplamak

```
// Gistogrammany hasaplamak
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void calc_Histo(const Mat& image, Mat& hist, int bins, int range_max
= 256)
{
	int histSize[] = { bins };
	float range[] = { 0, (float)range_max };
	int channels[] = { 0 };
	const float* ranges[] = { range };

	calcHist(&image, 1, channels, Mat(), hist, 1, histSize, ranges); //
	OpenCV
}
int main()
{
	Mat image = imread("../image/pixel_test.jpg", IMREAD_GRAYSCALE);
	CV_Assert(!image.empty());
	Mat hist, hist_img;
	calc_Histo(image, hist, 256);

	cout << hist.t() << endl;
	imshow("image", image);
	waitKey();
	return 0;
}
```

- Çykarylan netije





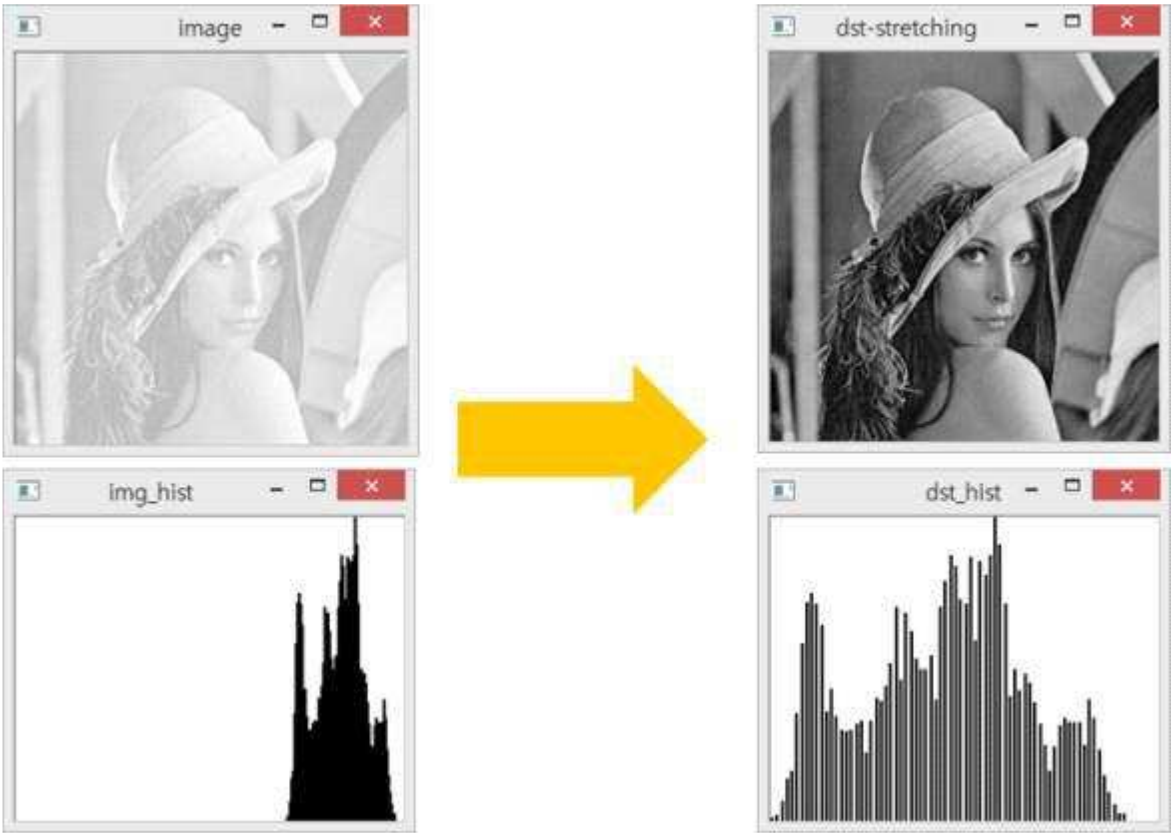
6.5 Gistogramma çyzmak

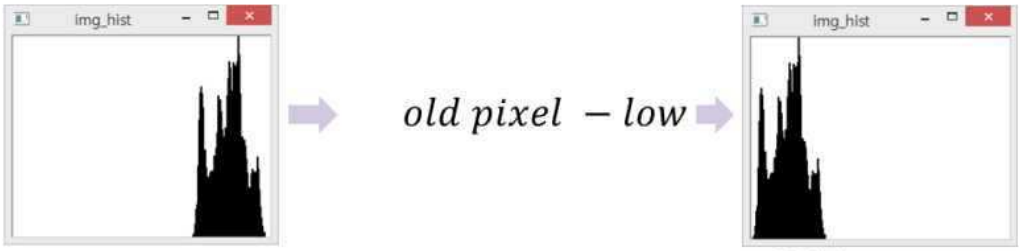
```
// Gistogrammany çyzmak programmasy
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void calc_Histo(const Mat& image, Mat& hist, int bins, int range_max = 256)
{
	int histSize[] = { bins };
	float range[] = { 0, (float)range_max };
	int channels[] = { 0 };
	const float* ranges[] = { range };

	calcHist(&image, 1, channels, Mat(), hist, 1, histSize, ranges);
}
void draw_histo(Mat hist, Mat &hist_img, Size size = Size(256, 200))
{
	hist_img = Mat(size, CV_8U, Scalar(255));
	normalize(hist, hist, 0, hist_img.rows, NORM_MINMAX);

	for (int i = 0; i < hist.rows; i++) {
		Point2f pt1 = Point2f(i, 0);
		Point2f pt2 = Point2f((i + 1), hist.at <float>(i));

		if (pt2.y > 0)
		rectangle(hist_img, pt1, pt2, Scalar(0), -1);
	}
	flip(hist_img, hist_img, 0);
}
int main()
```

```
{
    Mat image = imread("../image/lena_std.tif", IMREAD_GRAYSCALE);
    CV_Assert(!image.empty());

    Mat hist, hist_img;
    calc_Histo(image, hist, 256);
    draw_histo(hist, hist_img);

    imshow(“Histogram Image”, hist_img);
    imshow(“Image", image);
    waitKey();
    return 0;
}
```

6.6 Gistogrammany giňeltmek

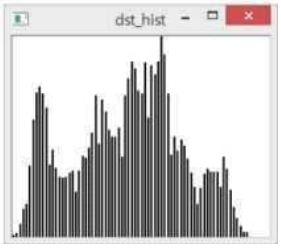
- Gistogrammalaryň insiz paýlanmagy sebäpli pes kontrastly şekiller üçin şekiliň hilini ýokarlandyrmak usullary





$$(high - low) \times x = 255 \Rightarrow x = \frac{255}{(high - low)}$$

$$(old\ pixel - low) \times x$$



$$new\ pixel = (old\ pixel) \times \frac{255}{(high - low)}$$

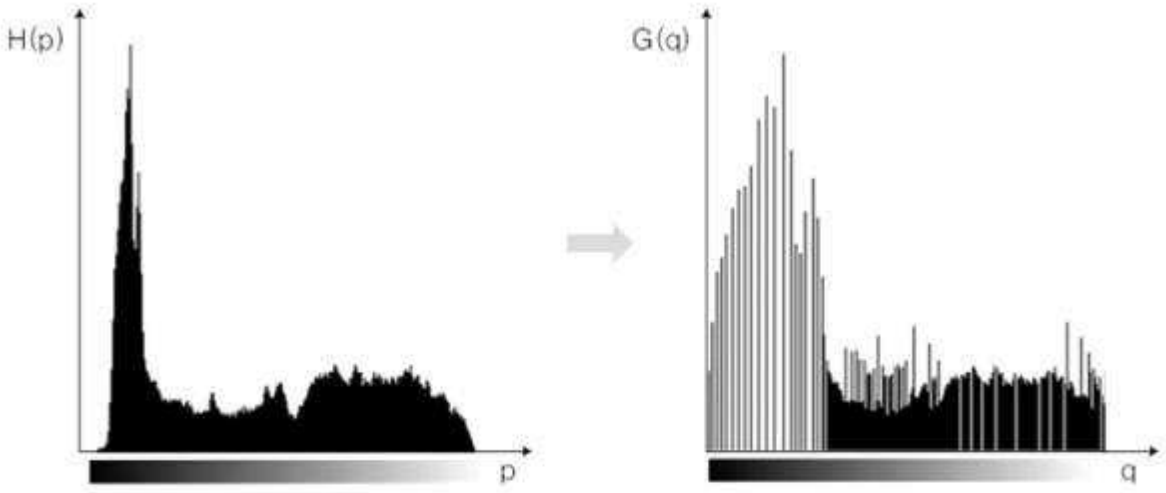
```
// Gistogrammany giňeltme programmasy
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;
void calc_Histo(const Mat& image, Mat& hist, int bins, int range_max
= 256)
{
int histSize[] = { bins };
float range[] = { 0, (float)range_max };
int channels[] = { 0 };
const float* ranges[] = { range };
calcHist(&image, 1, channels, Mat(), hist, 1, histSize, ranges);
}
void draw_histo(Mat hist, Mat &hist_img, Size size = Size(256, 200))
{
hist_img = Mat(size, CV_8U, Scalar(255));
normalize(hist, hist, 0, hist_img.rows, NORM_MINMAX);
for (int i = 0; i < hist.rows; i++) {
Point2f pt1 = Point2f(i, 0);
Point2f pt2 = Point2f((i+1), hist.at <float>(i));
if (pt2.y > 0)
rectangle(hist_img, pt1, pt2, Scalar(0), -1);
}
flip(hist_img, hist_img, 0);
}
void search_valueIdx(Mat hist, int &low_value, int &high_value)
{
int i;
for (i = 0; i < hist.rows; i++) {
if (hist.at<float>(i) > 0)
break;
```

```
}
low_value = i;

        for (i = hist.rows; i > 0; i--) {
if (hist.at<float>(i) > 0)
                break;
        }
        high_value = i;
}
int main()
{
        Mat image = imread("../image/Lenna-histo_str.tif", 0);
        CV_Assert(!image.empty());

        Mat hist, hist_dst, hist_img, hist_dst_img;
        int histsize = 256, ranges = 256;
        calc_Histo(image, hist, histsize, ranges);
        draw_histo(hist, hist_img);

        int low_value, high_value;
        search_valueIdx(hist, low_value, high_value);
        cout << "high_value = " << high_value << endl;
        cout << "low_value = " << low_value << endl;

        int d_value = high_value - low_value;
        Mat dst = (image - low_value) * (255.0 / d_value);

        calc_Histo(dst, hist_dst, histsize, ranges);
        draw_histo(hist_dst, hist_dst_img);

        imshow("Original Image", image);
```

```
    imshow("Hist. Stretch. Image", dst);
    imshow("Hist. of Origin", hist_img);
    imshow("Hist. of Stretch. Image", hist_dst_img);
    waitKey();
    return 0;
}
```

6.7 Gistogrammany deňlemek

- Giň gistogramma paýlanyşy bolan, emma kontrast paýlanyşy bir tarapa süýşirilen suratlar üçin degişlidir



[1-nji ädim] Gistogrammanyň hasaplanmasy

[2-nji ädim] Gistogrammanyň toplanan jemini hasaplamak

$$sum[i] = \sum_{j=0}^{i} hist[j]$$

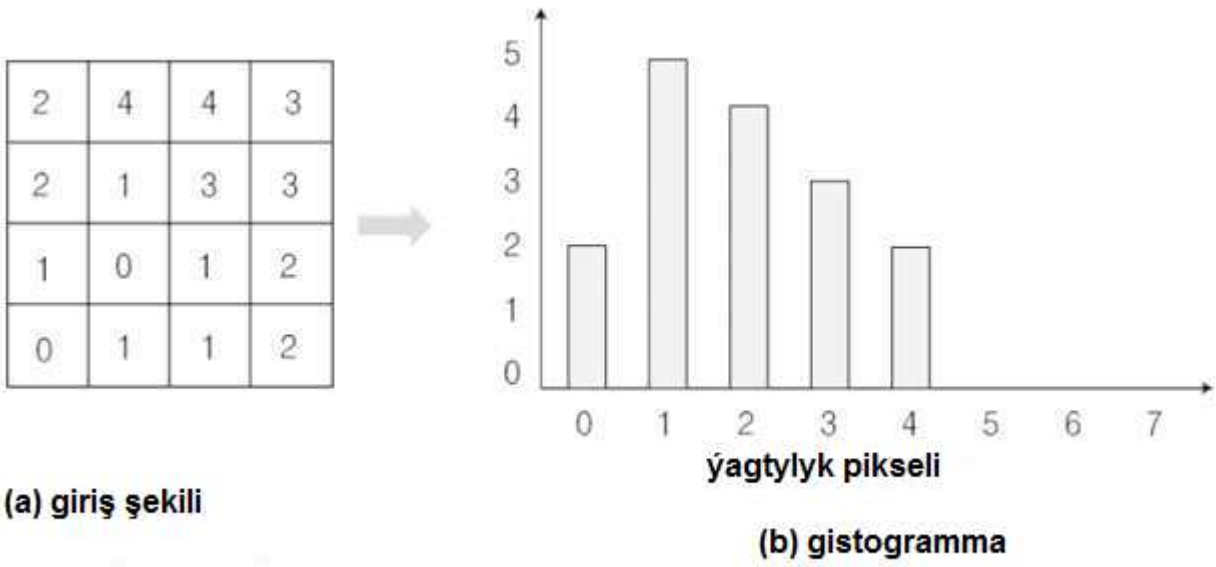
[3-nji ädim] Toplanan jemini̇ň kadalaşdyrylmagy

$$n[i] = sum[i] \times \frac{1}{N} \times I_{\max}$$

- N: Pikselleriň umumy sany
- I max: Iň ýokary piksel ýagtylyk derejesi

- **1-nji ädim**

- **Gistogrammanyň hasaplanmasy**

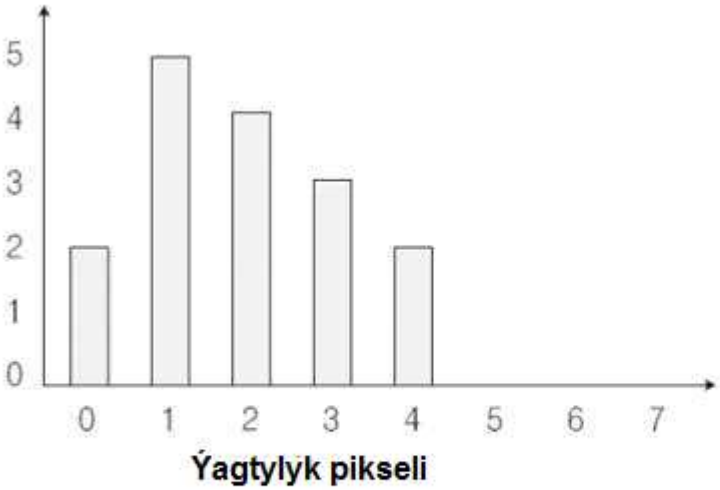


(a) giriş şekili

(b) gistogramma

- **2-nji ädim**

- **Gistogrammanyň toplanan jemini hasaplamak**



(a) Gistogramma

Bright.	Accum. Sum
0	2
1	7
2	11
3	14
4	16
5	16
6	16
7	16

(b) Toplanan jemi

- **3-nji ädim**

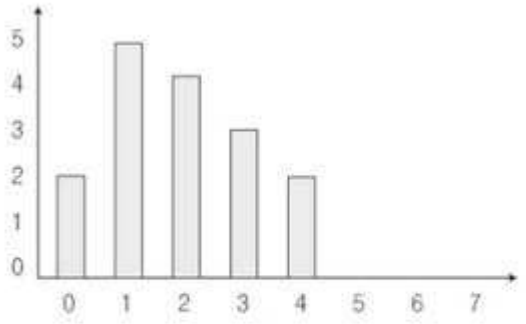
- **Toplanan jeminiň kadalaşdyrylmagy**
- **n[i]=sum[i]*(1/16)*7**

Bright. (i)	Accum. Sum (sum[i])	Normalized Accum. Sum (n[i])	
0	2	0.875	→1
1	7	3.0625	→3
2	11	4.8125	→5
3	14	6.125	→6
4	16	7	→7
5	16	7	→7
6	16	7	→7
7	16	7	→7

(a) Kadalaşdyrma

(b) Kadalaşdyrylan gistogramma

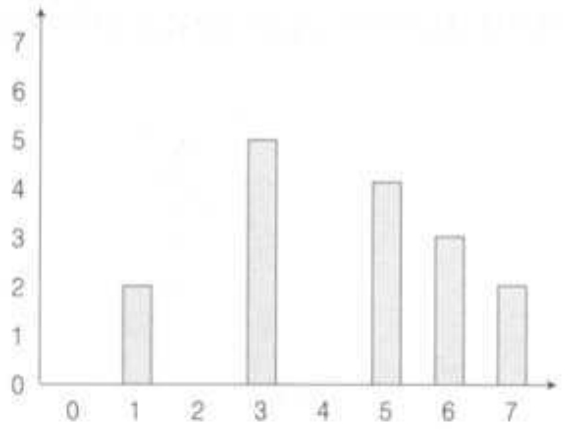
2	4	4	3
2	1	3	3
1	0	1	2
0	1	1	2



(a) Giriş şekili **(b) Gistogramma**

Asyl surat	Gist. EQ
0	1
1	3
2	5
3	6
4	7
5	7
6	7
7	7

5	7	7	6
5	3	6	6
3	1	3	5
1	3	3	5



(a) deňleşdirilen şekil **(b) gistogramma**

```
// Gistogrammanyň deňleşdirmek programmasy
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void calc_Histo(const Mat& image, Mat& hist, int bins, int range_max
        = 256)
{
        int histSize[] = { bins };
        float range[] = { 0, (float)range_max };
        int channels[] = { 0 };
        const float* ranges[] = { range };
        calcHist(&image, 1, channels, Mat(), hist, 1, histSize, ranges);
}

void draw_histo(Mat hist, Mat &hist_img, Size size = Size(256, 200))
{
        hist_img = Mat(size, CV_8U, Scalar(255));
        normalize(hist, hist, 0, hist_img.rows, NORM_MINMAX);
        for (int i = 0; i < hist.rows; i++) {
Point2f pt1 = Point2f(i, 0);
Point2f pt2 = Point2f((i+1), hist.at <float>(i));

if (pt2.y > 0)
        rectangle(hist_img, pt1, pt2, Scalar(0), -1);
        }
        flip(hist_img, hist_img, 0);
}
void create_hist(Mat img, Mat &hist, Mat &hist_img)
{
```

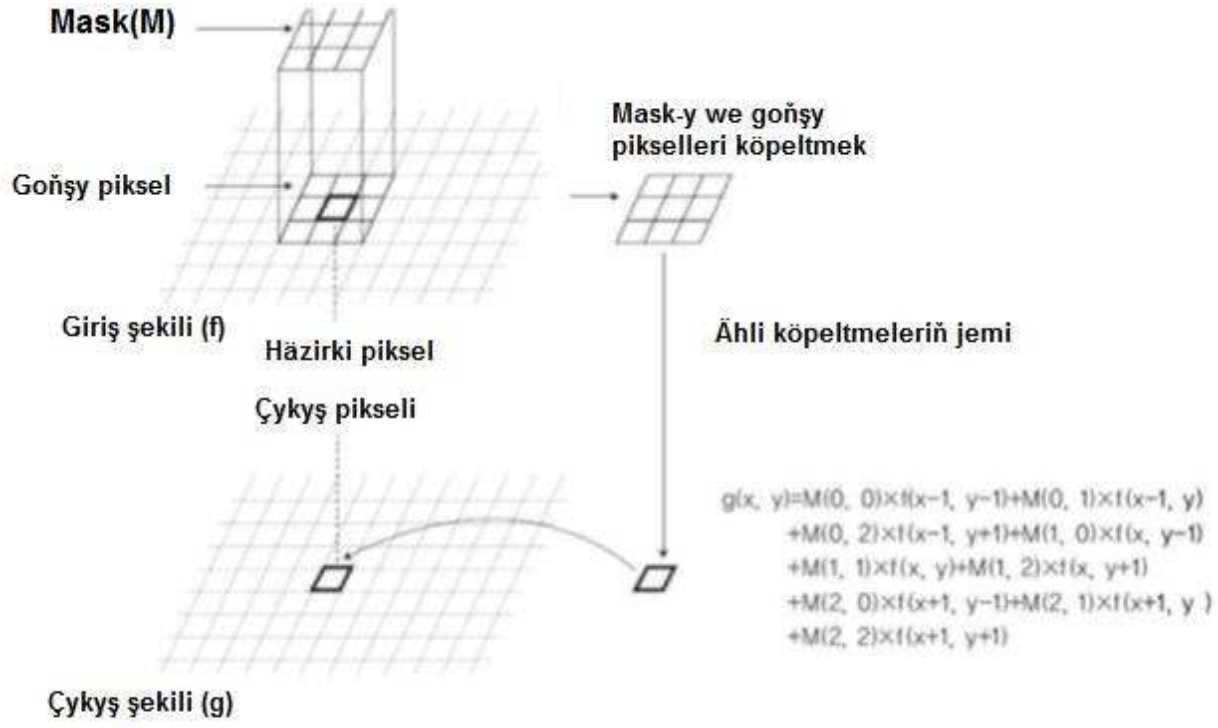
```
        int histsize = 256, range = 256;
        calc_Histo(img, hist, histsize, range);
        draw_histo(hist, hist_img);
}
int main()
{
        Mat image = imread("../image/Ave.tif", 0);
        CV_Assert(!image.empty());
        Mat hist, dst1, dst2, hist_img, hist_img1, hist_img2;

        create_hist(image, hist, hist_img);

        Mat accum_hist = Mat(hist.size(), hist.type(), Scalar(0));
        accum_hist.at<float>(0) = hist.at<float>(0);
        for (int i = 1; i < hist.rows; i++)
accum_hist.at<float>(i) = accum_hist.at<float>(i - 1) +
hist.at<float>(i);

        accum_hist /= sum(hist)[0];
        accum_hist *= 255;
        dst1 = Mat(image.size(), CV_8U);
        for (int i = 0; i < image.rows; i++) {
for (int j = 0; j < image.cols; j++) {
        int idx = image.at<uchar>(i, j);
        dst1.at<uchar>(i, j) = (uchar)accum_hist.at<float>(idx);
}
}
        create_hist(dst1, hist, hist_img1);
        // Using Histogram Equalization of OpenCV Function
        equalizeHist(image, dst2);
        create_hist(dst2, hist, hist_img2);
```

```
    imshow(“image", image), imshow("img_hist", hist_img);
    imshow("dst1-User", dst1), imshow("User_hist", hist_img1);
    imshow("dst2-OpenCV", dst2), imshow("OpenCV_hist", hist_img2);
    waitKey();
    return 0;
}
```

7. Açyk rezýume ulanyp dolamaklygy işläp taýýarlamak

7.1 Dolamaklygy işläp taýýarlamak

- Surat, pikseliň goňşy piksel bilen birleşmegi hem degişli bolmak bilen, dolamaklyk (konwolýasiýa) usuly bilen işläp taýýarlanýar.
- Mask ulanmak (= Ýadro, penjire, filter)



7.2 Näbellilik

- Piksel derejesiniň ýitiliginde ýuwaş-ýuwaşdan üýtgeşmeler girizmeli
- Bulaşyk Mask-ny ulanyp, dolamaklyk (konwolýasiýa) işläp taýýarlamak
- Bulaşyk Mask

$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$

$\frac{1}{25}$	$\frac{1}{25}$	$\frac{1}{25}$	$\frac{1}{25}$	$\frac{1}{25}$
$\frac{1}{25}$	$\frac{1}{25}$	$\frac{1}{25}$	$\frac{1}{25}$	$\frac{1}{25}$
$\frac{1}{25}$	$\frac{1}{25}$	$\frac{1}{25}$	$\frac{1}{25}$	$\frac{1}{25}$
$\frac{1}{25}$	$\frac{1}{25}$	$\frac{1}{25}$	$\frac{1}{25}$	$\frac{1}{25}$
$\frac{1}{25}$	$\frac{1}{25}$	$\frac{1}{25}$	$\frac{1}{25}$	$\frac{1}{25}$

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

90	90	90	90	90	90	90
90	90	90	90	90	90	90
90	90	255	255	255	90	90
90	90	255	255	255	90	90
90	90	255	255	255	90	90
90	90	90	90	90	90	90
90	90	90	90	90	90	90

(a) Giriş şekili

90	90	90	90	90	90	90
90	105	120	135	120	105	90
90	120	150	180	150	120	90
90	135	180	255	180	135	90
90	120	150	180	150	120	90
90	105	120	135	120	105	90
90	90	90	90	90	90	90

(b) Bulaşyk şekili

```
// Bulaşyk programmasy
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void filter(Mat img, Mat& dst, Mat mask)
{
		dst = Mat(img.size(), CV_32F, Scalar(0));

		// Surat giňeltmesi (3 x 3 Mask ulanyp)
		Mat ExtImg(img.rows+2, img.cols+2, CV_32F);
		for (int i = 0; i < img.rows; i++) {
	for (int j = 0; j < img.cols; j++)
		ExtImg.at<float>(i + 1, j + 1) = img.at<uchar>(i, j); // center
		}
		for (int i = 1; i < img.rows+1; i++) {
	ExtImg.at<float>(i, 0) = ExtImg.at<float>(i, 1); // Left line
	ExtImg.at<float>(i, img.cols + 1) = ExtImg.at<float>(i, img.cols); // Right
	line
		}
		for (int j = 1; j < img.cols+1; j++) {
	ExtImg.at<float>(0, j) = ExtImg.at<float>(1, j); // Ýokarky setir
	ExtImg.at<float>(img.rows + 1, j) = ExtImg.at<float>(img.rows, j); //
	Aşakdaky setir
		}
		ExtImg.at<float>(0, 0) = ExtImg.at<float>(1, 1); // Ýokarky çep burç
		ExtImg.at<float>(0, img.cols+1) = ExtImg.at<float>(1, img.cols); // Ýokarky
		sag burç
		ExtImg.at<float>(img.rows+1, 0) = ExtImg.at<float>(img.rows, 1); //
		Aşaky çep burç
		ExtImg.at<float>(img.rows+1, img.cols+1) = ExtImg.at<float>(img.rows,
```

```
            img.cols); // Aşaky sag

        for (int i = 0; i < img.rows; i++) {
for (int j = 0; j < img.cols; j++) {
        float sum = 0;
        for (int u = 0; u < mask.rows; u++) {
        for (int v = 0; v < mask.cols; v++)
                sum += ExtImg.at<float>(i + u, j + v) * mask.at<float>(u,
v);
                        }
                        dst.at<float>(i, j) = sum;
        }
                }
}
int main()
{
        Mat image = imread("../image/filter_blur.jpg", IMREAD_GRAYSCALE);
        CV_Assert(image.data);

                float data[] = {
1 / 9.f, 1 / 9.f, 1 / 9.f,
1 / 9.f, 1 / 9.f, 1 / 9.f,
1 / 9.f, 1 / 9.f, 1 / 9.f };

        Mat mask(3, 3, CV_32F, data), blur;
        filter(image, blur, mask);
        blur.convertTo(blur, CV_8U);

        namedWindow(“Original Image", WINDOW_AUTOSIZE);
        namedWindow(“Blurring", WINDOW_AUTOSIZE);
        moveWindow(“Original Image", 200, 200), moveWindow(“Blurring",
```

```
        600, 200);
        imshow(“Original image", image), imshow(“Blurring", blur);
        waitKey();
        return 0;
}
```

7.3 Ýitileşdirmek

- Goňşy pikselleriň arasyndaky tapawut, olary ýiti duýmagydyr
- Surat aýratynlyklaryna ünsi çekip bolýar we gyrada ýagtylygyň kontrastyny ýokarlandyryp bolýar
- Suratyň aýdyňlygyny ýitilendiriş Mask-y

0	−1	0
−1	5	−1
0	−1	0

−1	−1	−1
−1	9	−1
−1	−1	−1

1	−2	1
−2	5	−2
1	−2	1

0	−1	0
−1	5	−1
0	−1	0

10	10	10	10	10	10	10
10	10	10	10	10	10	10
10	10	50	50	50	10	10
10	10	50	50	50	10	10
10	10	50	50	50	10	10
10	10	10	10	10	10	10
10	10	10	10	10	10	10

(a) Giriş şekili

10	10	10	10	10	10	10
10	10	0	0	0	10	10
10	0	130	90	130	0	10
10	0	90	50	90	0	10
10	0	130	90	130	0	10
10	10	0	0	0	10	10
10	10	10	10	10	10	10

(b) Ýitilendirilen şekil

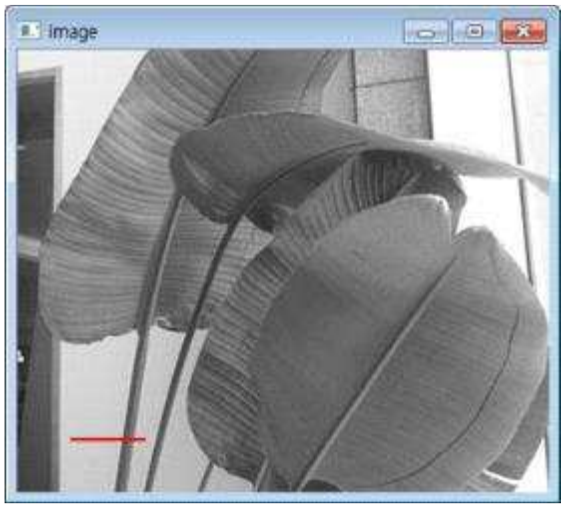
```
// Ýitilendiriş programmasy
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void filter(Mat img, Mat& dst, Mat mask)
{
	dst = Mat(img.size(), CV_32F, Scalar(0));

	// Surat giňeltmesi (3 x 3 Mask-y ulanyp)
	Mat ExtImg(img.rows+2, img.cols+2, CV_32F);
	for (int i = 0; i < img.rows; i++) {
for (int j = 0; j < img.cols; j++)
		ExtImg.at<float>(i + 1, j + 1) = img.at<uchar>(i, j); // merkez
	}
	for (int i = 1; i < img.rows+1; i++) {
ExtImg.at<float>(i, 0) = ExtImg.at<float>(i, 1); // Left line
ExtImg.at<float>(i, img.cols + 1) = ExtImg.at<float>(i, img.cols); //
Sag setir
	}
	for (int j = 1; j < img.cols+1; j++) {
ExtImg.at<float>(0, j) = ExtImg.at<float>(1, j); // Ýokarky setir
ExtImg.at<float>(img.rows + 1, j) = ExtImg.at<float>(img.rows, j); //
Aşaky setir
	}
	ExtImg.at<float>(0, 0) = ExtImg.at<float>(1, 1); // Ýokaryky çep burç

	ExtImg.at<float>(0, img.cols+1) = ExtImg.at<float>(1, img.cols); //
Ýokarky sag burç
	ExtImg.at<float>(img.rows+1, 0) = ExtImg.at<float>(img.rows, 1);
// Aşaky çep burç
```

```
        ExtImg.at<float>(img.rows+1, img.cols+1) =
ExtImg.at<float>(img.rows, img.cols); // Aşaky sag burç

        for (int i = 0; i < img.rows; i++) {
for (int j = 0; j < img.cols; j++) {
        float sum = 0;
        for (int u = 0; u < mask.rows; u++) {
        for (int v = 0; v < mask.cols; v++)
                        sum += ExtImg.at<float>(i + u, j + v) *
mask.at<float>(u, v);
        }
        dst.at<float>(i, j) = sum;
        }
                }
}
int main()
{
Mat image = imread("../image/filter_sharpen.jpg", IMREAD_GRAYSCALE);
CV_Assert(image.data);

        float data[] = {
-1, -1, -1,
-1, 9, -1,
-1, -1, -1 };

        Mat mask(3, 3, CV_32F, data), sharpen;
        filter(image, sharpen, mask);
        sharpen.convertTo(sharpen, CV_8U);

        namedWindow(“Original Image", WINDOW_AUTOSIZE);
        namedWindow(“Sharpening Image", WINDOW_AUTOSIZE);
```

```
        moveWindow("Original Image", 200, 200),
moveWindow("Sharpening Image", 600, 200);
        imshow("Original Image", image), imshow("Sharpening Image",
sharpen);
        waitKey();
        return 0;
}
```

7.4 Gyralary kesgitlemek

- Bir suratyň gyrasy, piksel derejesiniň birden üýtgeýän bölegi



7.5 Gyralary kesgitlemek (birmeňzeş / diferensial operator)

- Birmeňzeş operator

- merkezi pikseliň tapawudyny hasaplamak

- Differensial operator

- merkezi piksel we ters tarapdaky piksel we 4 diferensial hasaplamalar

m_0	m_1	m_2
m_3	C	m_5
m_6	m_7	m_8

Birmeňzeş operatoryň çykyşy =

$$\max(|c-m_0|, |c-m_1|, |c-m_2|, |c-m_3| \\ |c-m_5|, |c-m_6|, |c-m_7|, |c-m_8|)$$

Differensial operatoryň çykyşy =

$$\max(|m_0-m_8|, |m_1-m_7|, |m_2-m_6|, |m_3-m_5|$$

- **Gyralary kesgitlemek (Birmeňzeş operator)**

```
// Gyralary kesgitlemegiň (birmeňzeş operator) programmasy
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void homogenOp(Mat img, Mat& dst, int mask_size)
{
	dst = Mat(img.size(), CV_8U, Scalar(0));

	// Surat giňeltmesi (3 x 3 Mask-y ulanyp)
	Mat ExtImg(img.rows + 2, img.cols + 2, CV_32F);
	for (int i = 0; i < img.rows; i++) {
		for (int j = 0; j < img.cols; j++)
```

```
                ExtImg.at<float>(i + 1, j + 1) = img.at<uchar>(i, j);
}
for (int i = 1; i < img.rows + 1; i++) {
        ExtImg.at<float>(i, 0) = ExtImg.at<float>(i, 1);
        ExtImg.at<float>(i, img.cols + 1) = ExtImg.at<float>(i, img.cols);
}
for (int j = 1; j < img.cols + 1; j++) {
        ExtImg.at<float>(0, j) = ExtImg.at<float>(1, j);
        ExtImg.at<float>(img.rows + 1, j) = ExtImg.at<float>(img.rows, j);
}
ExtImg.at<float>(0, 0) = ExtImg.at<float>(1, 1);
ExtImg.at<float>(0, img.cols + 1) = ExtImg.at<float>(1, img.cols);
ExtImg.at<float>(img.rows + 1, 0) = ExtImg.at<float>(img.rows, 1);
ExtImg.at<float>(img.rows+1,img.cols+1)=ExtImg.at<float>(img.rows, img.cols);

for (int i = 0; i < img.rows; i++) {
        for (int j = 0; j < img.cols; j++) {
                float max = 0;
                for (int u = 0; u < mask_size; u++) {
                        for (int v = 0; v < mask_size; v++) {
                                float difference=abs(ExtImg.at<float>(i+1, j+1) -
ExtImg.at<float>(i+u, j+v));
                                if (difference > max)
                                        max = difference;
                }
        }
        dst.at<uchar>(i, j) = max;
        }
        }
}
int main()
```

```
{
	Mat image = imread("../image/edge_test.jpg", IMREAD_GRAYSCALE);
	CV_Assert(image.data);
	Mat edge;
	homogenOp(image, edge, 3);
	namedWindow(“Original Image", WINDOW_AUTOSIZE);
	namedWindow(“Homogen. Edge", WINDOW_AUTOSIZE);
	moveWindow(“Original Image", 200, 200);
	moveWindow("Homogen. Edge", 600, 200);
	imshow(“Original Image", image);
	imshow("Homogen. Edge", edge);
	waitKey();
	return 0;
}
```

- **Gyralary kesgitlemek (Differensial operator)**

```
// Gyralary kesgitlemegiň (differensial operator) programmasy
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void homogenOp(Mat img, Mat& dst, int mask_size)
{
	dst = Mat(img.size(), CV_8U, Scalar(0));

	// Surat giňeltmesi (3 x 3 Mask-y ulanyp)
	Mat ExtImg(img.rows + 2, img.cols + 2, CV_32F);
	for (int i = 0; i < img.rows; i++) {
	for (int j = 0; j < img.cols; j++)
		ExtImg.at<float>(i + 1, j + 1) = img.at<uchar>(i, j);
```

```
}
for (int i = 1; i < img.rows + 1; i++) {
        xtImg.at<float>(i, 0) = ExtImg.at<float>(i, 1);
        ExtImg.at<float>(i, img.cols + 1) = ExtImg.at<float>(i, img.cols);
}
for (int j = 1; j < img.cols + 1; j++) {
        ExtImg.at<float>(0, j) = ExtImg.at<float>(1, j);
        ExtImg.at<float>(img.rows + 1, j) = ExtImg.at<float>(img.rows, j);
}
        ExtImg.at<float>(0, 0) = ExtImg.at<float>(1, 1);
        ExtImg.at<float>(0, img.cols + 1) = ExtImg.at<float>(1, img.cols);
        ExtImg.at<float>(img.rows + 1, 0) = ExtImg.at<float>(img.rows, 1);
        ExtImg.at<float>(img.rows+1, img.cols + 1) = ExtImg.at<float>(img.rows,
        img.cols);

for (int i = 0; i < img.rows; i++) {
        for (int j = 0; j < img.cols; j++) {
                vector<float> mask;
                for (int u = 0; u < mask_size; u++) {
                        for (int v = 0; v < mask_size; v++)
                                mask.push_back(ExtImg.at<float>(i + u, j + v));
                        }
                        float max = 0;
                        for (int k = 0; k <= mask_size; k++) {
                                float difference = abs(mask[k]-mask[8-k]);
                                if (difference > max) max = difference;
                        }
                        dst.at<uchar>(i, j) = max;
                }
        }
}
```

```
int main()
{
	Mat image = imread("../image/edge_test.jpg", IMREAD_GRAYSCALE);
	CV_Assert(image.data);

	Mat edge;
	differOp(image, edge, 3);

	namedWindow(“Original Image", WINDOW_AUTOSIZE);
	namedWindow(“Diff. Edge", WINDOW_AUTOSIZE);
	moveWindow("Original Image", 200, 200), moveWindow("Diff. Edge", 600, 200);
	imshow("Original Image", image), imshow("Diff. Edge", edge);
	waitKey();
	return 0;
}
```

7.6 Gyralary kesgitlemek (Roberts / Sobel Mask operatory)

- **Roberts Mask operatory**

$G_x =$

−1	0	0
0	1	0
0	0	0

$G_y =$

0	0	−1
0	1	0
0	0	0

- **Sobel Mask operatory**

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad G_y = \begin{bmatrix} 1 & -2 & 1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

- **Gyralary kesgitlemek (Roberts Mask operatory)**

```
// Gyralary kesgitlemegiň (Roberts Mask operatory) programmasy
// OpenCV-iň filter2D funksiýasyny ulanmak
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void differential(Mat image, Mat& dst, float data1[], float data2[])
{
Mat dst1, mask1(3, 3, CV_32F, data1);
Mat dst2, mask2(3, 3, CV_32F, data2);

filter2D(image, dst1, CV_32F, mask1, Point(-1, -1), 0, BORDER_REPLICATE);
filter2D(image, dst2, CV_32F, mask2, Point(-1, -1), 0, BORDER_REPLICATE);
                // Point: ýadronyň filterleme başlangyç nokady, 0: netijä goşulan dereje
                // BORDER_REPLICATE: çägi köpeltmek arkaly şekil giňeltmesi

        magnitude(dst1, dst2, dst);
        dst.convertTo(dst, CV_8U);

        dst1 = abs(dst1);
        dst2 = abs(dst2);
```

```
    dst1.convertTo(dst1, CV_8U);
    dst2.convertTo(dst2, CV_8U);
    namedWindow(“Mask 1 Result", WINDOW_AUTOSIZE);
    namedWindow("Mask 2 Result", WINDOW_AUTOSIZE);
    moveWindow("Mask 1 Result", 200, 500);
    moveWindow("Mask 2 Result", 700, 500);
    imshow("Mask 1 Result", dst1);
    imshow("Mask 2 Result", dst2);
}
int main()
{
    Mat image = imread("../image/edge_test1.jpg", IMREAD_GRAYSCALE);
    CV_Assert(image.data);

    float data1[] = {
        -1, 0, 0,
        0, 1, 0,
        0, 0, 0
    };
    float data2[] = {
        0, 0, -1,
        0, 1, 0,
        0, 0, 0
    };
    Mat dst;
    differential(image, dst, data1, data2);

    namedWindow(“Original Image", WINDOW_AUTOSIZE);
    namedWindow(“Roberts Edge", WINDOW_AUTOSIZE);
    moveWindow("Original Image", 200, 100);
    moveWindow("Roberts Edge", 700, 100);
```

```
    imshow("Original Image”, image);
    imshow("Roberts Edge", dst);
    waitKey();
    return 0;
}
```

- **Gyralary kesgitlemek (Sebel Mask operatory)**

```
// Gyralary kesgitlemegiň (Sobel Mask operatory) programmasy
// OpenCV-iň filter2D we Sobel funksiýasyny ulanmak
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void differential(Mat image, Mat& dst, float data1[], float data2[])
{
    Mat dst1, mask1(3, 3, CV_32F, data1);
    Mat dst2, mask2(3, 3, CV_32F, data2);

    filter2D(image, dst1, CV_32F, mask1, Point(-1, -1), 0, BORDER_REPLICATE);
    filter2D(image, dst2, CV_32F, mask2, Point(-1, -1), 0, BORDER_REPLICATE);
    magnitude(dst1, dst2, dst);
    dst.convertTo(dst, CV_8U);

    convertScaleAbs(dst1, dst1);
    convertScaleAbs(dst2, dst2);
    namedWindow("dst1 – Vertical Mask", WINDOW_AUTOSIZE);
    namedWindow("dst2 – Horiz. Mask", WINDOW_AUTOSIZE);
    moveWindow("dst1 – Vertical Mask", 700, 100);
    moveWindow("dst2 – Horiz. Mask", 700, 500);
    imshow("dst1 – Vertical Mask", dst1);
```

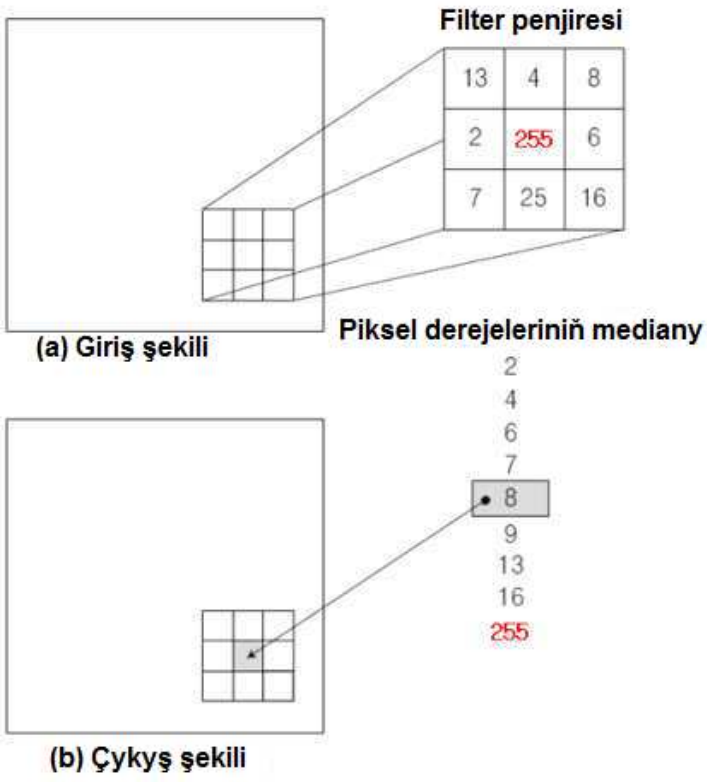
```
	imshow("dst2 – Horiz. Mask", dst2);
}
int main()
{
	Mat image = imread("../image/edge_test1.jpg", IMREAD_GRAYSCALE);
	CV_Assert(image.data);
	float data1[] = {
		-1, 0, 1,
		-2, 0, 2,
		-1, 0, 1 };
	float data2[] = {
		-1, -2, -1,
		0, 0, 0,
		1, 2, 1 };
	Mat dst, dst3, dst4;
	differential(image, dst, data1, data2);
	namedWindow(“Original Image", WINDOW_AUTOSIZE);
	namedWindow(“Sobel Edge", WINDOW_AUTOSIZE);
	moveWindow("Original Image", 200, 100);
	moveWindow("Sobel Edge", 200, 500);
	imshow("Original Image", image);
	imshow("Sobel Edge", dst);

	// OpenCV funksiýasyny ulanyp, Sobel gyrasyny kesgitlemek
	Sobel(image, dst3, CV_32F, 1, 0, 3); // 4th = 1 -> vertical mask (vertical
edge)
	Sobel(image, dst4, CV_32F, 0, 1, 3); // 5th = 1 -> horizontal mask
(horizontal edge)
	convertScaleAbs(dst3, dst3); // 3 -> kernal size
	convertScaleAbs(dst4, dst4);
```

```
    namedWindow("dst3 – vert._OpenCV", WINDOW_AUTOSIZE);
    namedWindow("dst4 – horiz._OpenCV", WINDOW_AUTOSIZE);
    moveWindow("dst3 – vert._OpenCV", 1200, 100);
    moveWindow("dst4 – horiz._OpenCV", 1200, 500);
    imshow("dst3 – vert.OpenCV", dst3), imshow("dst4 – horiz._OpenCV", dst4);
    waitKey();
    return 0;
}
```

7.7 Median filteri

- **Median filteri**

- Goňşy piksel derejelerini ýokarlanýan tertipde deňleşdireniňizden soň, çykyş derejesi hökmünde merkezi derejesini saýlaň
- Suratdaky uçgun ýaly duýdansyz reňk üýtgeşmesine eýe bolan impuls sesini aýyrmak üçin ulanylýar



```
// Median filterleme programmasy
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

void medianFilter(Mat img, Mat& dst, int size)
{
	dst = Mat(img.size(), CV_8U, Scalar(0));

	// Surat giňeltmesi (3 x 3 Mask-y ulanyp)
	Mat ExtImg(img.rows + 2, img.cols + 2, CV_32F);
	for (int i = 0; i < img.rows; i++) {
	for (int j = 0; j < img.cols; j++)
		ExtImg.at<float>(i + 1, j + 1) = img.at<uchar>(i, j);
	}
	for (int i = 1; i < img.rows + 1; i++) {
	ExtImg.at<float>(i, 0) = ExtImg.at<float>(i, 1);
	ExtImg.at<float>(i, img.cols + 1) = ExtImg.at<float>(i, img.cols);
	}
	for (int j = 1; j < img.cols + 1; j++) {
	ExtImg.at<float>(0, j) = ExtImg.at<float>(1, j);
	ExtImg.at<float>(img.rows + 1, j) = ExtImg.at<float>(img.rows, j);
	}
	ExtImg.at<float>(0, 0) = ExtImg.at<float>(1, 1);
	ExtImg.at<float>(0, img.cols + 1) = ExtImg.at<float>(1, img.cols);
	ExtImg.at<float>(img.rows + 1, 0) = ExtImg.at<float>(img.rows, 1);
	ExtImg.at<float>(img.rows + 1, img.cols + 1) =
ExtImg.at<float>(img.rows, img.cols);
	for (int i = 0; i < img.rows; i++) {
	for (int j = 0; j < img.cols; j++) {
```

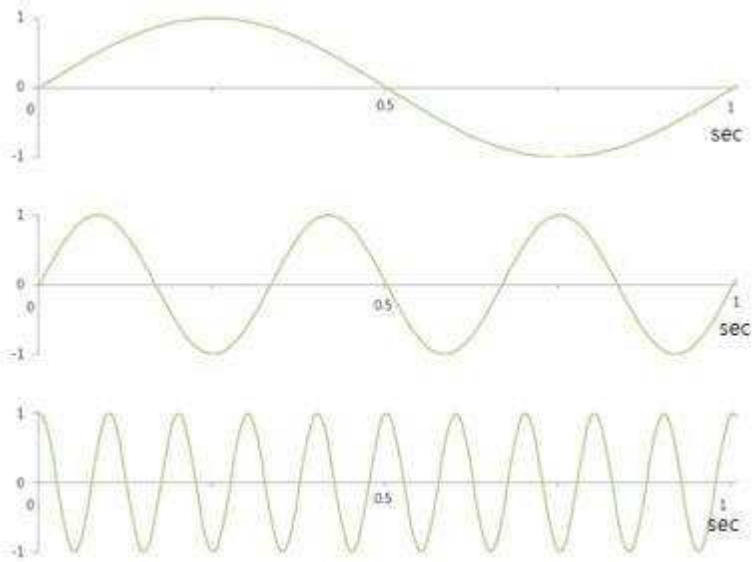
```
            vector<float> mask;
            for (int u = 0; u < size; u++) {
            for (int v = 0; v < size; v++)
                mask.push_back(ExtImg.at<float>(i + u, j + v));
            }
                cv::sort(mask, mask, SORT_EVERY_ROW); // cv:: writing to
distinguish from std::sort()
            dst.at<uchar>(i, j) = (uchar)mask[4];
        }
            }
}
int main()
{
    Mat gray = imread("../image/Boat-noise.tif", IMREAD_GRAYSCALE);
    CV_Assert(gray.data);
    Mat med_img1, med_img2;
    namedWindow("Original Image", WINDOW_AUTOSIZE);
    moveWindow("Original Image", 200, 200);
    imshow("Original Image", gray);
    medianFilter(gray, med_img1, 3); // Median Filtering using User Function
    namedWindow("median-User", WINDOW_AUTOSIZE);
    moveWindow("median-User", 600, 200);
    imshow("median-User", med_img1);
    medianBlur(gray, med_img2, 3); // Median Filtering using OpenCV Function
    namedWindow("median-OpenCV", WINDOW_AUTOSIZE);
    moveWindow("median-OpenCV", 1000, 200);
    imshow("median-OpenCV", med_img2);
    waitKey();
    return 0;
}
```

8. Domen işleýişini üýtgetmek

8.1 Giňişlik ýygylygy

- Gerts (Hz)
- Ýygylyk birligi
- 1 sekuntda yrgyldynyň sany

- Suraty işläp taýýarlamak, giňişlik ýygylygy düşünjesini ulanýar
- Bir pikseliň ýagtylyk derejesini kesgitlemek

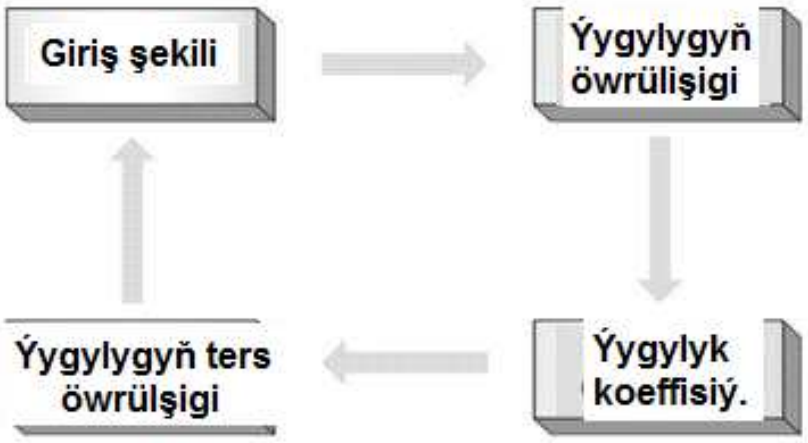


- Giňişligiň ýygylygy
- Pes ýygylykly giňişligiň meýdany
- Pikseliň ýagtylygynyň kän bir üýtgemeýän ýa-da kem-kemden üýtgeýän giňişliginde
- Suratlaryň fon bölegi
- Ýokary ýygylyly giňişlik meýdany
- Pikseliň ýagtylygy çalt üýtgeýär
- Suratlaryň gyra bölegi

- Suratlary ýygylyk domeni boýunça bölýän bolsaňyz näme etmeli?

- Ýokary ýygylykly komponentler bilen aýrylan surat ☞ Bulaşyk gyralary bolan surat
- Diňe ýokary ýygylykly komponentler bilen düşürilen surat ☞ Diňe bir gyrasy bolan surat, ýagny gyra şekili

- Ýygylygy öwürmek



8.2 Fýurýeriň üýtgemegi

- Giňişli domeni → ýygylyk domeni

- suratyň in, beýiklik ölçegi: N

$$F(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y)\exp\left[\frac{-j2\pi(ux+vy)}{N}\right]$$

$$for\ \ u,v=0,\ 1,\ 2,\ ....,\ N-1$$

- Ýygylyk domeni → giňişlik domeni

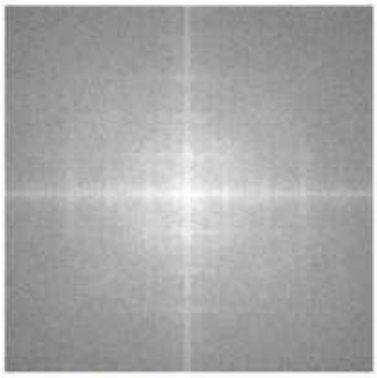
- suratyň in, beýiklik ölçegi: N

$$f(x,y)=\frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}F(u,v)\exp\left[\frac{j2\pi(ux+vy)}{N}\right]$$

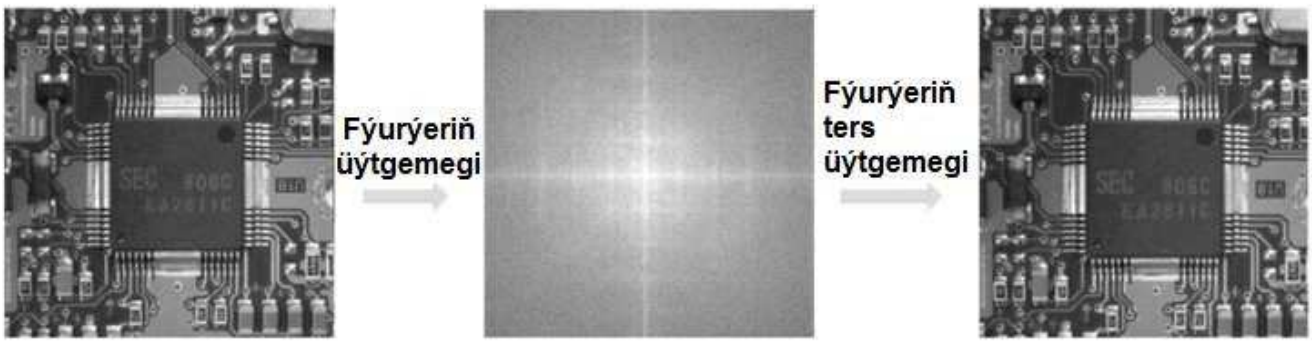
$$for\ \ x,y=0,\ 1,\ 2,\ ....,\ N-1$$

(a) Giriş şekili

(b) ýygylyk şekili



(c) Üýtgedilen ýygylyk şekili



8.3 Diskret kosinus öwrülişigi (DCT)

- Giňişli domeni → ýygylyk domeni

- suratyň in, beýiklik ölçegi: N

$$C(u,v)=\alpha(u)\alpha(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y)\cos\left[\frac{(2x+1)u\pi}{2N}\right]\cos\left[\frac{(2y+1)v\pi}{2N}\right]$$

$$\alpha(u)=\begin{cases}\sqrt{\frac{1}{N}} & for \quad u=0\\ \sqrt{\frac{2}{N}} & for \quad u=1,2,3,\cdots,N-1\end{cases}$$

- Ýygylyk domeni → giňişlik domeni

- suratyň in, beýiklik ölçegi: N

$$f(x,y)=\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}\alpha(u)\alpha(v)C(u,v)\cos\left[\frac{(2x+1)u\pi}{2N}\right]\cos\left[\frac{(2y+1)v\pi}{2N}\right]$$

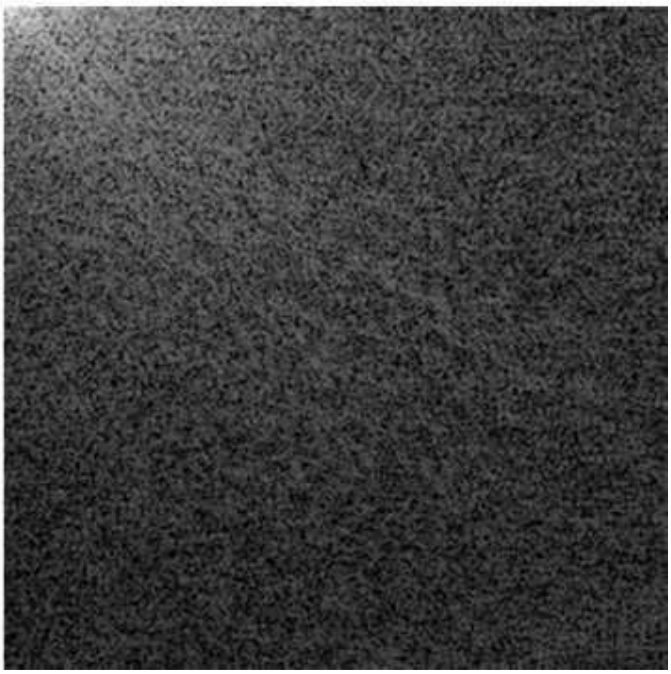
162	162	162	163	164	161	155	159	157	161	153	154	154	155	156	154	153
162	162	162	163	164	161	155	159	157	161	153	154	154	155	156	154	153
162	162	162	163	164	161	155	159	157	161	153	154	154	155	156	154	153
160	163	160	159	159	156	155	156	153	154	158	155	154	152	151	152	158
155	157	159	157	163	158	158	155	155	161	156	155	157	151	152	156	154
155	155	156	155	157	157	159	157	155	153	158	156	151	153	155	150	154
157	157	156	158	159	158	155	156	156	161	156	157	150	153	152	151	155
157	158	154	157	155	151	156	155	154	154	152	154	153	152	154	155	160
156	155	157	157	154	153	158	154	152	154	155	153	154	150	150	164	161
157	159	157	154	157	153	157	160	158	158	155	154	154	154	160	161	165
156	154	155	156	157	158	157	159	157	155	153	159	154	154	158	160	162
158	155	153	156	156	152	157	158	159	157	157	156	149	154	157	161	167
156	156	152	158	159	155	158	157	159	155	157	159	153	158	162	163	166
152	153	153	155	157	155	159	159	158	155	158	155	154	160	160	165	165
158	157	161	157	156	156	159	157	158	155	159	159	157	163	165	169	167
156	157	158	156	158	159	158	163	159	158	158	161	162	166	167	168	167

8.4 Lenna 16x16 blogyň DCT koeffisiýentleri

- Lenna 16x16 blogyň DCT koeffisiýentleri Üýtgeme=128

-1150	-2614	-594	2474	2072	-1178	-314	991	-538	-1117	-1599	-1925	-1269	-620	-925	-459
1793	1963	-1634	-579	-581	847	354	-526	1134	135	1113	350	86	-238	-11	296
-44	750	-1456	1991	-1413	2531	30	-308	469	247	-509	301	596	-650	458	259
-427	-422	1939	-1436	331	841	-1690	791	-21	-734	399	-22	287	548	-196	18
-619	519	542	598	-68	321	-110	312	-744	1494	-158	-292	795	-724	674	-476
-63	-179	1272	90	-1120	-368	920	1311	-888	612	-955	572	-464	343	178	-49
728	-512	617	-547	-344	42	974	-958	435	-383	-76	579	-845	205	-319	236
869	-664	-425	561	-100	294	293	-777	-248	223	447	-184	169	303	-518	178
-125	86	24	-176	338	-297	-424	-147	656	-333	365	-212	-699	491	-316	298
120	-362	50	-144	478	-691	59	343	27	-637	119	-434	555	-24	-211	14
-299	279	-262	-18	368	-685	376	315	-324	-413	412	181	34	-127	-311	475
-444	105	329	-500	46	-145	0	-284	148	-150	-80	-55	-153	-374	570	-35
178	112	-156	3	-103	195	178	-283	176	174	-64	-129	69	198	-191	-244
7	-71	-292	259	150	-160	-234	-157	290	48	-278	159	131	-130	-233	436
440	-315	-68	-54	61	119	-134	-176	357	-365	-74	123	118	-206	13	250
-98	225	2	96	-43	160	78	123	-3	9	119	10	-75	370	179	-120

$$D(u,v) = |C(u,v)| \qquad D(u,v) = const \cdot \log\left[1 + |C(u,v)|\right]$$



8.5 Diskret kosinus öwrülişigi (DCT)

```
// Diskret kosinus öwrülişiginiň (DCT) programmasy
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

Mat DCT_block(Mat g) // Forward DCT
{
	Mat dst(g.size(), g.type());
	int N = g.rows, M = g.cols;

	for (int k = 0; k < N; k++) {
	for (int l = 0; l < M; l++) {

		float sum = 0;
		for (int n = 0; n < N; n++) {
		for (int m = 0; m < M; m++) {
			float theta1 = (float)((2 * n + 1) * k * CV_PI / (2 * N));
			float theta2 = (float)((2 * m + 1) * l * CV_PI / (2 * M));
			sum += g.at<float>(n, m) * cos(theta1) * cos(theta2);
		}
		}
		float ck = (k) ? sqrt(2.0f / N) : sqrt(1.0f / N);
		float cl = (l) ? sqrt(2.0f / M) : sqrt(1.0f / M);
		dst.at<float>(k, l) = ck * cl * sum;
	}
	}
	return dst;
}
Mat IDCT_block(Mat f) // Inverse DCT
```

```
{
	Mat dst(f.size(), f.type());
	int N = f.rows, M = f.cols;

	for (int n = 0; n < N; n++) {
	for (int m = 0; m < M; m++) {
		float sum = 0;
		for (int k = 0; k < N; k++) {
		for (int l = 0; l < M; l++) {
		float theta1 = (float)((2 * n + 1) * k * CV_PI / (2 * N));
		float theta2 = (float)((2 * m + 1) * l * CV_PI / (2 * M));
		float ck = (k) ? sqrt(2.0f / N) : sqrt(1.0f / N);
		float cl = (l) ? sqrt(2.0f / M) : sqrt(1.0f / M);
		sum += ck * cl * f.at<float>(k, l) * cos(theta1) * cos(theta2);
	}
	}
	dst.at<float>(n, m) = sum;
	}
	}
	return dst;

	}

	void DCT_2D(Mat img, Mat& dst, int N, int M, int dir)
	{
		dst = Mat(img.size(), CV_32F);
		img.convertTo(img, CV_32F);

		for (int bi = 0; bi < img.rows; bi += N) {
		for (int bj = 0; bj < img.cols; bj += M) {
			Rect rect(Point(bj, bi), Size(M, N));
```

```
                Mat block = img(rect);
                Mat new_block = (dir == 0) ? DCT_block(block) : IDCT_block(block);
                new_block.copyTo(dst(rect));
            }
            }
        }
        int main()
        {
            Mat image = imread("../image/dct_test1.jpg", IMREAD_GRAYSCALE);

        CV_Assert(image.data);
        Mat m_dct, m_idct;
        DCT_2D(image, m_dct, 8, 8, 0);
        DCT_2D(m_dct, m_idct, 8, 8, 1);
        m_idct.convertTo(m_idct, CV_8U);
        namedWindow("Original Image", WINDOW_AUTOSIZE);
        namedWindow("Inverse DCT", WINDOW_AUTOSIZE);
        moveWindow("Original Image", 200, 200), moveWindow("Inverse DCT",
600, 200);
        imshow("Original Image", image), imshow("Inverse DCT", m_idct);

        Rect rect(0, 0, 8, 8);
        cout << “First 8x8 Block Original Image Elements" << endl;
        cout << image(rect) << endl << endl;
        cout << “First 8x8 Block DCT Coefficients" << endl;
        cout << m_dct(rect) << endl;
        waitKey();
}
```