**Одеський національний морський університет**

Кораблебудівний факультет

Кафедра «Технічна кібернетика»

**Пояснювальна записка**

до дипломної роботи

\_\_\_\_\_\_\_\_\_\_\_\_\_Магістр\_\_\_\_\_\_\_\_\_\_\_\_

(освітньо-кваліфікаційний рівень)

на тему

IOT: Виділення лімфоцитів на зображеннях мазка крові з використанням OpenCV

Спеціальність:

8.05010101 «Інформаційні

управляючі системи та технології»

Виконав: студент 2 курсу, 4 групи

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Вєрсанов К.О.

(підпис) (прізвище та ініціали)

Керівник к.т.н., професор

(вчене звання, посада)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Челабчі В.М.

(підпис) (прізвище та ініціали)

Рецензент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(вчене звання, посада)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

Рецензент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(вчене звання, посада)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

Одеса - 2016

АННОТАЦИЯ

Разработан программный продукт для выявления клеток лимфоцитов в мазке крови для электронного микроскопа.

Программный продукт реализован с помощью библиотеки компьютерного зрения OpenCV и языка программирования С++.

Разработаны архитектура приложения и алгоритмы поиска и детектирования клеток лимфоцитов в мазке крови, представлен готовый программный продукт.

Программный продукт могут использовать любые реализации электронных микроскопов, он позволяет автоматизировать и ускорить поиск клеток, независимо от качества мазка крови.

АНОТАЦІЯ

Розроблено програмний продукт для виявлення клітин лімфоцитів в мазку крові для електронного мікроскопа.

Програмний продукт реалізований за допомогою бібліотеки комп'ютерного зору OpenCV і мови програмування С++.

Розроблено архітектура програми та алгоритми пошуку і детектування клітин лімфоцитів в мазку крові, представлений готовий програмний продукт.

Програмний продукт можуть використовувати будь які реалізації електронних мікроскопів, він дозволяє автоматизувати і прискорити пошук клітин, незалежно від якості мазка крові.

ANNOTATION

A software product for identifying cells lymphocytes in a smear of blood for the virtual microscopy.

The software product is implemented using computer vision library OpenCV and C ++ programming language.

The architecture and algorithms of the search application and detect cell of lymphocytes in the blood smear, presented the finished software product.

The software product can use within any implementation of electron microscopes, it allows you to automate and speed up the search for cells, regardless of the quality of blood smear.

ЗМІСТ

[ВВЕДЕННЯ 9](#_Toc442960988)

[1 АНАЛІТИЧНИЙ ОГЛЯД 10](#_Toc442960989)

[1.1 Бібліотека комп’ютерного зору OpenCV 10](#_Toc442960990)

[1.2 Мова програмування С++ та середовище MS Visual Studio 2015 16](#_Toc442960991)

[1.2.1 Стандартна бібліотека шаблонів STL 18](#_Toc442960992)

[1.2.2 Об'єктно-орієнтоване програмування 19](#_Toc442960993)

[1.3 Кольорова модель HSV 21](#_Toc442960994)

[1.4 Технологія Ruspberry pi 22](#_Toc442960995)

[2 ПОСТАНОВКА ЗАВДАННЯ 24](#_Toc442960996)

[2.1 Об'єкт дослідження 25](#_Toc442960997)

[2.2 Інтернет речей 27](#_Toc442960998)

[2.3 Вимоги до розробки 28](#_Toc442960999)

[3 РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ 29](#_Toc442961000)

[3.1 Призначення програмного продукту та мета разробки 29](#_Toc442961001)

[3.2 Об'єкт реалізації 29](#_Toc442961002)

[3.3 Цифрова патологія. 30](#_Toc442961003)

[3.4 Кроковий двигун 28BYJ-48 32](#_Toc442961004)

[3.5 Алгоритми роботи програми 35](#_Toc442961005)

[3.5.1 Алгоритм RGB – HSV зображення 39](#_Toc442961006)

[3.5.2 Алгоритм знаходження контура клітин та площі 40](#_Toc442961007)

[4 ОПИС ГОТОВОГО ПРОГРАМНОГО ПРОДУКТА 42](#_Toc442961008)

[4.1 Головний інтерфейс програми 42](#_Toc442961009)

[4.2 Сегментація зображення 46](#_Toc442961010)

[4.3 Виявлення контурів 49](#_Toc442961011)

[4.3.1 Інтерфейс програми для визначення HSV параметрів 52](#_Toc442961012)

[4.4 Вимоги до програмного і апаратного забезпечення 60](#_Toc442961013)

[5 ТЕСТУВАННЯ 61](#_Toc442961014)

[5.1 Типи помилок 61](#_Toc442961015)

[5.2 Етапи налагодження 62](#_Toc442961016)

[6 ОХОРОНА ПРАЦІ 64](#_Toc442961017)

[6.1 Вимоги до організації робочого місця з обслуговування, ремонту та налагодження ЕОМ 64](#_Toc442961018)

[6.2 Раціональне та профілактичне харчування 66](#_Toc442961019)

[6.3 Міжнародні нормативні документи з охорони праці користувачів комп'ютерів 67](#_Toc442961020)

[6.4. Ефективні принципи професійного відбору операторів ВДТ 69](#_Toc442961021)

[7 ЕКОНОМІЧНЕ ОБГРУНТУВАННЯ ПРОГРАМНОГО ПРОДУКТУ 74](#_Toc442961022)

[7.1 Загальна характеристика проекту 74](#_Toc442961023)

[7.2 Розрахунок трудомісткості 75](#_Toc442961024)

[7.3 Визначення ціни програмного продукту 80](#_Toc442961025)

[8 ЦИВІЛЬНИЙ ЗАХИСТ 89](#_Toc442961026)

[8.1 Методика визначення величини негативного впливу надзвичайної ситуації на людину та довкілля 90](#_Toc442961027)

[8.2 Особливості структури системи управління ризиками 91](#_Toc442961028)

[8.3 Методика прогностичної оцінки небезпеки об'єкту економіки (ОЕ) (адміністративно-територіальної одиниці) 92](#_Toc442961029)

[8.4 Оцінка параметрів та порядок визначення коефіцієнта небезпеки об'єкту економіки 96](#_Toc442961030)

[ВИСНОВКИ 99](#_Toc442961031)

[ПЕРЕЛІК ПОСИЛАНЬ 100](#_Toc442961032)

[ДОДАТОК А 102](#_Toc442961033)

[А1. Лістинг модуля lympho-catcher.cpp 102](#_Toc442961034)

[А2. Лістинг модуля HSVFinder.cpp 105](#_Toc442961035)

ВВЕДЕННЯ

Пристрої Інтернету Речей (далі ИР) можуть бути використані для того, щоб здійснювати дистанційний контроль здоров'я та надзвичайної системи оповіщення. Ці пристрої моніторингу здоров'я можуть варіюватися від артеріального тиску (пульсометри), до передових пристроїв, здатних моніторити імплантати, такі як кардіостимулятори або передові слухові апарати.

Телепатологія практика патології на відстані. Вона використовує телекомунікаційні технологіі для полегшення передачі даних зображення патології між віддаленими центрами для виконання цілей діагностики, освіти і наукових досліджень.

Телепатологія мається на увазі, що лікар вибирає зображення для аналізу та надання діагнозу.

Використання телевізійних мікроскопів є предтечею телепатології, вони дозволяють бути лікарю як фізичним так і віртуальним, для аналізу та діагностики.

Темою даного дипломного проекту є розробка програми для діагностики мазка крові, а виділення в ньому клітин лимфоцитів. Для реалізації даного завдання було вибрано середовище розробки MS Visual Studio 2015 мова програмування C++, та бібліотека комп'ютерного зору OpenCV.

Він дозволяє автоматизувати і прискорити пошук клітин, незалежно від якості мазка крові.

1 АНАЛІТИЧНИЙ ОГЛЯД

1.1 Бібліотека комп’ютерного зору OpenCV

OpenCV (англ. Open Source Computer Vision Library, бібліотека комп'ютерного зору з відкритим кодом) — бібліотека функцій та алгоритмів комп'ютерного зору, обробки зображень і чисельних алгоритмів загального призначення з відкритим кодом. Бібліотека надає засоби для обробки і аналізу вмісту зображень, у тому числі розпізнавання об'єктів на фотографіях (наприклад, осіб і фігур людей, тексту тощо), відстежування руху об'єктів, перетворення зображень, застосування методів машинного навчання і виявлення загальних елементів на різних зображеннях.

Бібліотека містить понад 2500 оптимізованих алгоритмів, серед яких повний набір як класичних так і практичних алгоритмів машинного навчання і комп'ютерного зору. Алгоритми OpenCV застосовують у таких сферах:

* аналіз та обробка зображень;
* системи з розпізнавання обличчя;
* ідентифікації об'єктів;
* розпізнавання жестів на відео;
* відстежування переміщення камери;
* побудова 3D моделей об'єктів;
* створення 3D хмар точок зі стерео камер;
* склеювання зображень між собою, для створення зображень всієї сцени з високою роздільною здатністю;
* система взаємодії людини з комп'ютером;
* пошуку схожих зображень із бази даних;
* усування ефекту червоних очей при фотозйомці зі спалахом;
* стеження за рухом очей;
* аналіз руху;
* ідентифікація об'єктів;
* сегментація зображення;
* трекінґ відео;
* розпізнавання елементів сцени і додавання маркерів для створення доповненої реальності;

та інші.

OpenCV написана на C++ і її основний інтерфейс також реалізовано на C++, але бібліотека і досі представляє старіший C інтерфейс. На даний момент реалізовано інтерфейс на мовах Python, Java і MATLAB / OCTAVE (починаючи з версії 2.5) . API для цих інтерфейсів можна знайти в онлайн документації. Оболонки для інших мов, таких як C#, CH, Ruby були розроблені з метою охоплення ширшої аудиторії.

Всі нові розробки та алгоритми OpenCV у даний момент розробляються у C++ інтерфейсі. [1]

Структура OpenCV представлена на рисунку 1.1.

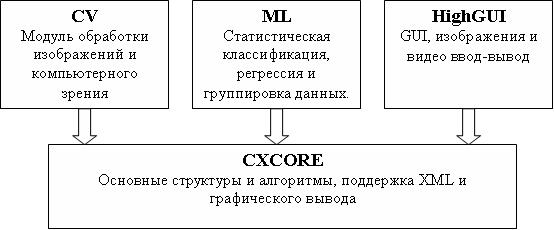


Рисунок 1.1 – Структура OpenCV

У ПП застосовуються функції для знаходження контурів.

void cv::findContours( InputOutputArray \_image, OutputArrayOfArrays \_contours,

OutputArray \_hierarchy, int mode, int method, Point offset )

{

Mat image = \_image.getMat();

MemStorage storage(cvCreateMemStorage());

CvMat \_cimage = image;

CvSeq\* \_ccontours = 0;

if( \_hierarchy.needed() )

\_hierarchy.clear();

cvFindContours(&\_cimage, storage, &\_ccontours, sizeof(CvContour), mode, method, offset);

if( !\_ccontours )

{

\_contours.clear();

return;

}

Порогові перетворення для елементів масиву зображення

double cv::threshold( InputArray \_src, OutputArray \_dst, double thresh, double maxval, int type )

{

CV\_OCL\_RUN\_(\_src.dims() <= 2 && \_dst.isUMat(),

ocl\_threshold(\_src, \_dst, thresh, maxval, type), thresh)

Mat src = \_src.getMat();

int automatic\_thresh = (type & ~CV\_THRESH\_MASK);

type &= THRESH\_MASK;

CV\_Assert( automatic\_thresh != (CV\_THRESH\_OTSU | CV\_THRESH\_TRIANGLE) );

if( automatic\_thresh == CV\_THRESH\_OTSU )

{

CV\_Assert( src.type() == CV\_8UC1 );

thresh = getThreshVal\_Otsu\_8u( src );

}

else if( automatic\_thresh == CV\_THRESH\_TRIANGLE )

{

CV\_Assert( src.type() == CV\_8UC1 );

thresh = getThreshVal\_Triangle\_8u( src );

}

\_dst.create( src.size(), src.type() );

Mat dst = \_dst.getMat();

if( src.depth() == CV\_8U )

{

int ithresh = cvFloor(thresh);

thresh = ithresh;

int imaxval = cvRound(maxval);

if( type == THRESH\_TRUNC )

imaxval = ithresh;

imaxval = saturate\_cast<uchar>(imaxval);

if( ithresh < 0 || ithresh >= 255 )

{

if( type == THRESH\_BINARY || type == THRESH\_BINARY\_INV ||

((type == THRESH\_TRUNC || type == THRESH\_TOZERO\_INV) && ithresh < 0) ||

(type == THRESH\_TOZERO && ithresh >= 255) )

{

int v = type == THRESH\_BINARY ? (ithresh >= 255 ? 0 : imaxval) :

type == THRESH\_BINARY\_INV ? (ithresh >= 255 ? imaxval : 0) :

/\*type == THRESH\_TRUNC ? imaxval :\*/ 0;

dst.setTo(v);

}

else

src.copyTo(dst);

return thresh;

}

thresh = ithresh;

maxval = imaxval;

}

else if( src.depth() == CV\_16S )

{

int ithresh = cvFloor(thresh);

thresh = ithresh;

int imaxval = cvRound(maxval);

if( type == THRESH\_TRUNC )

imaxval = ithresh;

imaxval = saturate\_cast<short>(imaxval);

if( ithresh < SHRT\_MIN || ithresh >= SHRT\_MAX )

{

if( type == THRESH\_BINARY || type == THRESH\_BINARY\_INV ||

((type == THRESH\_TRUNC || type == THRESH\_TOZERO\_INV) && ithresh < SHRT\_MIN) ||

(type == THRESH\_TOZERO && ithresh >= SHRT\_MAX) )

{

int v = type == THRESH\_BINARY ? (ithresh >= SHRT\_MAX ? 0 : imaxval) :

type == THRESH\_BINARY\_INV ? (ithresh >= SHRT\_MAX ? imaxval : 0) :

/\*type == THRESH\_TRUNC ? imaxval :\*/ 0;

dst.setTo(v);

}

else

src.copyTo(dst);

return thresh;

}

thresh = ithresh;

maxval = imaxval;

}

else if( src.depth() == CV\_32F )

;

else

CV\_Error( CV\_StsUnsupportedFormat, "" );

parallel\_for\_(Range(0, dst.rows),

ThresholdRunner(src, dst, thresh, maxval, type),

dst.total()/(double)(1<<16));

return thresh;

}

Перетворення зображення у HSV

void cv::cvtColor( InputArray \_src, OutputArray \_dst, int code, int dcn )

{

int stype = \_src.type();

int scn = CV\_MAT\_CN(stype), depth = CV\_MAT\_DEPTH(stype), bidx;

CV\_OCL\_RUN( \_src.dims() <= 2 && \_dst.isUMat() && !(depth == CV\_8U && (code == CV\_Luv2BGR || code == CV\_Luv2RGB)),

ocl\_cvtColor(\_src, \_dst, code, dcn) )

Mat src = \_src.getMat(), dst;

Size sz = src.size();

CV\_Assert( depth == CV\_8U || depth == CV\_16U || depth == CV\_32F );

CV\_IPP\_RUN(true, ipp\_cvtColor(src, \_dst, code, dcn));

switch( code )

{

case CV\_BGR2BGRA: case CV\_RGB2BGRA: case CV\_BGRA2BGR:

case CV\_RGBA2BGR: case CV\_RGB2BGR: case CV\_BGRA2RGBA:

CV\_Assert( scn == 3 || scn == 4 );

dcn = code == CV\_BGR2BGRA || code == CV\_RGB2BGRA || code == CV\_BGRA2RGBA ? 4 : 3;

bidx = code == CV\_BGR2BGRA || code == CV\_BGRA2BGR ? 0 : 2;

\_dst.create( sz, CV\_MAKETYPE(depth, dcn));

dst = \_dst.getMat();

if( depth == CV\_8U )

{

#ifdef HAVE\_TEGRA\_OPTIMIZATION

if(tegra::useTegra() && tegra::cvtBGR2RGB(src, dst, bidx))

break;

#endif

CvtColorLoop(src, dst, RGB2RGB<uchar>(scn, dcn, bidx));

}

1.2 Мова програмування С++ та середовище MS Visual Studio 2015

C++ (Сі-плюс-плюс) — мова програмування високого рівня з підтримкою декількох парадигм програмування: об'єктно-орієнтованої, узагальненої та процедурної. Розроблена Б'ярном Страуструпом (англ. Bjarne Stroustrup) в AT&T Bell Laboratories (Мюррей-Хілл, Нью-Джерсі) у 1979 році та початково отримала назву «Сі з класами». Згодом Страуструп перейменував мову у C++ у 1983 р. Базується на мові С.

У 1990-х роках С++ стала однією з найуживаніших мов програмування загального призначення. Мову використовують для системного програмування, розробки програмного забезпечення, написання драйверів, потужних серверних та клієнтських програм, а також для розробки розважальних програм таких як відеоігри. С++ суттєво вплинула на інші, популярні сьогодні, мови програмування: С# та Java.

При створенні С++ прагнули зберегти сумісність з мовою С. Більшість програм на С справно працюватимуть і з компілятором С++. С++ має синтаксис, заснований на синтаксисі С (див. список операторів мов С та С++).

Нововведеннями С++ порівняно з С є:

* підтримка об'єктно-орієнтованого програмування через класи;
* підтримка узагальненого програмування через шаблони;
* доповнення до стандартної бібліотеки;
* додаткові типи даних;
* обробка винятків;
* простори імен;
* вбудовані функції;
* перевантаження операторів;
* перевантаження імен функцій;
* посилання і оператори управління вільно розподіленою пам'яттю.

У 1998 році ратифіковано міжнародний стандарт мови С++: ISO/IEC 14882 «Standard for the C++ Programming Language». Поточна версія цього стандарту — ISO/IEC 14882:2011.

Microsoft Visual Studio — серія продуктів фірми Майкрософт, які включають інтегроване середовище розробки програмного забезпечення та ряд інших інструментальних засобів. Ці продукти дозволяють розробляти як консольні програми, так і програми з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms, а також веб-сайти, веб-застосунки, веб-служби як в рідному, так і в керованому кодах для всіх платформ, що підтримуються Microsoft Windows, Windows Mobile, Windows Phone, Windows CE, .NET Framework, .NET Compact Framework та Microsoft Silverlight.

Visual Studio включає в себе редактор вихідного коду з підтримкою технології IntelliSense і можливістю найпростішого рефакторінга коду. Вбудований відладчик може працювати як відладчик рівня вихідного коду, так і як відладчик машинного рівня. Решта вбудовуються інструменти включають в себе редактор форм для спрощення створення графічного інтерфейсу додатку, веб-редактор, дизайнер класів і дизайнер схеми бази даних. Visual Studio дозволяє створювати і підключати сторонні додатки (плагіни) для розширення функціональності практично на кожному рівні, включаючи додавання підтримки систем контролю версій вихідного коду (як, наприклад, Subversion і Visual SourceSafe), додавання нових наборів інструментів (наприклад, для редагування і візуального проектування коду на предметно-орієнтованих мовах програмування) або інструментів для інших аспектів процесу розробки програмного забезпечення [2].

1.2.1 Стандартна бібліотека шаблонів STL

Стандартна бібліотека шаблонів (англ. Standard Template Library; STL) — бібліотека для C++, що містить набір узгоджених узагальнених алгоритмів, контейнерів, засобів доступу до їхнього вмісту і різних допоміжних функцій.

Стандартна бібліотека шаблонів до включення в стандарт C++ була сторонньою розробкою, на початку — фірми HP, а потім SGI. Стандарт мови не називає її «STL», оскільки ця бібліотека стала невід'ємною частиною мови, проте багато людей досі використовують цю назву, щоб відрізняти її від решти частини стандартної бібліотеки (потоки вводу/виводу (iostream), підрозділ Сі тощо).

Проект під назвою STLPort, заснований на SGI STL, здійснює постійне оновлення STL, IOstream і рядкових класів. Деякі інші проекти також займаються розробкою приватних застосувань стандартної бібліотеки для різних конструкторських завдань. Кожен виробник компіляторів C++ обов'язково поставляє яку-небудь реалізацію цієї бібліотеки, оскільки вона є дуже важливою частиною стандарту і широко використовується.

У бібліотеці виділяють чотири основних компонентів:

* контейнер (container) - зберігання набору об'єктів в пам'яті;
* ітератор (iterator) - забезпечення засобів послідовного доступу до вмісту контейнера;
* алгоритм (algorithm) - визначення обчислювальної процедури;
* функціональний об'єкт (functor) - заховання функції в об'єкті для використання іншими компонентами.

Розділення дозволяє зменшити кількість компонентів. Наприклад, замість написання окремої функції пошуку елементу для кожного типу контейнера забезпечується єдина версія, яка працює з кожним з них, поки дотримуються основні вимоги.

1.2.2 Об'єктно-орієнтоване програмування

Об'є́ктно – орієнтоване програмува́ння (ООП) — одна з парадигм програмування, яка розглядає програму як множину «об'єктів», що взаємодіють між собою. Основу ООП складають три основні концепції: інкапсуляція, успадкування та поліморфізм. Одною з переваг ООП є краща модульність програмного забезпечення (тисячу функцій процедурної мови, в ООП можна замінити кількома десятками класів із своїми методами). Попри те, що ця парадигма з'явилась в 1960-тих роках, вона не мала широкого застосування до 1990-тих, коли розвиток комп'ютерів та комп'ютерних мереж дозволив писати надзвичайно об'ємне і складне програмне забезпечення, що змусило переглянути підходи до написання програм. Сьогодні багато мов програмування або підтримують ООП (Lua) або ж є цілком об'єктно-орієнтованими (зокрема, Java, C#, C++, Python, PHP, Ruby та Objective-C, ActionScript 3, Swift, Vala).

Об'єктно-орієнтоване програмування сягає своїм корінням до створення мови програмування Симула в 1960-тих роках, одночасно з посиленням дискусій про кризу програмного забезпечення. Разом із тим, як ускладнювалось апаратне та програмне забезпечення, було дуже важко зберегти якість програм. Об'єктно-орієнтоване програмування частково розв'язує цю проблему шляхом наголошення на модульності програми.

На відміну від традиційних поглядів, коли програму розглядали як набір підпрограм, або як перелік інструкцій комп'ютеру, ООП програми можна вважати сукупністю об'єктів. Відповідно до парадигми об'єктно-орієнтованого програмування, кожний об'єкт здатний отримувати повідомлення, обробляти дані, та надсилати повідомлення іншим об'єктам. Кожен об'єкт — своєрідний незалежний автомат з окремим призначенням та відповідальністю.

Об'єктно-орієнтоване програмування - це метод програмування, заснований на поданні програми у вигляді сукупності взаємодіючих об'єктів, кожен з яких є екземпляром певного класу, а класи є членами певної ієрархії наслідування. Програмісти спочатку пишуть клас, а на його основі при виконанні програми створюються конкретні об'єкти (екземпляри класів). На основі класів можна створювати нові, які розширюють базовий клас і таким чином створюється ієрархія класів.

На думку Алана Кея, розробника мови Smalltalk, якого вважають одним з «батьків-засновників» ООП, об'єктно-орієнтований підхід полягає в наступному наборі основних принципів:

* все є об'єктами;
* всі дії та розрахунки виконуються шляхом взаємодії (обміну даними) між об'єктами, при якій один об'єкт потребує, щоб інший об'єкт виконав деяку дію. Об'єкти взаємодіють, надсилаючи і отримуючи повідомлення. Повідомлення — це запит на виконання дії, доповнений набором аргументів, які можуть знадобитися при виконанні дії;
* кожен об'єкт має незалежну пам'ять, яка складається з інших об'єктів;
* кожен об'єкт є представником (екземпляром, примірником) класу, який виражає загальні властивості об'єктів;
* у класі задається поведінка (функціональність) об'єкта. Таким чином усі об'єкти, які є екземплярами одного класу, можуть виконувати одні й ті ж самі дії;
* класи організовані у єдину деревоподібну структуру з загальним корінням, яка називається ієрархією успадкування. Пам'ять та поведінка, зв'язані з екземплярами деякого класу, автоматично доступні будь-якому класу, розташованому нижче в ієрархічному дереві.

Таким чином, програма являє собою набір об'єктів, що мають стан та поведінку. Об'єкти взаємодіють використовуючи повідомлення. Будується ієрархія об'єктів: програма в цілому — це об'єкт, для виконання своїх функцій вона звертається до об'єктів що містяться у ньому, які у свою чергу виконують запит шляхом звернення до інших об'єктів програми. Звісно, щоб уникнути безкінечної рекурсії у зверненнях, на якомусь етапі об'єкт трансформує запит у повідомлення до стандартних системних об'єктів, що даються мовою та середовищем програмування. Стійкість та керованість системи забезпечуються за рахунок чіткого розподілення відповідальності об'єктів (за кожну дію відповідає певний об'єкт), однозначного означення інтерфейсів міжоб'єктної взаємодії та повної ізольованості внутрішньої структури об'єкта від зовнішнього середовища (інкапсуляції)[3].

1.3 Кольорова модель HSV

HSB — колірна модель, що використовується тільки для оформлення векторних і текстових об'єктів документа. Описує колірний простір, заснований на трьох характеристиках кольору: колірному тоні (Hue), насиченості (Saturation) і яскравості (Brightness).

Шкала відтінків — Hue

Hue — колірний тон, (наприклад, червоний, зелений або синьо-блакитний). Варіюється в межах 0-360°, але іноді приводиться до діапазону 0-100 або 0-1. У Windows весь колірний спектр ділиться на 240 відтінків (що можна спостерігати в редакторі палітри MS Paint), тобто тут «Hue» зводиться до діапазону 0-240 (відтінок 240 відсутній, оскільки він дублював би 0).

Saturation — насиченість. Варіюється в межах 0-100 або 0-1. Чим більший цей параметр, тим «чистіший» колір, тому цей параметр іноді називають чистотою кольору. А чим ближчий цей параметр до нуля, тим ближчий колір до нейтрального сірого.

Value (значення кольору) або Brightness — яскравість. Також задається в межах 0-100 або 0-1.

Модель була створена Елві Реєм Смітом, одним із засновників Pixar, в 1978 році. Вона є нелінійним перетворенням моделі RGB.

Колір, представлений в HSV, залежить від пристрою, на який він буде виведений, так як HSV — перетворення моделі RGB, яка теж залежить від пристрою. Для отримання коду кольору, не залежного від пристрою, використовується модель Lab.

Слід зазначити, що HSV (HSB) і HSL — дві різні колірні моделі.

Часто художники вважають за краще використовувати HSV замість інших моделей, таких як RGB і CMYK, тому що вони вважають, що пристрій HSV ближче до людського сприйняття кольорів. RGB і CMYK визначають колір як комбінацію основних кольорів (червоного, зеленого і синього або жовтого, пурпурного, блакитного і чорного відповідно), в той час як компоненти кольору в HSV відображають інформацію про колір у більш звичній людині формі : Що це за колір? Наскільки він насичений ? Наскільки він світлий чи темний? Кольорова палітра HSL представляє колір схожим і навіть, можливо, більш інтуїтивно зрозумілим чином, ніж HSV[2,3].

1.4 Технологія Raspberry pi

Raspberry Pi — одноплатний комп'ютер, розроблений британським фондом Raspberry Pi Foundation. Його головне призначення — стимулювати навчання базовим комп'ютерним наукам у школах.

У жовтні 2013 розробники одноплатного комп'ютера Raspberry Pi оголосили про продаж більше двох мільйонів цих плат, що закріпило за Raspberry Pi звання найпопулярнішої платформи для ентузіастів. Примітно, що за прогнозами двохмільйонного рубежу планувалося досягти в кінці лютого 2014 року, але популярність платформи перевищила очікування. Причиною успіху проекту можна назвати низьку ціну, відкритий характер розробки та орієнтацію на навчання та експерименти.

Raspberry Pi побудований на системі-на-чипі (SoC) Broadcom BCM2835, яка включає в себе процесор ARM із тактовою частотою 700 МГц, графічний процесор VideoCore IV, і 512 чи 256 мегабайт оперативної пам'яті. Твердий диск відсутній, натомість використовується SD карта. Така апаратна начинка дозволяє відтворювати відео формату H.264 в роздільній здатності 1080p, і запускати комп'ютерні ігри на зразок Quake III Arena.

Ініціатором проекту Raspberry Pi є британський благодійний фонд Raspberry Pi Foundation. Комп'ютер замислювався як пристрій для навчання дітей програмуванню, однак здобув популярність і в інших сферах — зокрема, на його основі роблять домашні медіацентри. Наддешевий Raspberry Pi поставляється без корпусу і має вигляд плати розміром з кредитну карту. Плата важить 45 грамів. У комп'ютері задіяний 700-мегагерцевий процесор на архітектурі ARM; має роз'єм для навушників і слот для карт пам'яті. Молодша (A) і старша (B) моделі Raspberry Pi відрізняються об'ємом оперативної пам'яті (256 мегабайт проти 512 мегабайт) і кількістю USB-портів (один проти двох). Крім цього, у старшої моделі є роз'єм Ethernet 10/100, а молодша споживає на третину менше енергії.

Старша модель Raspberry Pi надійшла в продаж в кінці лютого 2012 року по ціні 35 доларів. У лютому 2013 продаж в Європі надійшла молодша модель комп'ютера Raspberry Pi ціною 25 доларів США. Комп'ютери Raspberry Pi можна придбати у дистриб'юторів RS Components та element14. З лютого 2012 року було реалізовано понад мільйон пристроїв.

Для Raspberry Pi випущені спеціалізований дистрибутив Linux, Raspbian OS (він заснований на дистрибутиві Debian) і магазин застосунків Pi Store, де є як платні, так і безплатні програми. Серед іншого Raspberry Pi використовує веб-браузер Iceweasel та KOffice, які поставляються в комплекті. Офіційна мова програмування для навчання — Python[4].

2 ПОСТАНОВКА ЗАВДАННЯ

Дипломна робота є частиною комплексного завдання на розробку програм автоматичного керування медичними мікроскопами для отримання корисної інформації про клітини із мазку крові.

Алгоритм роботи мікроскопу має скадатися з таких кроків:

* переміщення предметного столику в кут прямокутника досліджуваної області мазка крові;
* знаходження необхідної чіткості зображення;
* виявлення клітин лімфоцитів на зображенні;
* класіфікація типу лімфоцитів;
* додавання результатів до статистики та збереження некласифікованих зображень для подальшої класифікацією фахівцем;
* якщо не досягнуто протилежної границі в напрямі X, то перемістити предметний столик на 1 крок в напрямі X. Інакше - перемістити предметний столик в напрямі X до протилежної границі прямокутника досліджуваної області мазка крові та зробити 1 крок в нарямі Y;
* якщо досягнуто протилежну границю пряокутника досліджуваної області мазка крові – зберегти дані статистики та припинити роботу програми. Інакше перейти до кроку 2.

Програми, розробляємі в дипломних роботах мають реалізують наступні функції:

* керування переміщенням предметного столику мікроскопу за двома координатами за допомогою двох крокових двигунів, на необхідну кількість кроків. Це дозволить розпізнати та обчислити кількість різних видів лімфоцитів на послідовності зображень різних участків мазку крові, отриманих відео- камерою окуляра;
* знаходження найчіткішого серед послідовності зображень клітин крові при зміненні різкості кроковим двигуном об´єктиву;
* виявлення клітин лімфоцитів на зображенні фрагмента мазку крові, отриманого з відео- камери, що дозволяє автоматизувати та прискорити пошук клітин. Доатковою вимогою до алгоритму є незалежність результатів від кольорових відмінностей зображення мазка крові.

У данній дипломній роботі потрібно розробити програмний продукт призначений для виявлення та детектування клітии лімфоцитів у мазку крові, відповідно до вимог віртуального мікроскопу, незалежно від якості мазка (шуми, пил, погана якість зображення, та тому подібне сміття). Програмний продукт повинен обробляти стек зображень мазка, які були зроблені віртуальним мікроскопом, шукає клітини лімфоцитів та детектує їх, це дозволяє прискорити роботу обробки мазків, у патологічних центрах.

2.1 Об'єкт дослідження

Об'єктом дослідження є розробка програми для знаходження клітин лімфоцитів у мазку крові(рис.2.1). Для реалізації даного завдання було вибрано середовище розробки MS Visual Studio 2015 мова програмування С++ та бібліотека комп'ютерного зору OpenCV.

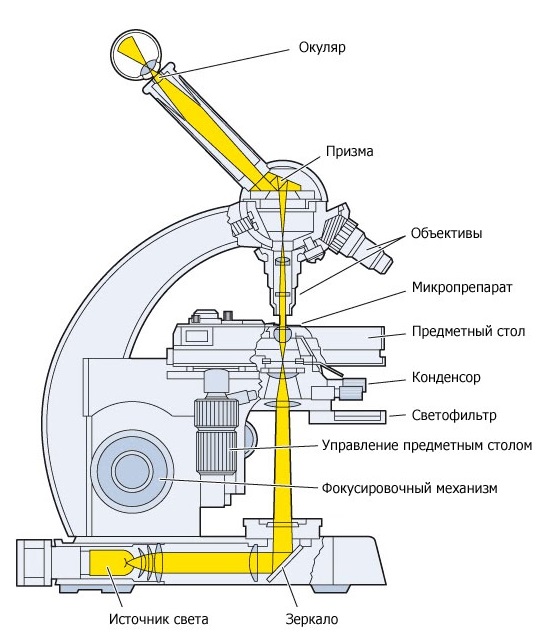


Рисунок 2.1 – Мікроскоп

ПП, реалізовано у концепції IOT(Internet Of Things).

2.2 Інтернет речей

Інтерне́т рече́й (англ. Internet of Things, IoT) — концепція комунікаційної мережі фізичних або віртуальних об'єктів («речей»), які мають технології для взаємодії між собою та з навколишнім середовищем, а також можуть виконувати певні дії без втручання людини.

Набуває поширення також термін англ. Internet of Everything, IoE — всеохопний, або всеосяжний інтернет.

Концепція полягає в тому, щоб всі предмети побуту, товари, вузли технологічних процесів тощо, були оснащені вбудованими комп'ютерами та сенсорами, мали змогу обробляти інформацію, що надходить із навколишнього середовища, обмінюватися нею та виконувати різні дії в залежності від отриманої інформації. Прикладом впровадження такої концепції може бути система «розумний будинок», яка вже застосовується у країнах Європи. Вона контролює параметри навколишнього середовища, в залежності від цього регулює температуру у приміщенні. У зимовий період нагріваючи прилади, в залежності від температури повітря ззовні, вітру, часу доби без втручання людини регулюють інтенсивність опалення, що дозволяє значно зменшити споживання енергоносіїв. На випадок спекотної погоди будинок має механізми відчинення та зачинення вікон, завдяки чому провітрюється будинок, та зменшується температура у ньому, а також систему керування кондиціонером[5].

Для об'єднання повсякденних речей у мережу потрібні декілька технологій.

Для ідентифікації кожного об'єкту потрібна проста, компактна технологія. Тільки при наявності системи унікальної ідентифікації можна збирати та накопичувати інформацію про певний предмет. Такий функціонал можна забезпечити за допомогою чіпів RFID (Radio-Frequency IDentification). Вони здатні без власного джерела струму передавати інформацію приладам зчитування. Кожен чіп має індивідуальний номер. Як альтернатива для даної технології для ідентифікації об'єктів можуть використовуватись QR-коди[6,7]. Для визначення точного місця знаходження речі підійде технологія GPS, яка ефективно використовується вже сьогодні у смартфонах та навігаторах:

* для відслідковування змін у стані елементу чи оточуючого середовища об'єкти повинні оснащуватися сенсорами;
* для обробки та накопичення даних з сенсорів повинен використовуватися вбудований комп'ютер (наприклад Raspberry Pi, Intel Edison);
* для обміну інформацією між пристроями можуть бути використані технології бездротових мереж (Wi-Fi, Bluetooth, ZigBee, 6LoWPAN).

Але вже сьогодні існують рішення, здатні об'єднати різні програмні комплекси і різні пристрої в єдину екосистему. Для цього потрібні API. А інструментом для збору всього воєдино є, наприклад, хмарна OC Corezoid.com. Ця система базується на теорії кінцевих автоматів і вміє не тільки описувати логіку процесу, але і описувати стани об'єктів цього процесу, що надзвичайно важливо для iнтернету речей.[5,8]

2.3 Вимоги до розробки

Розробити програмний продукт, який би виявляв клітини лімфоцитів у мазку крові. Це допоможе більш якісно виявляти та детектувати клітини лімфоцитів у мазку крові. Це потрібно для подальшої більш ефективної роботи мікроскопа.

Основні вимоги:

* розробити алгоритм для знаходженя клітин у системі HSV;
* розробити алгоритм для знаходження клітин лімфоцитів у мазку крові;
* детектувати лімфоцити на мазку;

3 РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

3.1 Призначення програмного продукту та мета разробки

Програмний продукт (далі ПП), призначений для виявлення та детектування клітин лімфоцитів у мазку крові, відповідно до вимог віртуального мікроскопу.

ПП, реалізовано у концепції IOT(Internet Of Things).

Метою розробки даного ПП, є підвищення ефективності виявлення та детектування клітин у віртуальних мікроскопах, алгоритм реалізований у даному ПП, дозволяє ефективно виявляти клітини, детектувати їх незалежно від якості мазка (шуми, пил, погана якість зображення, та тому подібне сміття).

ПП обробляє стек зображень мазка, які були зроблені віртуальним мікроскопом, шукає клітини лімфоцитів та детектує їх, це дозволяє прискорити роботу обробки мазків, у патологічних центрах.

3.2 Об'єкт реалізації

Електронна – медицина (далі Е – медицина) повинна забезпечувати взаємодію між пацієнтами, медичними працівниками та установами за допомогою інформаційно-комунікаційних технологій.

Е – медицина є відносно новим терміном охорони здоров'я за підтримки електронних процесів і комунікацій, що почав набувати розповсюдження у світі з 1999 року. З тих пір значення терміну постійно змінюється: деякі стверджують, що це термін, що заміняє термінкомп'ютерні технології в медицині, а інші використовують його в більш вузькому сенсі практик охорони здоров'я з використанням Інтернету.Він також може включати в себе програми для слідкуванням здоров'ям на мобільні телефони, виокремлюється поняття мобільної медицини. Приблизно з 2011 року, у світі визнається все більша необхідність посиленого захисту спеціалізованих ресурсів е-медицини від кібер-атак, що вимагає розробки більш надійної інфраструктури в області електронної медицини, яка зможе витримати зростаючі загрози.

Основними напрямами діяльності в галузі розвитку е – медицини є:

* впровадження автоматизованих інформаційних галузевих систем, які, зокрема, дають змогу перейти до ведення медичної документації в електронному вигляді;
* розвиток телемедицини;
* удосконалення розвитку системи моніторингу стану здоров’я населення;
* створення та впровадження нових комп’ютерних технологій профілактики захворювань, діагностики, забезпечення лікувальних процесів;
* створення загальнодоступних електронних медичних ресурсів.

Е — медицина дозволяє, клієнту зберігати на своїх пристроях зображення, відео, аудіо, їх обстежень, потім у незалежний час вони можуть безпечно передати їх у клініки для обстеження різними лікарями. Потім цій спеціаліст мусить передати ці данні назад. Ця операція може займати від 1 хвилини до 48 годин[9].

3.3 Цифрова патологія.

Цифрова патологія – інформаційна середа, заснована на цифрових зображеннях препаратів. Цифрова патологія є частиною віртуальної мікроскопії та за фактом являє собою перетворення препаратів, що знаходяться на предметних стеклах, в їх високоякісні цифрові копії. Цифрові копії препаратів згодом можуть бути використані для перегляду, аналізу, архівування, проведення телеконференції, зовнішнього консультування, навчання і т. д. Цифрова патологія дозволяє знизити вартість, одночасно збільшивши швидкість і якість діагностики та прогнозування перебігу патології. Цифрова патологія є затребуваним і ефективним інструментом при постановці діагнозу, дозволяє перейти від перегляду необхідної кількості полів зору в мікроскоп до візуалізації всього препарату цілком в надвисокому дозволі.

Можливості цифрової патології:

* перегляд цифрової копії препарату в надвисокому дозволі;
* застосування алгоритмів морфометричного аналізу;
* створення архіву оцифрованих препаратів з первозданним забарвленням і можливістю додавання інформації про пацієнта, лікарі, діагнозі і т. д .;
* отримання другого думки;
* проведення зовнішнього консультування в онлайн режимі;
* навчання на віртуальних препаратах.

Приклад цифрової патології на рис. 3.1



Рисунок 3.1 – Приклад цифрової патології

Для створення цифрових копій препаратів, що знаходяться на предметному склі, використовуються спеціальні прилади – сканери препаратів.

Всі сучасні сканери препаратів глобально можна розділити за принципом сканування на дві великі групи:

* панорамування;
* лінійне сканування.

Панорамування є традиційним способом отримання зображень, що складаються більше, ніж з одного поля зору. Результуюче зображення утворюється шляхом програмної зшивання різних полів зору. Хоча сучасне програмне забезпечення дозволяє проводити дуже якісну обробку зображень, проте на результуючих зображеннях відсканованих препаратів нерідко можна зустріти помилки зшивання у вигляді зрушень фрагментів.

Лінійне сканування – більш прогресивний і високоточний метод. Сканування здійснюється з використанням лінійної TDI матриці. Подібні матриці використовуються при картографуванні та аерофотозйомки. Принцип дії сканера, робота якого заснована на лінійному методу сканування, можна зрозуміти, уявивши собі звичайний сканер паперу: здійснюється порядкове сканування об'єкту. Результуюче зображення формується відразу попиксельно. Імовірність помилок в обробці підсумкового зображення і некоректності зшивок значно знижується.

Цифрова патологія — це оточення засноване на зображенні зробленного комп'ютером. Ця технологія дозволяє управляти інформацією на зображенні. Цифрова патологія в даний час вважається одним з найбільш перспективних напрямків в діагностичній медицині. Цифрова патологія вбудована у віртуальний мікроскоп, який конвертує зображення з лінз до цифрового зображення, які можуть бути проаналізовані[9,10].

3.4 Кроковий двигун 28BYJ-48

Об’єктом реалізації є мікроскоп, для керування різкістю були застосовані крокові двигуни. Керування кроковим двигуном дозволяє здійснити програмне фокусування мікроскопу.

28BYJ-48 — пятівольтовий кроковий двигун, використовується в робототехніці. Кроковий двигун може точно переміщатися на мінімально можливий кут, званий кроком. Для практичних завдань можна вважати, що кроковий мотор трохи схожий на сервопривід. Можна задати йому повернутися в деякий стан і можна розраховувати отримати досить стабільні результати в декількох повторних експериментах. Зазвичай, сервоприводи обмежені кутом повороту в діапазоні від 0 до 180 °, кроковий же двигун може обертатися безперервно, подібно двигуну постійного струму. Перевагою крокових двигунів є те, що можна досягти набагато більшому ступені контролю над рухом. До недоліком крокових двигунів можна віднести кілька більш складне управління, ніж у випадках з сервами або моторами постійного струму.

Кроковий двигун зображен на наступному рисунку(рис 3.2):



Рисунок 3.2 – Кроковий двигун 28BYJ-48

Для роботи мікроскопа потрібно розробити модуль керування кроковими двигунами на базі Arduino, який буде орієнтован на керування предметним стільцем мікроскопу.

Модуль керування представляє собою програму яка виконується на платформі Arduino Nano, котрий відправляє дані на крокові двигуни (індекс та кількість обертань відповідного двигуна). Потім, двигуни обертаються необхідну кількість раз. Після обертання необхідно програмно повернути крокові двигуни у стартову позицію.

Модуль керування складається з декількох взаємодіючих частин: крокових двигунів, драйверів та платформи Arduino Nano як показано на рисунку 3.3.

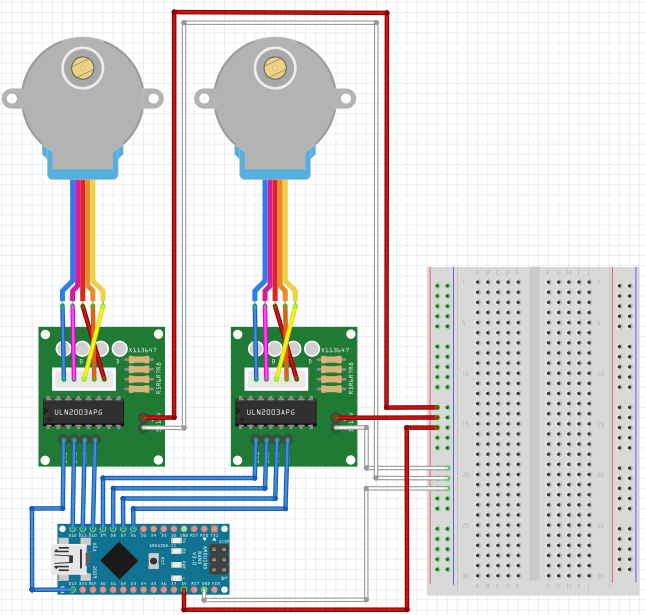


Рисунок 3.3 – Схема модуля керування кроковими двигунами

3.5 Алгоритми роботи програми

На початку роботи ПП, він сканує шлях до зображень, якщо він знаходить зображення формату .jpg/.png, починає їх обробляти.

void loadFromDir(char\* argv[]) {

string fullPath = "";

string dirPath = argv[1];

struct dirent \*dentry = NULL;

char \*ext = NULL;

DIR \*d = opendir(argv[1]);

while ((dentry = readdir(d)) != NULL)

{

ext = dentry->d\_name + (strlen(dentry->d\_name) - 4);

if (ext > dentry->d\_name)

if ((strncmp(ext, ".png", 4) == 0) ||

(strncmp(ext, ".jpg", 4) == 0)) {

fullPath = dirPath + dentry->d\_name;

srcImage = imread(fullPath);

origImage = srcImage;

namedWindow("src", cv::WINDOW\_AUTOSIZE);

imshow("src", srcImage);

myHSV(srcImage);

findingContours(hsvImage);

cout << endl << dentry->d\_name << endl;

char c = waitKey(0);

if (c == 27) { // если нажата ESC - выходим

break;

}

}

}

}

Далі зображення переводяться до формату HSV, та знаходить об’єкти характеризовані для клітин лімфоцитів.

Приклад як зображення перетворюється у формат HSV представлен на рисунку 3.4

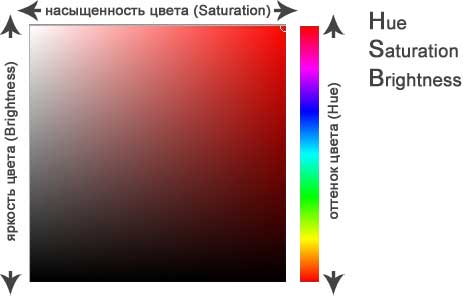


Рисунок 3.4 – Приклад як зображення перетворюється у формат HSV

Код перетворення наведен нижче

void myHSV(const Mat& src) {

Mat tempHsvImage;

cvtColor(src, hsvImage, COLOR\_BGR2HSV);

Mat upperRange;

Mat lowerRange;

inRange(hsvImage, cv::Scalar(122, 140, 100), cv::Scalar(145, 255, 255), upperRange);

Mat lympho;

namedWindow("hsv image", cv::WINDOW\_AUTOSIZE);

imshow("hsv image", upperRange);

hsvImage = upperRange;

}

Якщо він їх знаходить він вичислює, їх контур та площю, завдяки контуру детектується клітина, а площа допомагає відкинути усі «сміттєві» детектування.

Блок схема головного алгоритму представлена на рис. 3.2.

C:\Users\Кирилл\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Untitled Diagram.png

Рисунок 3.2 – Блок-схема алгоритму

3.5.1 Алгоритм RGB – HSV зображення

При переводі зображень з RGB – HSV, ПП використовує скалярні значення для знаходження кольору клітини[12].

Рахуємо, що:


\begin{align}
  H & \in \left[ 0, 360 \right) \\
  S,V,R,G,B & \in \left[ 0, 1 \right]
\end{align}
 (3.1)

Нехай  MAX — максимальное значение из R, G и B, а MIN — минимальное из них.

|  |  |
| --- | --- |
| H =  \begin{cases} \\ \\ \\ \\ \\ \\ \\ \\ \end{cases} | 0, если MAX = MIN |
| 60 \times \frac{G - B}{MAX - MIN} + 0, если MAX = R~ и G \ge B |
| 60 \times \frac{G - B}{MAX - MIN} + 360, если MAX = R~ и G < B~ |
| 60 \times \frac{B - R}{MAX - MIN} + 120, если MAX = G~ |
| 60 \times \frac{R - G}{MAX - MIN} + 240, если MAX = B~ |

(3.2)

|  |  |
| --- | --- |
| S =  \begin{cases} \\ \\ \end{cases} | 0,если  MAX = 0; |
| 1 - \dfrac{MIN}{MAX},иначе |

V = {MAX} (3.3)

Код роботи алгоритма RGB – HSV зображення представлен нижче

cvtColor(src, hsvImage, COLOR\_BGR2HSV);

Mat upperRange;

Mat lowerRange;

inRange(hsvImage, cv::Scalar(122, 140, 100), cv::Scalar(145, 255, 255), upperRange);

3.5.2 Алгоритм знаходження контура клітин та площі

Після знаходження контурів плям клітин, ПП знаходить центр цього контура, и площу.

Алгоритм знаходження контура та площі представлені на рис 3.3



Рисунок 3.3 – Блок схема алгоритму

Код роботи алгоритм знаходження контура клітин та площі представлен нижче.

findContours(src, contours, hierarchy, CV\_RETR\_EXTERNAL, CV\_CHAIN\_APPROX\_SIMPLE, Point(0, 0));

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

{

double area = fabs(contourArea(contours[i]));

double perim = arcLength(contours[i], true);

if (area > threshArea) {

Point coordinateOfContour = center[i];

cell = new Lymphocytes(coordinateOfContour.x, coordinateOfContour.y);

lymphocyts.push\_back(cell);

cout << endl << "area " << area;

cout << endl << "count " << lymphocyts.size();

}

}

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

circle(origImage, Point(lymphocyts[i]->getX(), lymphocyts[i]->getY()), circleRadius, Scalar(0, 0, 0), 2, 8, 0);

cout << endl << "contour with coordinates: x = " << lymphocyts[i]->getX() << " y = " << lymphocyts[i]->getY();

}

4 ОПИС ГОТОВОГО ПРОГРАМНОГО ПРОДУКТА

4.1 Головний інтерфейс програми

Комп'ю́терний зір або Комп'ютерне бачення — теорія та технологія створення машин, які можуть проводити виявлення, стеження та класифікацію об'єктів[13].

Алгоритм сегментації зображень представлен на рис 4.1

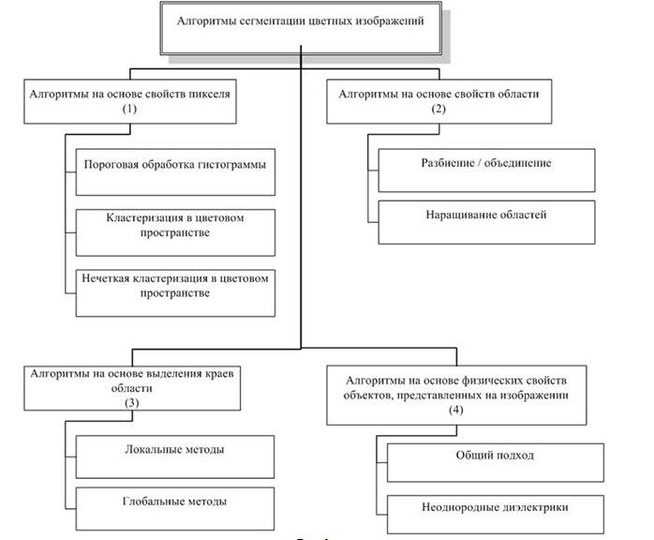


Рисунок 4.1 – Алгоритм сегментації зображень

Як наукова дисципліна, комп'ютерний зір належить до теорії та технології створення штучних систем, які отримують інформацію у вигляді зображень. Відеодані можуть бути представлені у вигляді багатьох форм, таких як відеопослідовність, зображення з різних камер або тривимірними даними з медичного сканера.

Як технологічна дисципліна, комп'ютерний зір прагне застосувати теорії та моделі комп'ютерного зору до створення систем комп'ютерного зору. Прикладами таких систем можуть бути:

* системи керування процесами (промислові роботи, автономні транспортні засоби);
* системи відео спостереження;
* системи організації інформації (наприклад, для індексації баз даних зображень);
* системи моделювання об'єктів або оточуючого середовища (аналіз медичних зображень, топографічне моделювання);
* системи взаємодії (наприклад, пристрої введення для систем людино-машинної взаємодії).

Комп'ютерний зір також може бути описаний як доповнення (але не обов'язково протилежність) біологічному зору. В біології вивчається зорове сприйняття людини і різноманітних тварин, в результаті створюються моделі роботи таких систем в термінах фізіологічних процесів. Комп'ютерний зір, з іншої сторони, вивчає і описує системи комп'ютерного зору, які виконані апаратно або програмно. Міждисциплінарний обмін між біологічним та комп'ютерним зором виявився досить продуктивним для обох наукових галузей.

Підрозділи комп'ютерного зору включають відтворення дій, виявлення подій, стеження, розпізнавання образів, відновлення зображень.

Комп'ютерний зір, Обробка зображень, Аналіз зображень, Зір робота і Машинний зір — тісно пов'язані області. Але досі точно не визначено, чи є вони розділами однієї, ширшої галузі. При детальному аналізі може здатись, що це лише різні назви однієї і тієї ж області. Щоб не виникло плутанини, прийнято розрізняти їх як напрямки, сфокусовані на певному предметі вивчення. Нижче наведено опис деяких з них, найбільш важливих[13,14].

Обробка зображень і Аналіз зображень в основному зосереджені на роботі з двовимірними зображеннями, тобто як перетворити одне зображення в інше. Наприклад, попіксельні операції збільшення контрастності, операції з виділення країв, усунення шумів чи геометричні перетворення, такі як обертання зображення. Дані операції припускають, що обробка/аналіз зображення діють незалежно від вмісту самих зображень.

Одним з найбільш важливих застосувань є обробка зображень в медицині. Ця область характеризується отриманням інформації з відеоданих для визначення медичного діагнозу пацієнту. В більшості випадків, відеодані отримують за допомогою мікроскопії, рентгенографії, ангіографії, ультразвукових досліджень та томографії. Прикладом інформації, яка може бути отримана з таких відеоданих є виявлення пухлин, атеросклерозу чи інших злоякісних змін. Також прикладом може слугувати вимірювання розмірів органів, кровообігу тощо. Ця прикладна область також сприяє медичним дослідженням, надаючи нової інформації, наприклад, про будову мозку чи якості медичного лікування.

Класична задача в комп'ютерному зорі, обробці зображень і машинному зорі — це визначення того, чи містять відеодані деякий характерний об'єкт, особливість чи активність. Ця задача може бути достовірно і легко вирішена людиною, але досі не вирішена задовільно в комп'ютерному зорі в загальному випадку: випадкові об'єкти у випадкових ситуаціях.

Існуючі методи вирішення цієї задачі ефективні тільки для окремих об'єктів, таких як прості геометричні об'єкти (наприклад, багатокутники), людські обличчя, друковані чи рукописні символи, автомобілі і лише у визначених умовах, зазвичай це певне освітлення, фон і положення об'єкта відносно камери[15].

В літературі описане різноманіття проблем розпізнавання:

* розпізнавання: один чи декілька попередньо заданих чи вивчених об'єктів або класів об'єктів можуть бути розпізнані, зазвичай разом з їх двовимірним положенням на зображенні чи тривимірним положенням в сцені;
* ідентифікація: розпізнається індивідуальний екземпляр об'єкта. Приклади: ідентифікація визначеного людського обличчя чи відбитка пальців чи автомобіля;
* виявлення: відеодані перевіряються на наявність визначеної умови. Наприклад, виявлення можливих неправильних клітин чи тканин в медичних зображеннях. Виявлення, що базується на відносно простих і швидких обчисленнях, іноді використовується для знаходження невеликих ділянок в зображенні, що аналізується, які потім аналізуються за допомогою заходів, що потребують більше ресурсів, для отримання правильної інтерпретації.

Існує кілька спеціалізованих задач, що базуються на розпізнаванні, наприклад:

Пошук зображень за вмістом: знаходження всіх зображень серед великого набору зображень, які мають певний вміст. Вміст може бути визначено різними шляхами, наприклад в термінах схожості з конкретним зображенням (знайти всі зображення, що схожі на дане зображення), чи в термінах високорівневих критеріїв пошуку, що вводяться як текстові дані (знайти всі зображення на яких зображено багато будинків, які зроблені взимку і на яких нема машин).

Коли ПП закінчує аналіз зображень він відкриває вікно з готовим зображенням та вже знайденими клітинами а також зображення в градаціях сірого і плямою клітини на зображенні, у консолі ПП говорить скільки він знайшов клітин, яка в них площа, та її координати[14,15].

Консоль після знайдених клітин представленні на рис. 4.2

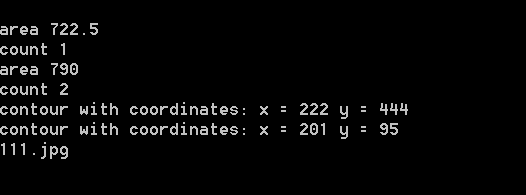


Рисунок 4.2 – Консоль після знайдених клітин

4.2 Сегментація зображення

Однією з основних завдань обробки та аналізу зображень є сегментація, тобто поділ зображення на області, для яких виконується певний критерій однорідності, наприклад, виділення на зображенні областей приблизно однакової яскравості. Поняття області зображення використовується для визначення зв'язковий групи елементів зображення, що мають певний загальний ознак (властивість).

Один з основних і простих способів - це побудова сегментації за допомогою порога. Поріг - це ознака (властивість), яке допомагає розділити шуканий сигнал на класи. Операція порогового поділу полягає в зіставленні значення яскравості кожного пікселя зображення з заданим значенням порога.

Операція порогового поділу, яка в результаті дає бінарне зображення, називається бінаризація. Метою операції бінаризації є радикальне зменшення кількості інформації, що міститься на зображенні. В процесі бінаризації вихідне півтонування, що має певну кількість рівнів яскравості, перетворюється в чорно-біле зображення, пікселі якого мають тільки два значення - 0 і 1

Порогова обробка зображення може проводитися різними способами.

Бінаризація з нижнім порогом

Бінаризація з нижнім порогом є найбільш простою операцією, в якій використовується тільки одне значення порога:

(4.1)

Всі значення замість критерію стають 1, в даному випадку 255 (білий) і все значення (амплітуди) пікселів, які більше порога t - 0 (чорний).

Визначення порога на основі градієнта яскравості зображення

Алгоритм обчислення граничного значення складається з наступних 2 кроків:

1. Визначається модуль градієнта яскравості для кожного пікселя зображення:

(4.2)

2. Обчислення порога:

(4.3)

В даний час існує група алгоритмів обробки зображень: поелементні перетворення, виділення контурів, обробка ковзної вікном (рисунок 4.3). Названі алгоритми приводять до поліпшення якості при сприйнятті зображення людиною спостерігачем[16].

В процесі розвитку теорії та практики роботи з зображеннями під обробкою зображень приймаються не тільки перетворення, що призводять до поліпшення якості бачення зображення, а й ті, які призводять вихідне зображення до виду, зручного для подальшої їх обробки за допомогою машин (забезпечення машинного (технічного) зору ). У цих випадках інтерес викликають зображення не самі по собі, а та інформація, яка в них міститься.

Витяг цієї інформації і становить мета обробки відеоінформації (зображень), яке проводиться автоматично, т. б. без участі оператора. Така обробка зображень містить сегментацію зображень, порогову обробку, проведену в інтересах виявлення об'єктів[15,17].

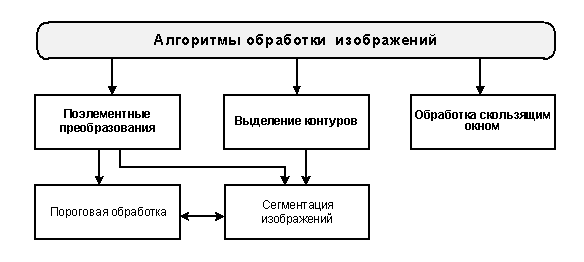


Рисунок 4.3 – Алгоритм обробки зображення

Зображення у градаціях сірого представлене на рис. 4.4

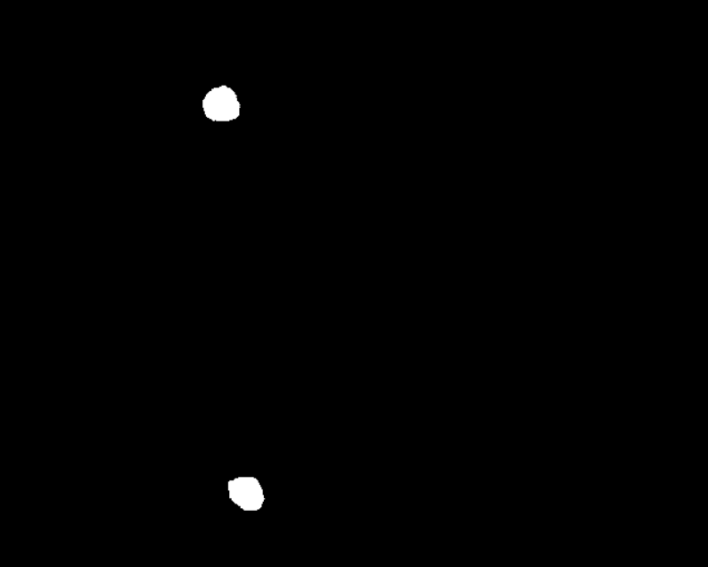


Рисунок 4.4 – Зображення у градаціях сірого

4.3 Виявлення контурів

Ви́явлення ко́нтурів (англ. edge detection) — це назва набору математичних методів, спрямованих на виявлення точок цифрового зображення, в яких яскравість зображення змінюється різко, або, формальніше, має розриви. Точки, в яких яскравість зображення змінюється різко, зазвичай утворюють набір відрізків кривих ліній, що називаються контурами (англ. edge). Аналогічна задача знаходження розривів в одномірних сигналах відома як виявлення сходинок, а задача знаходження розривів сигналу в часі відома як виявлення змін. Виявлення контурів є основним інструментом в обробці зображень, машинному зорі та комп'ютерному зорі, особливо в областях виявлення ознак та виділяння ознак.

Метою виявлення різких змін яскравості зображення є фіксування важливих подій та змін у властивостях світу. Може бути показано, що за досить загальних припущень стосовно моделі формування зображення, розриви яскравості зображення, ймовірно, відповідають:

* + розривам в глибині;
  + розривам в орієнтації поверхні;
  + змінам у властивостях матеріалу;
  + відхиленням в освітленні сцени.

В ідеальному випадку, результатом застосування до зображення детектора контурів може бути набір з'єднаних кривих, що позначають межі об'єктів, межі забарвлення поверхонь, а також усі криві, що відповідають розривам в орієнтації поверхонь. Таким чином, застосування алгоритму виявлення контурів до зображення може значно зменшувати кількість даних, що підлягають обробці, відфільтровуючи інформацію, яка може розглядатися як менш значуща, але зберігаючи важливі структурні властивості зображення. Якщо крок виявлення контурів є успішним, то подальшу задачу інтерпретування інформаційного вмісту первісного зображення може бути істотно спрощено. Проте не завжди можливо отримувати такі ідеальні контури в картинах реального світу середньої складності[17].

Контури, виділені з нетривіальних зображень, часто пошкоджено фрагментацією, що означає, що криві контурів не є з'єднаними, відсутні відрізки контурів, а також є хибні контури, що не відповідають досліджуваному явищу в зображенні — ускладнюючи таким чином подальшу задачу інтерпретування даних зображення.

Приклад виявдення контурів представлен на рис 4.5

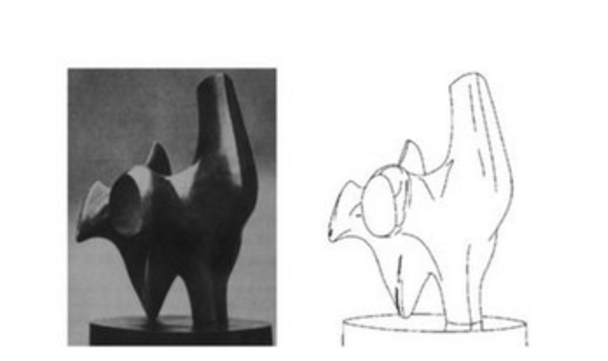


Рисунок 4.5 – Приклад виявдення контурів

Виявлення контурів є одним з основних кроків в обробці зображень, аналізі зображень, розпізнаванню образів у зображеннях, та в методиках комп'ютерного бачення.

Контури, виділені з двовимірного зображення тривимірної сцени, може бути класифіковано або як залежні від точки огляду (англ. viewpoint), або як незалежні від неї. Незалежний від точки огляду контур зазвичай відображає властивості, притаманні тривимірним об'єктам, такі як забарвлення поверхні та її форма. Залежний від точки огляду контур може змінюватися при зміні точки огляду, і зазвичай відображає геометрію сцени, таку як затуляння одного об'єкту іншим.

Типовий контур може бути, наприклад, межею між блоками червоного та жовтого кольорів. З іншого боку, лінія (яку може бути виділено детектором хребтів) може бути невеликою кількістю пікселів іншого кольору на в іншому незмінному тлі. Тому зазвичай для лінії може бути по одному контуру з кожного з її боків.

Контури мають досить важливе значення в багатьох застосуваннях обробки зображень, особливо в системах машинного зору, які аналізують сцени штучних об'єктів за фіксованого освітлення. В останні роки, однак, було послідовно (й успішно) проведено дослідження методів комп'ютерного зору, які не покладаються на виявлення контурів як на крок перед обробки[18].

Зображення зі знайденими клітинами представленні на рис. 4.6

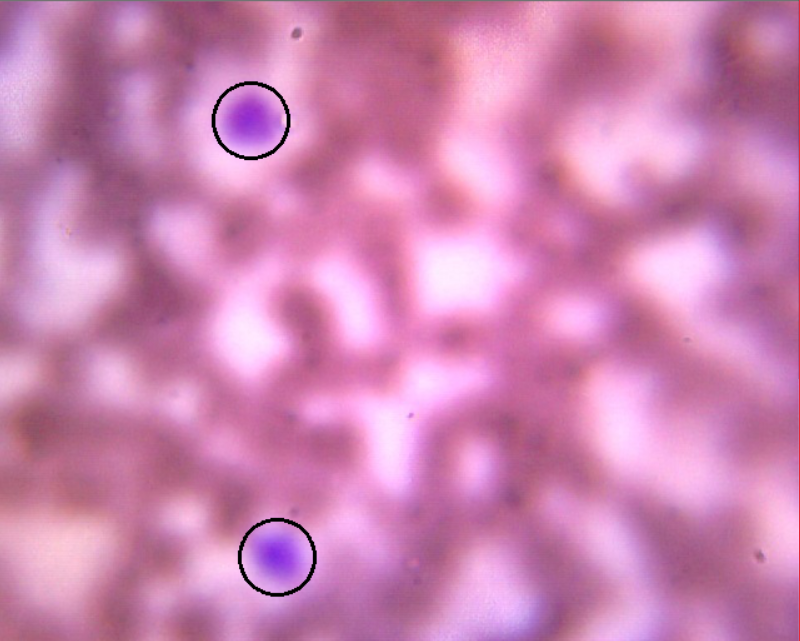


Рисунок 4.6 – Зображення зі знайденими клітинами

4.3.1 Інтерфейс програми для визначення HSV параметрів

Для визначення значень HSV використовувалась програма для знаходження параметрів HSV, шляхом зміни повзунків, знаходяться значення параметрів, це скалярне значення (H,S,V).

Для приклада візьмемо тіж самі значення якими оперує програма для знаходження клітин.

Програма відкриває вікна з зображеннями переводить їх у формат HSV, та розбиває на різні канали.

hsv = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 3);

h\_plane = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 1);

s\_plane = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 1);

v\_plane = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 1);

h\_range = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 1);

s\_range = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 1);

v\_range = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 1);

hsv\_and = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 1);

cvCvtColor(image, hsv, CV\_BGR2HSV);

cvSplit(hsv, h\_plane, s\_plane, v\_plane, 0);

В комп'ютерному зорі та обробці зображень поняття ви́явлення озна́к (англ. feature detection) стосується методів, спрямованих на обчислення абстракцій інформації зображення, та ухвалення локальних рішень в кожній точці зображення, чи є ознака зображення заданого типу в цій точці, чи ні. Отримані ознаки будуть підмножинами області визначення зображення, часто в вигляді ізольованих точок, неперервних кривих або зв'язних областей.

Не існує універсального чи точного визначення, що собою являє ознака, і точне визначення часто залежить від задачі або типу застосування. Враховуючи це, ознака визначається як «цікава» частина зображення, і ознаки використовуються як відправні точки для багатьох алгоритмів комп'ютерного зору. Оскільки ознаки використовуються як відправні точки та основні примітиви для наступних алгоритмів, загальний алгоритм часто буде лише настільки добрим, наскільки добрим є його детектор ознак. Отже, бажаною властивістю детектора ознак є повторюваність: чи буде одну й ту ж ознаку виявлено на двох або більше різних зображеннях однієї й тієї ж сцени, чи ні.

Виявлення ознак є низькорівневою операцією обробки зображень. А саме, вона зазвичай виконується як перша операція на зображенні, і перевіряє кожен піксель, щоби побачити, чи присутня ознака в цьому пікселі. Якщо це є частиною більшого алгоритму, то цей алгоритм зазвичай перевірятиме зображення лише в областях ознак. Як вбудована передумова для виявлення ознак, вхідне зображення зазвичай згладжується ґаусовим ядром у масштабно-просторовому представленні, й обчислюється одне або декілька зображень ознак, часто виражених в термінах операцій локальних похідних зображень.

Іноді, коли виявлення ознак є обчислювально витратним, і присутні часові обмеження, може застосовуватися алгоритм вищого рівня для скеровування етапу виявлення ознак, так що пошук ознак здійснюватиметься лише деякими частинами зображення.

Багато алгоритмів комп'ютерного зору використовують виявлення ознак в якості першого кроку, так що в результаті було розроблено дуже велику кількість детекторів ознак. Вони сильно різняться за типами ознак, що виявляють, за обчислювальною складністю та повторюваністю.

Плями (англ. blob) забезпечують додатковий опис структур зображення в термінах областей (англ. region), на противагу до кутів, що є більш точкоподібними. Тим не менше, плямови описувачі можуть часто містити особливу точку (локальний максимум відповіді оператора, або центр мас), що означає, що багато детекторів плям можуть також розглядатися як оператори особливих точок. Детектори плям можуть виявляти такі області в зображенні, які є занадто згладженими для виявлення детектором кутів.

Розгляньмо зменшення зображення із наступним виконанням виявлення кутів. Детектор зреагує на точки, що є контрастними у зменшеному зображенні, але можуть бути згладженими в первісному зображенні. Це саме той момент, коли різниця між детекторами кутів та плям стає дещо невиразною. В значній мірі це розрізнення може бути виправлено включенням відповідного поняття масштабу. Тим не менше, завдяки своїм властивостям реагування на різні типи структур зображення на різних масштабах, детектори плям ЛҐ та РҐ згадуються також і в статті про виявлення кутів[17,18,19].

Поки працює програма вона показує ресурси H,S,V після того як ви вибрали цікаві для вас ділянки вона їх зводить в одну «Карту пошуку клітини лімфоцитів»

while (true) {

cvShowImage("original", image);

cvShowImage("H", h\_plane);

cvShowImage("S", s\_plane);

cvShowImage("V", v\_plane);

cvShowImage("H range", h\_range);

cvShowImage("S range", s\_range);

cvShowImage("V range", v\_range);

cvAnd(h\_range, s\_range, hsv\_and);

cvAnd(hsv\_and, v\_range, hsv\_and);

cvShowImage("hsv and", hsv\_and);

char c = cvWaitKey(33);

if (c == 27) {

break;

}

}

Приклад знаходження H на рис.4.7

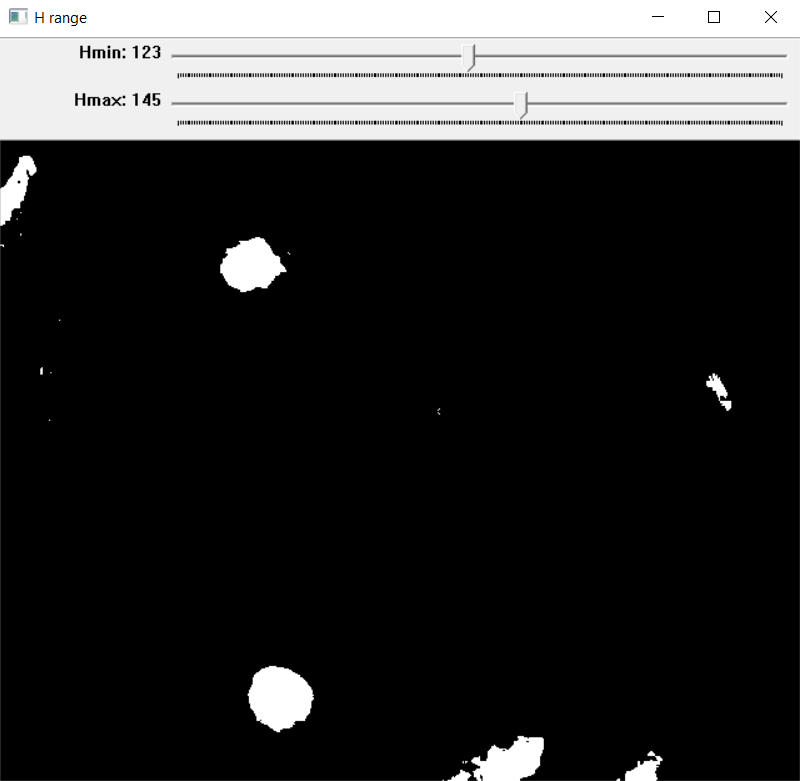


Рисунок 4.7– Приклад знаходження H

Ми виставляємо повзунки таким чином щоб знайти наші клітини. Також ми повинні це зробити на S та V.

Приклад знаходження S представлено на рисунку 4.8



Рисунок 4.8 – Приклад знаходження S

Приклад знаходження V представлено на рисунку 4.9

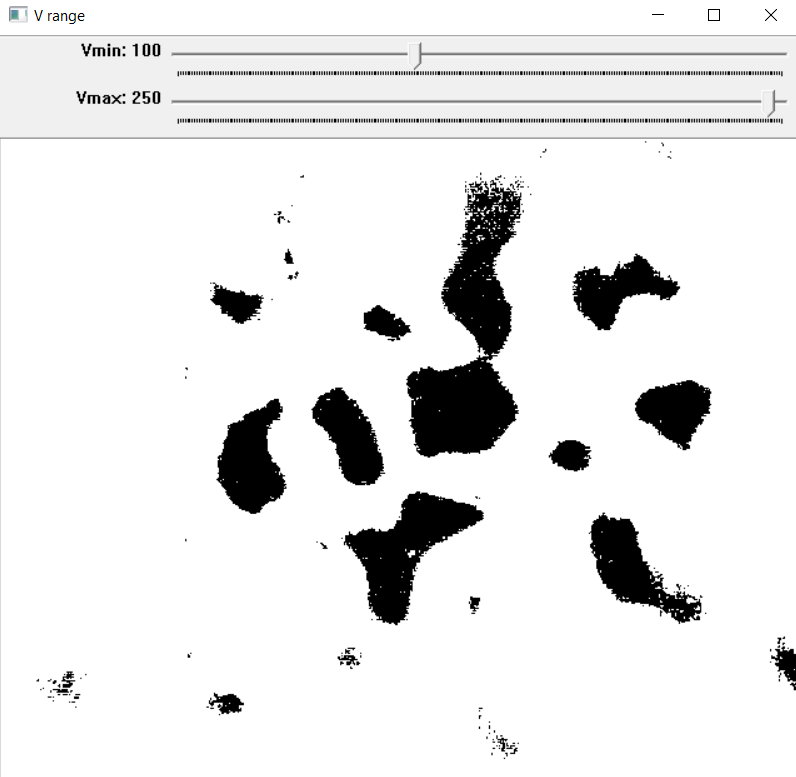


Рисунок 4.9 – Приклад знаходження V

У той час як ми зміняємо параметри, на карті HSV змінюється зображення, таким чином ми можемо переконатися що ми будемо шукати само те що треба у програмі пошуку клітин.

Карта пошуку клітин представлена на рисунку 4.10.

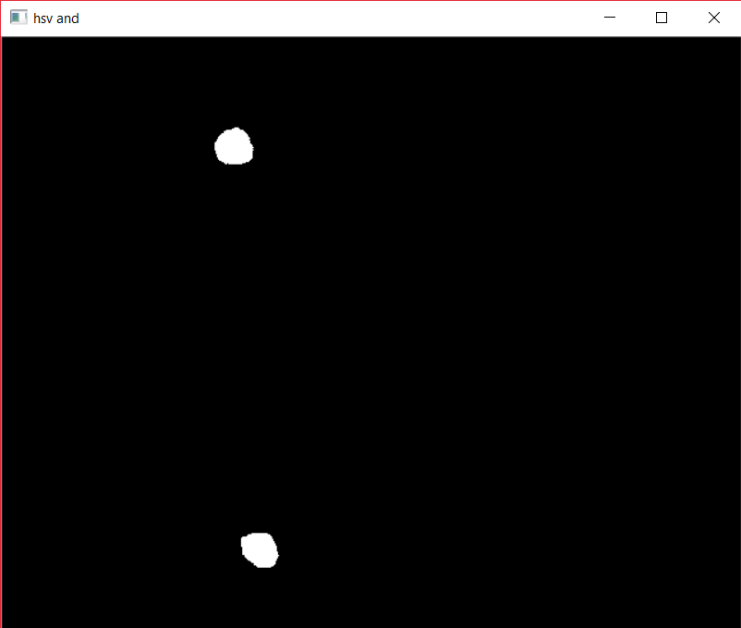


Рисунок 4.10 – Карта пошуку клітин

Після знаходження усіх мін – макс значень можна заповнювати скалярне значення для пошуку клітини у програмі пошуку клітин.

inRange(hsvImage, cv::Scalar(122, 140, 100), cv::Scalar(145, 255, 255), upperRange);

Карта пошуку клітин дозволяє легко детектувати лімфоцити, та знайти їх на зображенні для наступного аналізу.

Вона дозволяє ігнорувати «сміття» та тому подібний шум, та погану якість зображення.

Після того як програма закінчила свою роботу вона звільнює усі ресурси, щоб не викликати помилок.

cvReleaseImage(&image);

cvReleaseImage(&hsv);

cvReleaseImage(&h\_plane);

cvReleaseImage(&s\_plane);

cvReleaseImage(&v\_plane);

cvReleaseImage(&h\_range);

cvReleaseImage(&s\_range);

cvReleaseImage(&v\_range);

cvReleaseImage(&hsv\_and);

cvDestroyAllWindows();

4.4 Вимоги до програмного і апаратного забезпечення

Мінімальні вимоги, яким повинен відповідати комп'ютер для того, щоб на

нім працювала дана програма, наступні:

* процесор: AMD або Intel з тактовою частотою від 1000МГц і вище;
* об'єм оперативної пам'яті: 128 МБ і вище;
* 100 Мб простору на жорсткому або гнучкому диску для самої виконуваної
* відео карта на 8 Мб;
* операційна система Windows ХР і вище, UNIX, MacOS.

Оптимальні вимоги, яким повинен відповідати комп’ютер для того, щоб на нім працювала дана програма, а користувач відчував себе комфортно, наступні:

* процесор: AMD або Intel з тактовою частотою від 2000 МГц і вище;
* 512МБ оперативної пам’яті;
* монітор з діагоналлю 17 і вище;
* 100 Мб простору на жорсткому або гнучкому диску для самої виконуваної

відео карта на 32 Мб і вище.

5 ТЕСТУВАННЯ

5.1 Типи помилок

Розробка будь-якої програми передбачає наявність помилок у вихідному тексті і боротьбу з ними. Помилки зазвичай підрозділяють на три групи:

* синтаксичні помилки;
* помилки часу виконання програми (run-time-errors);
* смислові (логічні) помилки.

Синтаксичні помилки це найпростіші помилки, які легко усуваються вже на етапі компіляції. Причина їх один неправильний запис службових слів, операторів і тому подібне. Якщо при компіляції у вихідному тексті програми виявляються синтаксичні помилки, то їх список виводиться на панель повідомлень в редакторові коди. При цьому курсор поміщається в те місце, де виникла перша помилка, і цей рядок виділяється кольором. Опис кожної помилки в списку включає повне ім'я модуля і номер рядка, в якому виявлена помилка, а також коротке пояснення причини.

Помилки часу виконання (run-time-errors) дають про себе знати винятковими ситуаціями, які припиняють роботу ПЗ. При стандартній обробці цих виняткових ситуацій видається повідомлення, в якому вказується тип помилки, адреса, по якій вона сталася, і інша інформація.

Смислові (логічні) помилки найскладніші і важковловимі. Вони виявляються в тому, що програма робить не те, що треба, або робить те, що не треба. Наслідок логічних помилок можуть бути самими різними: неправильний вміст вікна, невиконання або невірне виконання команд користувача, неправильний вміст вихідних файлів, може виникнути помилка часу виконання і багато що інше. На боротьбу з логічними помилками уходить майже весь час налагодження.

Процес цілеспрямованого виявлення помилок інакше називають тестуванням. Деякі помилки виявляються після першого ж запуску ПЗ і помітні неозброєним поглядом; для їх виявлення, не треба удаватися ні до яких спеціальних засобів. Проте деякі помилки виявляються в абсолютно випадкові моменти роботи додатка. З такими помилками впоратися найважче. Якщо не можна зафіксувати умови виникнення помилки, не можна зрозуміти причину помилки і усунути її.

5.2 Етапи налагодження

Налагодження програми є досить складним процесом виявлення та виправлення помилок, який зазвичай вимагає послідовного виконання чотирьох етапів:

* виявлення помилки;
* локалізація помилки в тексті програми;
* встановлення причини помилки;
* виправлення помилки.

У конкретних ситуаціях перераховані етапи можуть перетинатися, деяких з них може і не бути, однак у загальному випадку справа йде саме так.

Помилки етапу виконання або семантичні помилки відбуваються, коли ви компіліруете повну програму, яка при її виконанні робить щось неприпустиме. Тобто, програма містить допустимі оператори, але при їх виконанні щось відбувається невірно. Наприклад, програма може намагатися відкрити для введення неіснуючий файл або виконати ділення на нуль.

Семантична налагодження - це процес знаходження і виправлення помилок, пов'язаних з неправильним зазначенням логічних сторінок даних.

Існує 3 способи налагодження програми:

* покрокова налагодження програм із заходом в підпрограми;
* покрокова налагодження програм з виконанням підпрограми як одного оператора;
* виконання програми до точки зупинки.

Покрокова налагодження програм полягає в тому, що виконується один оператор програми і, потім контролюються ті змінні, на які повинен був впливати даний оператор.

Якщо в програмі є вже налагоджені підпрограми, то підпрограму можна розглядати, як один оператор програми і скористатися другим способом налагодження програм.

Якщо в програмі існує досить велика ділянка програми, вже налагоджений раніше, то його можна виконати, не контролюючи змінні, на які він впливає. Використання точок зупинки дозволяє пропускати вже налагоджену частина програми. Точка зупинки встановлюється в місцях, де необхідно перевірити вміст змінних або просто проконтролювати, чи передається управління даного оператора.

6 ОХОРОНА ПРАЦІ

Робота фахівця по експлуатації та ремонту ЕОМ пов'язана із впливом на його організм ряду негативних виробничих факторів: мала рухова активність, велике навантаження на зір електромагнітні випромінювання різних частот, шум, підвищена іонізація повітря та ін. Все це за певних умов може завдати необоротної шкоди здоров'ю працівника. Щоб уникнути цього необхідно виконувати всі правила безпеки і санітарно-гігієнічні нормативи, які будуть представлені в даному розділі.

6.1 Вимоги до організації робочого місця з обслуговування, ремонту та налагодження ЕОМ

Організація робочого місця з обслуговування, ремонту та налагодження ЕОМ повинна забезпечувати відповідність усіх елементів робочого місця та їх розташування ергономічним вимогам відповідно до ГОСТ 12.2.032–78 “ССБТ Рабочее место при выполнении работ сидя. Общие эргономические требования“, характеру та особливостей трудової діяльності.

Площа робочого місця з обслуговування, ремонту та налагодження ЕОМ має бути не меншою 10 м2, робочі місця повинні бути відокремлені одне від одного перегородками.

Робоче місце з обслуговування, ремонту та налагодження ЕОМ повинно перебувати на відстані не менше 1 м від приладів опалення.

Настили (кришки) робочих столів або стендів мають бути вкриті струмонепровідними матеріалами та не мати металевої обшивки.

Робоче місце з обслуговування, ремонту та налагодження ЕОМ має бути обладнане пристроєм, що забезпечує зберігання та розміщення інструменту та матеріалів, потрібних для виконання робіт, а також збирання відходів виробництва.

Робоча поверхня столів, а також поверхня ящиків для зберігання інструменту повинна бути вкрита гладеньким матеріалом, що легко може бути помитий.

Робочі місця з обслуговування, ремонту та налагодження ЕОМ, на яких може проводитися паяння, зачищення ізоляції проводів об­палюванням, крім того, повинні відповідати вимогам СП 952–72, або ж робота на них має виконуватися з використанням електроінструменту (паяльника) з обладнаним відсмоктувачем.

За недостатнього загального освітлення робоче місце з обслуговування, ремонту та налагодження ЕОМ повинно бути обладнане місцевим освітленням (стаціонарним або переносним).

На робочому місці з обслуговування, ремонту та налагодження ЕОМ мають бути передбачені штепсельні гнізда та електророзетки для підключення електроінструменту на напругу 12В і 36В, підставки для паяльника з лотком, який запобігав би попаданню припою, флюсу та нагару на поверхню столу.

Для підключення ЕОМ, устаткування для обслуговування, ремонту та налагодження ЕОМ до електромережі на робочому столі або стенді має бути змонтований в зручному та безпечному місці, що закривається, електрощит із електроізоляційного матеріалу.

Над гніздами електрощита має бути чітко вказана номінальна напруга, а також зроблені написи, що відповідають включеному та відключеному положенню комутаційних пристро­їв та клемі “земля”.

Захисні засоби (діелектричні рукавиці, ізолювальні підставки, інструмент з ізольованими ручками тощо) мають бути наявними на підприємстві в достатній кількості згідно з чинними нормативами та задовольняти вимоги ПТЕ, ПБЕ та ПВЕ.

Ізолювальні засоби захисту необхідно зберігати в спеціально відведеному місці та періодично проводити їх випробування в установлені терміни згідно з чинними правилами користування засобами захисту, що застосовуються в електроустановках.

6.2 Раціональне та профілактичне харчування

Відомо, що здоров'я людини, її працездатність, активне розумове та фізичне довголіття вагомо залежить від правильного харчування. Воно повинно бути організовано таким чином, щоб забезпечувати нормальний розвиток та злагоджену роботу всього організму. Для цього раціон харчування має бути збалансованим як за кількісними, так і за якісними показниками залежно від потреб людини, які в значній мірі визначаються характером виконуваної роботи. Трудова діяльність користувачів ВДТ характеризується малими енергетичними витратами на фоні значного розумового та нервово-емоційного напруження. З огляду на це, їх їжа повинна бути калорійно обмеженою, однак якісною та повноцінною.

Харчування користувачів ВДТ має бути не лише раціональним, а й профілактичним. Основу профілактичного харчування складають продукти, багаті вітамінами А, В1, В2 та В12, які мають винятково важливе значення для нормального функціонування зорового аналізатора. Вітамін А (ретинол) необхідний для утворення в сітківці світлочутливої речовини. Ретинол є лише в тваринних продуктах, а його провітамін (каротин) — в рослинах. При нестачі вітаміну А в їжі знижується адаптаційні властивості ока, згодом розвивається "куряча сліпота", стає запальною і мутніє рогівка ока. Найбагатшим джерелом вітаміну А є печінка тварин, вершкове масло, жовтки яєць, молочний жир.

В зелених і оранжевих частинах рослин є каротин, який перетворюється в організмі у вітамін А. Особливо багаті на каротин червона морква, червоний перець, шпинат, абрикоси, зелений горошок та інші. Виготовлення страв з подрібнених продуктів у легкорозчинному жирі, наприклад олії (салат) в 4—5 разів покращує всмоктування каротину в кишківнику. Добова потреба у вітаміні А для дорослих 1—1,5 мг, а для користувачів ВДТ, особливо літніх — 1,5—2 мг. Половину необхідного вітаміну А можна замінити каротином.

Нестача вітамінів групи В призводить до порушення функцій нервів ока (в тому числі головного оптичного нерва), знижує прозорість оболонок ока тощо. Вітамін В1 (тіамін) міститься в дріжджах, особливо в сухих пивних (5 мг в 100 г продукту), хлібному квасі, зернових та бобових культурах (жито, пшениця, кукурудза, соя, квасоля, горох та ін). Тіамін знаходиться, в основному, в зародку зерна та його оболонках (висівках). Тому нестача вітаміну В1 часто зустрічається у людей, які вживають у їжу вироби лише з пшеничного борошна вищих гатунків. З продуктів тваринного походження найбільше тіаміну є в печінці та нежирній свинині (наприклад, в свинині його у 8 разів більше, ніжу яловичині)

Вітамін В2 (рибофлавін) здійснює регулюючу дію на стан центральної нервової системи, впливає на процеси обміну в рогівці, кришталику та сітківці ока, забезпечує світловий та кольоровий зір. Основним джерелом рибофлавіну є яйця, сир, молоко, м'ясо, а також зернові та бобові культури: арахіс, соя, сочевиця, зелений горошок. Нестача вітаміну В2 частіше зустрічається у людей, які не вживають молока та молочних продуктів, а лише хлібні вироби з вищих сортів борошна вищих гатунків.

Вітамін В12 (ціанокобаламін) належить до речовин з високою біологічною активністю. Він бере участь у синтезі метіоніну, нуклеїнових кислот, процесах кровотворення тощо. Нестача вітаміну В12, як правило, розвивається при порушенні його всмоктування і проявляється тяжкими формами анемії. Основним джерелом ціанокобаламіну є продукти тваринного походження, особливо багато його у воловій печінці.

Таким чином, раціональне та профілактичне харчування допоможе зоровому аналізатору та й всьому організму користувача вцілому, нормально виконувати необхідні функції[20].

6.3 Міжнародні нормативні документи з охорони праці користувачів комп'ютерів

Питаннями охорони праці в міжнародному масштабі та розробкою конвенцій, рекомендацій з різних соціально-правових проблем займається Міжнародна Організація праці (МОП). За роки діяльності МОП, підхід до проблеми охорони праці який, базувався на першочерговій увазі до випадків найбільш серйозних порушень і галузей з найвищим рівнем травматизму та захворюваності, значно розширився і переріс у всеосяжну систему, метою якої є досягнення найвищого рівня безпеки та гігієни праці у всіх галузях та професіях. Все вищесказане в повній мірі стосується користувачів комп'ютерів, з огляду на значний ріст цієї професії у всьому світі.

Розробкою загальних єдиних нормативних документів для користувачів ВДТ займаються кілька міжнародних організацій:

International Organization for Standardization (ISO) — міжнародна організація з стандартизації;

The ergonomics committee in ISO (TC 159) — комітет з ергономіки  
міжнародної організації з стандартизації;

European Standardization Organization (CEN) — європейська організація з стандартизації.

Серед низки розроблених нормативних документів щодо використання ВДТ найбільш часто використовуються наступні стандарти:

BS 7179 та ISO 9241-3 (більш сучасніший), які регламентують ергономічні вимоги стосовно умов праці та охорони здоров'я користувачів;-

ISO 9001, який визначає якість та рівень виробництва апаратури; :

ISO DIN 9995, який встановлює принципи розміщення елементів клавіатури для роботи з текстом в офісних системах;

ІЕС 950, який визначає норми безпеки електротехнічного обладнання:

На сьогодні найбільш новим нормативним документом, що регламентує роботу користувачів ВДТ є ISO 9241, який складається з наступних частин:

* загальний вступ;
* порадник з використання;
* вимоги для представлення даних;
* вимоги до клавіатури;
* вимоги до розміщення обладнання на робочому місці;
* вимоги до навколишнього середовища;
* вимоги до відблиску екрана;
* вимоги до кольорового зображення;
* вимоги до неклавіатурних пристроїв введення;
* принципи організації діалогу;
* порадник з використання ВДТ;
* вимоги до представлення інформації;
* порадник для користувача;
* діалогове меню;
* діалогові команди;
* управління діалогом;
* форма наповнення діалогів.

6.4. Ефективні принципи професійного відбору операторів ВДТ

На сучасному етапі розвитку НТП надійність та продуктивність функціонування комп'ютеризованих систем значною мірою залежить від правильного відбору та навчання фахівців. Завдання професійного відбору є визначення людей, які здатні найефективніше виконувати відповідні професійні завдання, використовуючи при цьому певні технічні засоби.

Для операторів ВДТ основним змістом роботи є складний комплекс розумової діяльності з невеликими елементами фізичної праці. В основі взаємодії оператора з апаратурою є приймання, опрацьовування інформації, прийняття та реалізація рішення.

Важливою особливістю людини-оператора є те, що оптимальне опрацювання інформації, яка надходить до нього, відбувається у тому випадку, коли вона за обсягом узгоджена з його можливостями сприйняття та не є занадто великою і занадто малою. При великому навантаженні оператори не встигають виконувати задані функції, але коли навантаження дуже зменшується, то оператори втрачають активність.

Аналіз праці операторів ВДТ показує, що коли до цього виду діяльності залучаються люди, які не мають достатніх здібностей до такої роботи, То вони не лише значно довше та з більшими труднощами оволодівають цією спеціальністю, але й частіше роблять помилки та прорахунки. Тому при професійному відборі необхідно ретельно перевірити відповідність психо­фізіологічних властивостей кандидата до вимог, що пред'являються, до операторської діяльності.

Професійний відбір може здійснюватися наступними методами: медичним, конкурсним та інженерно-психологічним.

Під час медичного відбору враховується лише один фактор — стан здоров'я. За заключениям медичної комісії про придатність до роботи відібрані кандидати можуть призначатися на посади.

Підчас конкурсного відбору кандидати на посади операторів відбираються за результатами перевірки їх індивідуальних здібностей шляхом проведення іспитів або конкурсу документів.

Найбільше відповідає потребам професійного відбору інженерно-психологічний метод. В цьому випадку кандидати на посаду оператора відбираються з повним врахуванням антропометричних, фізіологічних, психологічних та інших даних. В основі інженерно-психологічного методу лежать два основних принципи: активність та етапність відбору.

Під активністю відбору розуміють не лише факт визначення кандидатів на посади операторів, але й удосконалення методів навчання, а також органів керування. Основні напрями: максимальне пристосування органів керування та робочого місця оператора до функціональних характеристик людини, раціональна автоматизація керування; розробка алгоритмічних систем навчання, оптимізація режимів тренувань, застосування засобів покращення функціональних характеристик людини згідно з особливостями окремого кандидата на професію.

Під етапністю відбору розуміють послідовність проведення цієї роботи. Найбільш широко застосовується трьох етапний відбір.

Перший етап — відбір за висновками медичних комісій. Його основне завдання полягає в тому, щоб виключити осіб, які за станом здоров'я не можуть виконувати ті чи інші функціональні обов'язки оператора.

Під час другого етапу з'ясовується ступінь придатності тієї чи іншої людини до виконання даних професійних обов'язків.

Третій етап відбору є контролюючим. В його завдання входить:

* своєчасне виявлення між працюючими спеціалістами і тими, що навчаються, осіб, які не можуть ефективно виконувати свої функціональні обов'язки;
* розробка методики навчання, адаптованої до даного кандидата (групи кандидатів).

Розглянемо детальніше порядок відбору операторів на другому та третьому етапі. Завданням цих двох етапів є відбір кандидатів в оператори, які:

* мають здобуті до початку роботи операторами навички, що впливають на оволодіння спеціальністю оператора;
* характеризуються потрібною швидкістю отримання навичок оператора.

При цьому завдання другого етапу полягає в тому, щоб звести до мінімуму контингент осіб, яких залучають до відбору на апаратурі.

Завдання третього етапу полягає в тому, щоб з високим ступенем імовірності вирішити питання придатності до праці існуючої спеціальності кандидатів, які пройшли другий етап відбору.

На другому етапі відбір кандидатів для роботи за даною спеціальністю проводиться за наслідками обстеження за допомогою психофізіологічних тестів. Залежно від якості виконання кожного тесту кандидат отримує окремий бал. Сума балів, одержаних при виконанні усіх тестів, є критерієм можливості використання кандидата для роботи за спеціальністю.

На третьому етапі основною є оцінка діяльності кандидатів за результатами їх праці в ситуаціях, характерних для реальних умов праці. На цьому етапі і фіксуються дані, які дозволяють зробити кінцевий відбір за можливостями кандидата в отриманні необхідних знань.

З цією метою знаходять коефіцієнт навчання кандидата та початковий рівень підготовки до роботи за даною спеціальністю.

Усі кандидати проходять короткий курс навчання на місцях праці. Вони знайомляться з особливостями робочого місця, обов'язками, порядком роботи та правилами безпеки. Після короткочасного навчання проводиться контрольний іспит, за результатами якого і робляться висновки.

Профорієнтаційна робота у вузах і середніх спеціалізованих навчальних закладах являється ефективним способом підбору кадрів. Її завдання полягає в наступному: приходити до вищих навчальних закладів, проводити факультативні заняття. Підприємства проводять тренінги, різні практики для підготовки учнів до роботи з ними. Допомагаючи визначиться їм з майбутньою професією, самих успішних і зацікавлених роботодавець запрошує до себе на роботу отримуючи таким чином вже готових до роботи на їхньому підприємстві фахівців.

При необхідності формування бригади чи зміни, доцільно підбирати кандидатів з врахуванням психологічної сумісності, яка передбачає оптимальне поєднання людей. В основу покладено спільність мети, смаків, звичок, подібність динамічної спрямованості емоційно-вегетативних реакцій та ін.

Професійна сумісність передбачає зарахування до складу групи осіб, на підготовку яких потрібні майже однакові витрати часу та засобів.

Психологічна сумісність передбачає встановлення між особами групи відносин, які базуються на взаємодопомозі та увазі один до одного. Добра психологічна сумісність індивідуумів у колективі досягається, як правило, в тому випадку, коли рівень психомоторної та розумової діяльності у них достатньо великий і рівнозначний. Задовільна психологічна сумісність є також при сумісній діяльності осіб, які мають високу швидкість думки та слабке моторне реагування, з особами, які мають лише швидке моторне реагування.

Для формування емоційної стійкості у колективі при розподілі функціональних обов'язків між операторами за ступенем відповідальності потрібно призначати на відповідальніші посади тих операторів, які мають вищу емоційну стійкість. Доцільно у складі групи мати декілька операторів **з** високою емоційною стійкістю, які могли б при виникненні стресової ситуації і психологічної нестійкості частини операторів зберігати здатність до правильного прийняття рішення та оперативного втручання. Правильне та розумне виконання вимог щодо професійного добору дозволяє відібрати ' та навчити спеціалістів, які спроможні забезпечити найвищу працездатність та необхідну надійність роботи.

Дотримання правил безпеки при роботі з комп'ютером дозволяє не тільки зберегти здоров'я спеціалістів компьютерщиків, але і підвищити продуктивність їх праці, зменшивши стомлення від тривалої взаємодії з технікою[20,21].

7 ЕКОНОМІЧНЕ ОБГРУНТУВАННЯ ПРОГРАМНОГО ПРОДУКТУ

7.1 Загальна характеристика проекту

Порівняльний техніко-економічний аналіз проекту розробки програмного забезпечення.

В даному дипломному проекті розробляеться програмний продукт детектування лімфоцитів на зображенні, який цілком задовольняє поставленим вимогам.

Організаційне обґрунтування проекту містить такі складові частини.

Класифікаційна оцінка проекту:

* клас - моно проект;
* тип - соціальний;
* вид - учбово-освітній;
* тривалість - середньостроковий;
* рівень - галузевий.

Визначення мети і результатів проекту за схемою:

мета: розробка програмного забезпечення для детектування клітин, у віртуальному мікроскопі;

результати: результатом цієї роботи е програмний продукт детектуючий клітини .

Створений програмний продукт дозволить швидко та якісо детектувати клітини у мазку крові, у віртуальному мікроскопі;

Етапи виконання:

* постановка задачі;
* збір інформації;
* розробка алгоритму програми;
* створення програмного засобу;
* відладка;

Область використання - сфера медицини;

Коло споживачів - необмежений.

7.2 Розрахунок трудомісткості

Організація і планування процесу розробки

Розробка програмної продукції е складним і тривалим процесом, який потребує виконання великої кількості різноманітних операцій.

Трудомісткість розробки програмної продукції залежить від ряду чинників, основними з яких е такі: ступінь новизни програмного комплексу, що розробляється, складність алгоритму його функціонування, обсяг інформації, що використовується, вид її подання і спосіб обробки, а також рівень алгоритмічної мови програмування.

За ступенем новизни програмна продукція може бути віднесена до групи В- розробка програмної продукції, що має аналоги.

За ступенем складності алгоритму функціонування ПП може бути віднесена до 3 групи складності - ПП, яка реалізує алгоритми стандартних методів вирішення задач.

Від виду подання вхідної інформації, способу її контролю і структури вхідних документів відноситься до групи 12 (вхідна інформація подана в формі зображень мазка крові, та клітин).

Функціональне призначення – медичний аналіз.

Трудомісткість розробки програмної продукції () може бути визначена за формулою:

(7.1)

де - трудомісткість розробки технічного завдання на створенняПП;

- трудомісткість розробки ескізного проекту ПП;

- трудомісткість розробки технічного проекту ПП;

- трудомісткість розробки робочого проекту ПП;

- трудомісткість впровадження ПП.

Трудомісткість розробки технічного завдання розраховується за формулою:

(7.2)

де - витрати часу розробника постановки задач на розробку ТЗ, люд. - дні;

- витрати часу розробника ПЗ на розробку ТЗ, люд. - дні. Величини t3P3 та t3p розраховуються за формулами:

(7.3)

де t3 – норма часу на розробку ТЗ на ПП в залежності від функціонального призначення і ступеня новизни ПП, що розробляється, люд. – дні.

t3 = 43 (люд. - дні);

- коефіціент, який враховуе питому вагу трудомісткості робіт, що виконуються розробником постановки задач на стадії ТЗ (в разі спільної з розробником ПЗ розробки = 0,65)

(7.4)

де - коефіцієнт, який враховує питому вагу трудомісткості робіт, які виконуються розробником програмного забезпечення на стадії ТЗ (в разі спільної з розробником постановки задач = 0,35).

(7.5)

Трудомісткість розробки ескізного проекту ПП () розраховується за формулою:

(7.6)

де - витрати часу розробника постановки задач на розробку ЕП, люд. — дні; - витрати часу розробника ПЗ на розробку ЕП, люд. — дні.

Величини та розраховуються за формулами:

де — норма часу на розробку ескізного проекту ПП в залежності від його функціонального призначення і ступеня новизни, te = 30 (люд. — дні);

— коефіцієнт, який враховує питому вагу трудомісткості робіт, які виконуються розробником постановки задач на стадії ЕП (в разі спільної з розробником ПЗ розробки ЕП = 0,7).

(7.8)

де — коефіцієнт, який враховує питому вагу трудомісткості робіт, які виконуються розробником програмного забезпечення на стадії ЕП (в разі спільної з розробником постановки задач розробки ЕП = 0,3.

люд.-днів (7.9.)

Трудомісткість розробки технічного проекту () залежить від функціонального призначення ПП, кількості різновидів форм вхідної та вихідної інформації і визначаеться за формулою:

(7.10)

де - норми часу, який витрачається на розробку ТП розробником постановки задач і розробником ПЗ відповідно, люд.-дні.  
 Вважається, що у даному випадку існує один різновид форм вхідної інформації (введення даних у таблиці БД за допомогою клієнтського додатку) та два різновиди вихідної інформації

=10(люд.-дні), =30 (люд.-дні);

— коефіцієнт врахування виду інформації, що використовується;

— коефіцієнт врахування режиму обробки інформації, =1,26. Значення коефіцієнта визначаеться за формулою:

(7.11)

де , , – коефіцієнти врахування виду інформації для змінної, нормативно-довідкової інформації і баз даних відповідно;

, , - кількість наборів даних змінної, нормативно-довідкової інформації, баз даних відповідно.

(7.12)

Трудомісткість розробки робочого проекту залежить від функціонального призначення ПП, кількості різновидів форм вхідної і вихідної інформації, складності алгоритму функціонування, складності контролю інформації, ступеня використання готових проrрамних модулів, рівня алгоритмічної мови програмування і визначається за формулою:

(7.13)

де – коефіцієнт врахування складності контролю інформації (Ступінь складності контролю вихідної інформації - 22); = 1,07;

– коефіцієнт врахування рівня алгоритмічної мови програмування (Алгоритмічна мова високого рівню (типу PL 1)); = 1,00;

– коефіцієнт врахування ступеня використання готових програмних модулів (20-25%); кз = 0,8;

– коефіцієнт врахування виду інформації, що використовуеться, і складності алгоритму ПП.

Коефіцієнт розраховується за формулою:

(7.14)

де , , – значення коефіцієнтів врахування складності алгоритму ПП та виду інформації, що використовується, для змінної, нормативно- довідкової інформації і баз даних відповідно;

, — норми часу на розробку РП на алгоритмічній мові високого рівня (типу PL / І) розробником постановки задачі і розробником ПЗ відповідно, люд. — дні;

, = 18; = 65 (люд.- дні). Р

(7.15)

Трудомісткість виконання стадії `Впровадження" () може бути розрахована за формулою:

(7.16)

де – норма часу на виконання процедур впровадження розробником задач і розробником програмного забезпечення відповідно люд.-дні;

(7.17)

Трудомісткість розробки програмної продукції:

(7.18)

7.3 Визначення ціни програмного продукту

Програмна продукція може розглядатися як продукція виробничо- технічного призначення, яка допускає багаторазове їі тиражування і відчуження від безпосередніх розробників. Її ціна визначається за формулою:

(7.19)

де С – витрати на розробку програмної продукції (кошторис витрат);

- нормативний прибуток, який розраховується за формулою:

(7.20)

де – норматив рентабельності, %;

= 25%;

- матеріальні витрати, грн.

Витрати на розробку ПП можуть бути представлені у вигляді кошторису, який містить такі статті: матеріали; спеціальне обладнання; основна заробітна плата; додаткова заробітна плата; відрахування на соціальні заходи; виробничі відрядження; накладні витрати; контрагентські витрати.

Розрахунок кошторису на розробку ПП здійснюється за кожною статтею витрат і результати зводяться у таблицю.

Матеріали. В статті враховуються підсумкові витрати на матеріали, які придбаються для розробки даної ПП. Вони складаються з вартості матеріалів і транспортно-заготівельних витрат:

(7.21)

де – коефіцієнт транспортно-заготівельних витрат ( = 0,03 - 0,05) – ціна одиниці і-го матеріалу, грн.;

– обсяг (кількість) і-го матеріалу.

Розрахунки за цією статтею можуть бути представлені у вигляді таблиці 7.1– Розрахунок витрат на матеріали.

Таблиця 7.1 – Розрахунок витрат на матеріали

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| №  п/п | Найменування матеріалу | Одиниця вимірювання | Кількість | Ціна за одиницю, грн | Сума, грн |
| 1 | Диск | шт. | 1 | 7 | 7 |
| 2 | Програмне забезпечення | шт. | 1 | 60 | 60 |
| 3 | Ручка | шт. | 1 | 4 | 4 |
| 4 | Листи А4 | уп. | 1 | 35 | 35 |
| Сума | | | | | 106 |

За неможливістю визначення витрат на матеріали методом прямого розрахунку їх враховують в накладних витратах.

(7.22)

Спеціальне обладнання. Спеціального обладнання даний ПП не потребує, відповідно витрати по цій статі дорівнюють нулю.

Основна заробітна плата. В статтю включається вся заробітна плата всіх виконавців, які безпосередньо зайняті розробкою даного ПП, з урахуванням їхнього посадового окладу і часу участі в розробці. Розрахунок проводиться за формулою:

СЗО=(Зі∗τіа) (7.23)

де – середньомісячний оклад і-го виконавця, грн.; = 2000 грн;

– середня кількість робочих днів в місяці; а= 22 дні;



– трудомісткість робіт, які виконуються і-им виконавцем, люд.дні.

СЗО=(Зі∗τіа)=2000∗27922=25363 (7.24)

Додаткова заробітна плата. В статті враховуються всі виплати безпосереднім виконавцям за час (встановлений законодавством), який не відпрацьований на виробництві, в тому числі: оплата чергових відпусток, компенсації за невикористані відпустки, оплата пільгових годин підліткам та ін. Розрахунок проводиться за формулою:

(7.25)

де – коефіцієнт додаткової заробітної плати (= 0,4).

(7.26)

Відрахування на соціальні заходи. В статті враховуються відракування на соціальні потреби у відповідності до діючого законодавства:

(7.27)

де – коефіцієнт відрахування на соціальні заходи (пенсійний фонд, фонди страхування).

(7.28)

Виробничі відрядження. Витрати на відрядження, пов'язані з розробкою даної ПП не потрібні.

Контрагентські витрати. В статті враковуються витрати на виконання сторонніми організаціями робіт, які безпосередньо пов'язані з розробкою даного ПП. В зв'язку з тим, що сторонніми органїзаціями не було виконано ніяких робіт стосовно даного ПП, витрати за ціею статтею відсутні.

Накладні витрати. В статті враховуються витрати на загальногосподарські потреби, невиробничі витрати, витрати на управління. Накладні витрати, визначаються за формулою:

(7.29)

де - коефіцієнт накладних витрат, = 0,05;

(7.30)

Таблиця 7.2 – Кошторис витрат на розробку ПП

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Найменування статті | Витрати статтею | Питома вага, % |
| 1 | Матеріали | 106 | 0.2 |
| 2 | Основна заробітна плата | 25363 | 41.3 |
| 3 | Додаткова заробітна плата | 10145 | 16.5 |
| 4 | Відрахування на соціальні заходи | 13138 | 21.4 |
| 5 | Накладні витрати | 1268 | 20.6 |
| Всього |  | 61433 | 100 |

Розраховуємо нормативний прибуток:

(7.31)

де = 20% - норматив рентабельності;

Ціна програмного продукту:

(7.32)

5.4 Розрахунок початкових інвестицій

Розрахунок початкових інвестицій відбувається за формулою:

(7.33)

де – передвиробничі витрати;

– вартість комп'ютерного обладнання;

– вартість допоміжного обладнання, яке необхідне для надійної роботи ІС (КМ);

– вартість будівництва (реконструкції) у зв'язку з впровадженням (удосконаленням) ІС (КМ);

– доход від продажу обладнання, що замінюється на нове, з урахуванням витрат на його реалізацію;

– налог на прибуток від продажу обладнання, що замінюється на нове.

Передвиробничі витрати () містять усі видатки, що пов'язані з проектуванням, розробкою, налагодженням і впровадженням ІС (КM) - передпроектні і проектні дослідження, постановка завдань та їхня алгоритмізація, розробка, налагодження та впровадження програмного забезпечення (ПЗ), навчання обслуговуючого персоналу, перепідготовка частини персоналу підприемства та інш. В основному ці витрати спрямовуються на розробку П3, тому близько 90% їх загальної величини складае ціна програмного продукту.

Витрати на комп'ютерне та додаткове обладнання не потрібні, адже система використовує існуюче обладнання.

(7.34)

(7.35)

Розрахунок поточних витрат.

Розрахунок річних поточних витрат, які пов'язані с функціонуванням ІС (КM) відбувається за формулою:

(7.36)

де – фонд основної і додаткової оплати праці;

– відрахування на соціальні заходи;

– вартість електроенергії;

– вартість ремонту основного та допоміжного обладнання;

– вартість допоміжних матеріалів, приладів, які пов'язані з експлуатаціею ІС (КМ);

– накладні витрати.

Річний фонд основної і додаткової оплати праці визначаеться за формулою:

(7.37)

де – річний фонд основної заробітної плати персоналу, який обслуговуе ІС (КМ);

– коефіцієнт додаткової заробітної плати (визначається за даними підприємства або в розмірі 0,4);

(7.38)

де – чисельність спеціалістів і-ої категорії, які обслуговують ІС

– місячний оклад спеціаліста і-ої категорії;

К- рівень завантаженості співробітника, К=0,2.

(7.39)

(7.40)

(7.41)

– коефіцієнт відрахувань в фонди соціальних заходів (пенсійний, соціального страхування і т.ін.). Визначається у відповідності з діючим законодавством.

Річні витрати на електроенергію визначаються за формулою:

(7.42)

де М – потужність комп'ютерного обладнання, М= 0.6 кВт;

– річний фонд часу роботи ЕОМ з урахуванням часу профілактичних оглядів, = 8\* 22 \* 12 \* 0.2= 422 год;

– вартість 1 квт-год. електроенергії, = 0.9 грн.;

– коефіцієнт інтенсивності використання потужності (рекомендується = 0,9).

(7.43)

= 0 (грн).

= 0 (грн)

Накладні витрати:

(7.44)

= 4800 • 0,05 = 240 (грн) (7.45)

= 6720 + 2486 + 205 + 240 = 9651 (7.46)

Таблиця 7.3 – Економічні показники проекту

|  |  |  |
| --- | --- | --- |
| Показник | Одиниця вимір. | Значення |
| Трудомісткість розробки проекту | Люд.-дні. | 279 |
| Витрати на розробку ПП | грн. | 61433 |
| Ціна ПП | грн. | 76764 |
| Початкові інвестиції | грн. | 85293 |
| Річні поточні витрати | грн. | 9651 |

8 ЦИВІЛЬНИЙ ЗАХИСТ

Стратегічний напрям дій щодо реалізації державою гарантій безпечної життєдіяльності громадянам, ґрунтується на поширенні превентивних заходів безпеки. Відбувається перехід від концепції реагування на надзвичайні ситуації до створення умов їхнього запобігання. Останнє неможливо при наявності двох складових: по-перше, якщо бракує необхідних фахівців і, по-друге, — коштів.

Метою цього розділу є інформування населення щодо методів визначення існуючих характеристик надзвичайних ситуацій у аналітичному та загальному вигляді.

Задачі розділу:

* описати специфіку реалізації методики визначення величини негативного впливу надзвичайної ситуації на людину та довкілля;
* надати конкретні знання по структурісистеми управління ризиками об’єктів економіки у надзвичайних ситуаціях.

Досягнення компромісу між рівнем небезпек, які виникають внаслідок нерівно­мірності розвитку цих елементів чи розподілу ймовірності виникнення аварій і економічними можливостями сучасного суспільства. Однією з актуальних складових, які сприяють розв'язанню цього питання, є зниження ризику ПНО, котрі працюють в Україні.

Основу даної діяльності складає ризик орієнтований підхід. Основними його задачами є створення реальних наукових основ забезпечення безпеки складних технічних систем, людей і довкілля, розроблення методів оцінки небезпеки ОЕ та наукових засад концепції прийнятого ризику стосовно умов функціонування системи Л-Т-С. Ефективним напрямом на шляху зниження ризику є впрова­дження автоматизованих систем контролю, аналізу і запобігання надзвичайним подіям [22].

Опанування методології інтегрованої оцінки ризику, адаптованої до наших умов життєдіяльності, разом з якісним моніторингом небезпечних подій, дозволить не тільки порівнювати рівні ризиків від будь-яких джерел, розподіл ресурсів за АТО, але й досягти такий рівень зниження ризику, котрий є необ­хідним для забезпечення сталого розвитку нашої країни.

8.1 Методика визначення величини негативного впливу надзвичайної ситуації на людину та довкілля

Незалежно від джерела виникнення, всі надзвичайні ситуації мають практично однакові чинники негативного впливу яна людину та її життєве середовище. Це механічний (кінетичний) вплив ударної хвилі (УХ) при вибуху газоповітряних сумішей (ГПС), вибухових речовин (BP), технологічних установок, при ураженні уламками під час обвалення будинків і споруд, чи від сучасної зброї; тепловий вплив від всіляких пожеж; хімічний вплив від токсичних викидів небезпечних хімічних речовин (хімічної зброї); радіоакти­вне враження при радіаційній аварії (ядерному вибуху) і т. ін.

Встановлено низку фіксованих значень негативних чинників, які відповідають тому чи іншому ступеню враження людей, будинків і споруд, довкілля. Наприклад, при надмірному тиску на фронті УХ (ДРф), який дорівнює 70 кПа, можливі контузії людей, повне руйнування будинків, середній ступінь руйнування ліній електропостачання, сильний ступінь руйнування наземних резервуарів тощо. При концентрації токсиканта LD50 можливий летальний вихід 50 % людей, котрі підпали токсичному ураженню. У випадку теплового впливу пожежі при утворенні вогненної кулі зі щільністю теплового потоку 37 кВт/м відбувається руйнування розташованих поряд ємностей, а при тривалості експозиції 30 с буде смертельно вражено до 90 % людей [23,24].

Ефект ураження має ймовірнісний характер, оскільки будь який вплив (надмірний тиск УХ, кількість поглиненого токсиканту, теплоти, доза радіації тощо) може спричинити наслідки різної важкості у конкретних людей. Тому підхід до визначення вражаючої дії негативних чинників (ефекту ураження) має назву ймовірнісного.

Величина ймовірнісного ураження (ефект ураження) Рур (вимірюється в частках одиниці або відсотках) відображається, як

 (8.1)

 (8.2)

 (8.3)

 ( 8.4)

Величина змінюється в інтервалі від 0 до 1, наприклад при = -3, = 0,014; при = 0, = 0,5; при = 3, = 0,9986.

Таким чином можна визначити кількісні характеристики негативного впливу вражаючих чинників НС на людину і довкілля [22,24].

8.2 Особливості структури системи управління ризиками

Підхід до визначення величин чинників негативного впливу НС на населення і території на підставі аналізу кількісної оцінки ризику, особливо важливий для регіонів, в першу чергу тих, де зосереджено значний потенціал небезпечних виробництв і об'єктів з поєднанням із складною соціально-політичною обстановкою і недостатнім фінансуванням.

У загальному випадку управління ризиком — це розробка і обґрунтування оптимальних програм діяльності, спрямованих на ефективну реалізацію рішень у сфері забезпечення безпеки. Рішення приймаються в рамках системи управління ризиками. Головний елемент її діяльності — процес оптимального розподілу обмежених ресурсів на зниження різноманітних видів ризику з метою досягнення такого рівня без­пеки населення і навколишнього середовища, який тільки є мож­ливим з точки зору економічних і соціальних чинників.

Основні завдання системи:

* встановлення рівнів прийнятного ризику, виходячи з економічних і соціальних чинників, побудову механізмів державного регулювання безпеки;
* моніторинг навколишнього середовища, аналіз ризику для життєдіяльності населення і прогнозування НС;
* прийняття рішень про доцільність проведення заходів щодо захисту;
* раціональний розподіл засобів і ресурсів на превентивні заходи із зниження ризику і заходи щодо зменшення масштабів НС;
* здійснення превентивних заходів;
* проведення аварійно-рятувальних і відновних робіт у разі виникнення НС.

Для екстреного реагування, спрямованого на рятування людей, ліквідацію НС створюються, оснащуються, навчаються і утримуються в готовності до негайних дій оперативно-рятувальні, аварійно-рятувальні, відновні та пошукові формування, запаси матеріальних і фінансових ресурсів, страхові фонди тощо. Розробляються плани заходів щодо евакуації населення і його першочергового життєзабезпечення в потерпілих районах.

8.3 Методика прогностичної оцінки небезпеки об'єкту еконо­міки (ОЕ) (адміністративно-територіальної одиниці)

Комплексна оцінка техногенної та природної безпеки ОЕ (АТО) здійснюється методом системного аналізу через визначення інтегральних показників ризику. Це дозволяє визначити характе­ристики небезпек НС не тільки на рівні ОЕ, АТО, регіонів а і держави в цілому, провести їхнє ранжування та порівняльний аналіз[22,25].

Кількісна порівняльна оцінка техногенної та природної безпеки ОЕ (АТО) необхідна для управління безпекою населення шляхом надання рекомендацій щодо спрямування оптимальної кількості об'єктових, місцевих, регіональних та державних матеріальних і фінансових ресурсів на запобігання НС, зниження ризиків їхнього виникнення, розробки програм, щодо підвищення безпеки праців­ників ОЕ та населення АТО в найбільш небезпечних місцях.

Методика надає можливість працювати з неповною інформацією, разом із широким колом кількісних характеристик техногенної та природної безпеки, враховувати якісні сторони проблеми, тобто показники, котрі не мають безпосередньої числової оцінки, дозволяє обчислювати вагомі коефіцієнти окремих пока­зників, які впливають на безпеку, забезпечує динамічну співвід­ношення оцінок.

Вона реалізується за трьома етапами.

На першому***—*** відбуваються системний аналіз та структуризація проблем техногенної та природної безпеки***.***

На другому — визначення комплексних показників потенційної небезпеки ОЕ (АТО) щодо виникнення техногенних та природних НС.

На третьому — розрахунок інтегральних показників безпеки на основі комплексних показників потенційної небезпеки ОЕ, індивідуального ризику смерті та матеріального збитку.

Для визначення небезпеки ОЕ (АТО) в першу чергу розглядаються показники, пов'язані з техногенною, природною та соціально-політичною безпекою [22,25].

На засадах проведеного аналізу системи будується її ієрархічна модель, визначається множина елементів кожного рівня ієрархії, встановлюються зв'язки і зале­жності між ними. Ієрархічна модель техногенної безпеки ОЕ (АТО) складається з чотирьох основних рівнів (рис. 8.1).



Рисунок 8.1 – Ієрархічна модель гіпотетичного ОЕ (АТО)

Перший рівень ієрархії визначає загальну безпеку АТО (ОЕ) щодо виникнення НС.

Другий — містить макрообмеження: техногенні, природні, соціально-політичні НС. На другому рівні ви­значається який з перелічених ризиків більшої мірі визначає рівень безпеки, тобто відбиває ідентифікацію небезпек.

Третій — висвітлює загрози виникнення НС. На третьому рівні визначаєть­ся яка із загроз є більш вагомою щодо відповідного типу макро- обмежень. Четвертий — включає характеристики' ризиків загроз. На четвертому рівні визначається яка з них більше впливає на реалізацію загрози.

Ієрархія може будуватися з вершини, тобто з першого рівня, чи з четвертого— початкових подій, ідентифікованих заздале­гідь.

Визначення загроз виникнення НС здійснюється з урахуванням Державного класифікатора надзвичайних ситуацій (ДК 019- 2001). Для оцінки рівня техногенних небезпек ОЕ (АТО) необхідні наступні вихідні данні [24,26]:

* площа АТО (SАТО), на території якої розташовано ОЕ (визначається за табличними даними з метою з'ясування розмірів АТО, котрі позитивно чи негативно впливають на дію вражаючих чинників НС), менша площа в більшому ступені впливає на вагомість наслідків;
* площа ОЕ (SОЕ), котра визначається за відповідною документацією з метою визначення просторових розмірів об'єкту (чим більше розміри ОЕ, тим менше ймовірність того, що ситуація може вийти, за межі проммайданчика);
* чисельність виробничого персоналу (N), яка визначає можли­ву кількість постраждалих чи загиблих при НС об'єктового рівня;
* чисельність населення АТО, де знаходиться ОЕ (NАТО), тобто можлива кількість населення, яке може потрапити в зону дії вражаючих чинників НС;
* кількість ПНО, розташованих на території ОЕ (nПНО), яка свідчить про більшу, або меншу небезпеку виникнення НС (чисельність можливих НС техногенного походження знаходиться у безпосередній залежності від кількості ПНО, які знаходяться на території ОЕ;
* середнє річна кількість НС (NНС), тобто частота виникнення аварій (катастроф), стихійних лих на ОЕ;
* середнє річна кількість персоналу ОЕ, постраждалого при НС (NПЕР), підтверджує повторюваність впливу вражаючих чинників НС на персонал ОЕ;
* середнє річна кількість населення АТО, постраждалого при НС, пов'язаних з природними, техногенними та соціально- політичними небезпеками (NНАС), враховується як показник чисельності населення в зонах небезпеки (дії вражаючих чинників) для кожного виду НС;
* площа зони ураження (SУР), показник просторових розмірів впливу вражаючих чинників для кожного виду НС;
* сума збитків від НС (ZПР), показує прогнозовані матеріальні та фінансові втрати від впливу вражаючих чинників і витрати матеріальних і фінансових ресурсів під час локалізації та ліквідації НС [25,26].

На підставі вищезазначених даних визначаються додаткові відносні показники у відсотках (%), а саме:

* + відносна величина постраждало! території (Ps), яка розра­ховується за формулами:

, або  ; (8.5)

* + відносний показник постраждалого населення (PN), тобто розрахункова чисельність населення, яке може постраждати внаслідок НС відносно до загальної кількості населення;

 (8.6)

За визначеною множиною цих показників, ґрунтуючись на офіційних джерелах інформації та відповідних розрахунках формується база даних, за допомогою яких можна дослідити зміни показників, у вигляді відповідних таблиць [23,27].

8.4 Оцінка параметрів та порядок визначення коефіцієнта небезпеки об'єкту економіки

За базою даних визначаються максимальні, або мінімальні величини показників вихідних даних, в залежності від того, як останні впливають на безпеку ОЕ та АТО. Діїя визначення значущості та пріоритету вихідних даних застосовується індекс d. Його найбільше значення — d=l. Отже, всі інші показники вибра­ного значення будуть < 1. Індекс d розраховується для кожного окремого показника. Так, наприклад, на території АТО «К» знахо­дяться 10 ПНО, АТО «Л» — 6 ПНО, АТО «М» — 2 ПНО. Відпові­дно індекс d для АТО «К» дорівнює 1, для АТО «Л» d=6/10 = 0,6, для АТО «М» d = 2 / 10 = 0,2. Таким чином, індек­си значущості та пріоритету складають розрахункову базу для ви­значення сумарної прогностичної оцінки техногенно-природної небезпеки АТО. Кінцевим результатом розрахунків, виконаних за допомогою вихідних даних, є визначення порівняльного коефіціє­нту небезпеки (КНБ), показника сумарної прогностичної оцінки техногенно-природної небезпеки, тобто ступеня небезпеки відпо­відної АТО [11]. Він розраховується, ж сума індексів значущості та пріоритету за кожним з показників ОЕ за формулою:

 (8.7)

Підсумкове складання порівняльного коефіцієнту небезпеки дозволяє здійснити у першому наближені ранжування об'єктів за ступенем їхньої небезпеки. Чим більше КНБ, тим більша небезпека впливу вражаючих чинників НС на населення АТО та персонал ПНО.

Таким чином використання даного методу дозволяє визначити ступінь небезпеки відповідного ОЕ на обмежених територіях. А це у свою чергу сприяє обґрунтуванню управлінських рішень щодо розробки та реалізації превентивних заходів спрямованих на запобігання НС на конкретній АТО.

Нижче, для прикладу, наведено варіант оцінки параметрів та визначення коефіцієнту небезпеки гіпотетичної АТО де розглядаються показники, котрі пов'язані з техногенною безпекою, як однією з передбачуваних та прогнозованих. Для порівняння обрано З ПНО, ідентифікованих як:

* хімічно небезпечний об'єкт (об'єкт «А»);
* вибухо-пожеженебезпечний об'єкт (об'єкт «Б»);
* вибухонебезпечний об'єкт (об'єкт «В»).

На підґрунті проведеного аналізу системи техногенної безпеки побудовано її ієрархічну модель. З урахуванням Державного класифікатора НС (ДКО19-2001) проведено ідентифікацію основних загроз техногенній безпеці досліджуваної АТО. Серед найважливіших визначено хімічну небезпеку, вибухо-пожеженебезпеку, аварії на транспорті. До системи показників, які характеризують ризики виникнення техногенних НС на АТО, що розглядається, віднесені:

* небезпека ураження території АТО НХР;
* кількість населення, яке потрапляє у зону можливого зараження НХР, наявність систем виявлення НХР;
* небезпека загинути на АТО від пожежі чи вибуху;
* небезпека загинути та бути травмованим у дорожньо-транспортних пригодах [22,26].

За існуючими методиками та відповідними статистичними даними визначаються характеристики зон можливого ураження, кількість людей, що можуть опинитися у цих зонах, частота виникнення аварій, а на підґрунті проведеної ідентифікації— кількість та рівні ПНО, які знаходяться на території досліджуваної АТО.

Висновки. У процесі виконання даного розділу було досягнуто мети, яка полягає у інформуванні населення щодо методів визначення існуючих характеристик надзвичайних ситуацій у аналітичному та загальному виглядах. Описана специфіка реалізації методики визначення величини негативного впливу надзвичайної ситуації на людину та довкілля, наведено відповідні формули розрахунку показників, та надано конкретні знання по структурісистеми управління ризиками об’єктів економіки у надзвичайних ситуаціях.

ВИСНОВКИ

В результаті роботи над дипломним проектом, був розроблений програмний продукт для детектуваня клітин лімфоцитів у віртуальному мікроскопі.

В якості основи була використана бібліотека комп’ютерного зору Open CV. Завдяки використанню елементів вдалося розробити сучасний і водночас простий у використанні програмний продукт, який виконує чітко поставлені завдання.

Розроблен розділ «Безпека життєдіяльності», який включає в себе: вимоги до організації робочого місця з обслуговування, ремонту та налагодження ЕОМ, вимоги раціонального та профілактичного харчування, ефективні принципи професійного відбору операторів ВДТ технічні способи профілактики порушень здоров'я спеціалістів комп'ютерщиків.

Розроблен розділ «Економічне обґрунтування», який включає в себе: загальну характеристику проекту, розрахунок трудомісткості, визначення ціни програмного продукту та розрахунок початкових інвестицій.

Розроблен розділ «Цивільний захист», у якому надана інформація щодо стратегічного напряму дій щодо реалізації державою га­рантій безпечної життєдіяльності громадянам, ґрунту­ється на поширенні превентивних заходів безпеки. В результаті роботи над дипломним проектом був отриманий досвід роботи із середовищем MS Visual Studio 2015, з бібліотекою комп’ютерного зору OpenCV , а також досвід програмування на С++, що дозволить у майбутньому створювати високопродуктивні програмні продукти.

Підчас випробування программного продукта, ніяких помилок не було виявлено. Програма показала хороші результати і повністю готова до експлуатації

ПЕРЕЛІК ПОСИЛАНЬ

1. Официальный сайт Open CV [Электронный ресурс] – Режим доступа: <http://opencv.org/> – Дата доступа: 20.11.2015.
2. Montgomery John «Visual Studio 2015 Preview, Visual Studio Community 2013, Visual Studio 2013 Update 4, and More. Visual Studio Blog»/ J. Montgomery 2004. – 394p.
3. Буч, Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++ / Г. Буч; пер. с англ. И. Романовский, Ф. Андреев. - М.: Бином, 2012. – 560 с.
4. Технология Raspberry pi [Электронный ресурс] – Режим доступ:

<https://ru.wikipedia.org/wiki/Raspberry_Pi> – Дата доступа: 25.12.2015 г.

1. Дейв Эванс. «Интернет вещей. Как изменится вся наша жизнь на очередном витке развития Всемирной сети»/ под общ.ред. Э. Дейвю, 2011 г. – 500с.
2. Дейв Эванс. «Интернет вещей. Как изменится вся наша жизнь на очередном витке развития Всемирной сети», официальный документ Группа разработки интернет - решений Cisco для бизнеса (IBSG)/ Э. Дейвю., 2011 г. – 500
3. Алгулиев Расим, Махмудов Расим, «Интернет вещей», Информационное общество/Р.Алгулиев, под общ.ред. Р. Алгулиев 2013 г. – 656с.
4. Ходаковский Ф.В. Перспективы интернета вещей // Современные научные исследования и инновации. 2014. № 12 [Электронный ресурс] – Режим доступ: <http://web.snauka.ru/issues/2014/12/42161> – Дата доступа: 15.01.2016 г.
5. Державне агентство з питань електронного урядування України [Электронный ресурс] – Режим доступа: <http://dknii.gov.ua/content/elektronna-medycyna> – Дата доступа: 25.12.2015 г.
6. Гонзалес Р., Вудс Р. «Цифровая обработка изображений»: пер. с англ. М.: Техносфера / под общ. ред. Р.Гонзалес. 2005. 1072 с.
7. Цифрова патологія [Электронный ресурс] – Режим доступа: <http://referaty.in.ua/index.php?newsid=1617709> – Дата доступа: 29.01.2016 г.
8. Fourier G.H. Pre-processing for Hand Print Character Recognition/ G.H. Fourier // IEEE Transactions on Computers. 1972. Vol. C-21. P. 195-201.
9. Л. Шапиро, Дж. Стокман «Компьютерное зрение = Computer Vision. — М. : Бином. Лаборатория знаний» / под общ.ред. Л. Шапиро 2006г. — 752 с.
10. Бакут П.А., Колмогоров Г.С., Ворновицкий И. Э. «Сегментация изображений: методы пороговой обработки. // Зарубежная радиоэлектроника.» / П.А. Бакут. 1987. С. 6–24
11. Рейер И.А. Распознавание формы плоских объектов на основе гомеоморфного отображения границы / И.А. Рейер //В кн.: Труды 5-й межд. конф. «Распознавание образов и анализ изображений: новые информационные технологии». Самара, 2000.,С. 377-381.
12. Шапиро Л., Дж. Стокман «Компьютерное зрение = Computer Vision.» — М. : Бином. Лаборатория знаний, / Л. Шапиро – под общ.ред. Шапиро 2006. — 752 с.
13. Сойфер В.А. «Методы компьютерной обработки изображений» Изд. 2. М.: ФИЗМАТЛИТ/ под общ.ред. В.А. Сойфера. 2003г. – 784 с.
14. Кудрявцев Л.В. Краткий курс математического анализа – M.: Наука,/ под общ. ред. – Л.В. Кудрявцева, 1989г. – 736с.
15. Дэвид Форсайт, Жан Понс Компьютерное зрение. Современный подход = Computer Vision: A Modern Approach. — М. : «Вильямс» / Д.Форсайт – под. общ. ред.. Д.Форсайта 2004г. — 928 с.
16. Жидецкий В.Ц. «Основы охраны труда»/ В.Ц. Жидецкий, В.С. Джигирей, А.В. Мельников – под. общ. ред. В.Ц. Жидецкого. Изд-во «Питер», 2002. – 46-69 с.
17. «Про затвердження Правил охорони праці під час експлуатації електронно-обчислювальних машин» [Электронный ресурс] – Режим доступа: http://zakon3.rada.gov.ua/laws/show/z0382-99/page2 – Дата доступа: 2.11.2015 г.
18. Васійчук В.О. Основи цивільного захисту: Навч. посібник / В.О. Васійчук, В.Є. Гончарук. – Львів, 2010. – 384 с.
19. Михайлюк В.О. Цивільна безпека: Навч. Посібник / В.О. Михайлюк, Б.Д. Халмурадов – К: Центр учбової літератури, 2008.– 158 с.
20. Постник М.І. Захист населення і господарських об'єктів у надзвичайних ситуаціях. / М.І. Постник – Мн.: Виш.шк., 2003. -398с.
21. Русаловський А.В. Цивільний захист: Навч. Посібник / А.В.Русаловський – К.: АМУ, 2008, – 250с.
22. Стеблюк М.І. Цивільна оборона: Підручник / М.І. Стеблюк – К.: Знання, 2006. — 29 с.
23. Козлов К.А. Прогностические оценки опасностей субъектов (регионов) / К.А. Козлов, 2001г. - №6. – с. 41-47.

ДОДАТОК А

А1. Лістинг модуля lympho-catcher.cpp

Mat hsvImage;

Mat origImage;

Mat srcImage;

Cell\* cell = 0;

vector<Cell\*> lymphocyts;

int threshArea = 328;

void findingContours(const Mat& src) {

Mat thresholdOutput;

vector<vector<Point> > contours;

vector<Vec4i> hierarchy;

int circleRadius = 30;

/// Detect edges using Threshold

threshold(src, thresholdOutput, thresh, maxThresh, THRESH\_BINARY);

/// Find contours

findContours(src, contours, hierarchy, CV\_RETR\_EXTERNAL, CV\_CHAIN\_APPROX\_SIMPLE, Point(0, 0));

/// Approximate contours to polygons + get bounding rects and circles

vector<vector<Point> > contoursPoly(contours.size());

vector<Point2f>center(contours.size());

vector<float>radius(contours.size());

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

{

approxPolyDP(Mat(contours[i]), contoursPoly[i], 3, true);

minEnclosingCircle((Mat)contoursPoly[i], center[i], radius[i]);

}

Mat drawing = Mat::zeros(thresholdOutput.size(), CV\_8UC3);

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

{

double area = fabs(contourArea(contours[i]));

double perim = arcLength(contours[i], true);

if (area > threshArea) {

Point coordinateOfContour = center[i];

cell = new Lymphocytes(coordinateOfContour.x, coordinateOfContour.y);

lymphocyts.push\_back(cell);

cout << endl << "area " << area;

cout << endl << "count " << lymphocyts.size();

}

}

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

circle(origImage, Point(lymphocyts[i]->getX(), lymphocyts[i]->getY()), circleRadius, Scalar(0, 0, 0), 2, 8, 0);

cout << endl << "contour with coordinates: x = " << lymphocyts[i]->getX() << " y = " << lymphocyts[i]->getY();

}

namedWindow("Cells", CV\_WINDOW\_AUTOSIZE);

imshow("Cells", origImage);

if (!lymphocyts.empty()) {

for (register int i = 0; i < lymphocyts.size() - 1; i++) {

delete lymphocyts[i];

}

lymphocyts.clear();

}

}

void myHSV(const Mat& src) {

Mat tempHsvImage;

cvtColor(src, hsvImage, COLOR\_BGR2HSV);

Mat upperRange;

Mat lowerRange;

inRange(hsvImage, cv::Scalar(122, 140, 100), cv::Scalar(145, 255, 255), upperRange);

//inRange(hsvImage, cv::Scalar(100, 135, 100), cv::Scalar(255, 255, 255), lowerRange);

Mat lympho;

//addWeighted(lowerRange, 1.0, upperRange, 1.0, 0.0, lympho);

//GaussianBlur(upperRange, upperRange, cv::Size(9, 9), 3, 3);

namedWindow("hsv image", cv::WINDOW\_AUTOSIZE);

imshow("hsv image", upperRange);

hsvImage = upperRange;

//waitKey(0);

//imwrite("11.jpg", lower\_red\_hue\_range);

}

void loadFromDir(char\* argv[]) {

string fullPath = "";

string dirPath = argv[1];

struct dirent \*dentry = NULL;

char \*ext = NULL;

DIR \*d = opendir(argv[1]);

while ((dentry = readdir(d)) != NULL)

{

ext = dentry->d\_name + (strlen(dentry->d\_name) - 4);

if (ext > dentry->d\_name)

if ((strncmp(ext, ".png", 4) == 0) ||

(strncmp(ext, ".jpg", 4) == 0)) {

fullPath = dirPath + dentry->d\_name;

srcImage = imread(fullPath);

origImage = srcImage;

namedWindow("src", cv::WINDOW\_AUTOSIZE);

imshow("src", srcImage);

myHSV(srcImage);

findingContours(hsvImage);

cout << endl << dentry->d\_name << endl;

char c = waitKey(0);

if (c == 27) { // если нажата ESC - выходим

break;

}

}

}

}

int main(int argc, char\* argv[]) {

if (argc != 2) {

argv[1] = "./";

}

loadFromDir(argv);

//hsvFinder(1,argv);

return 0;

}

А2. Лістинг модуля HSVFinder.cpp

#include "stdafx.h"

#include "Lymphocytes.h"

using namespace Lympho;

IplImage\* image = 0;

IplImage\* dst = 0;

// для хранения каналов HSV

IplImage\* hsv = 0;

IplImage\* h\_plane = 0;

IplImage\* s\_plane = 0;

IplImage\* v\_plane = 0;

// для хранения каналов HSV после преобразования

IplImage\* h\_range = 0;

IplImage\* s\_range = 0;

IplImage\* v\_range = 0;

// для хранения суммарной картинки

IplImage\* hsv\_and = 0;

int Hmin = 0;

int Hmax = 256;

int Smin = 0;

int Smax = 256;

int Vmin = 0;

int Vmax = 256;

int HSVmax = 256;

Mat src; Mat src\_gray;

int thresh = 100;

int maxThresh = 255;

RNG rng(12345);

//

// функции-обработчики ползунков

//

void myTrackbarHmin(int pos) {

Hmin = pos;

cvInRangeS(h\_plane, cvScalar(Hmin), cvScalar(Hmax), h\_range);

}

void myTrackbarHmax(int pos) {

Hmax = pos;

cvInRangeS(h\_plane, cvScalar(Hmin), cvScalar(Hmax), h\_range);

}

void myTrackbarSmin(int pos) {

Smin = pos;

cvInRangeS(s\_plane, cvScalar(Smin), cvScalar(Smax), s\_range);

}

void myTrackbarSmax(int pos) {

Smax = pos;

cvInRangeS(s\_plane, cvScalar(Smin), cvScalar(Smax), s\_range);

}

void myTrackbarVmin(int pos) {

Vmin = pos;

cvInRangeS(v\_plane, cvScalar(Vmin), cvScalar(Vmax), v\_range);

}

void myTrackbarVmax(int pos) {

Vmax = pos;

cvInRangeS(v\_plane, cvScalar(Vmin), cvScalar(Vmax), v\_range);

}

void hsvFinder(int argc, char\* argv[]) {

// имя картинки задаётся первым параметром

char\* filename = argc == 2 ? argv[1] : "111.jpg";

// получаем картинку

image = cvLoadImage(filename, 1);

printf("[i] image: %s\n", filename);

assert(image != 0);

// создаём картинки

hsv = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 3);

h\_plane = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 1);

s\_plane = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 1);

v\_plane = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 1);

h\_range = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 1);

s\_range = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 1);

v\_range = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 1);

hsv\_and = cvCreateImage(cvGetSize(image), IPL\_DEPTH\_8U, 1);

// конвертируем в HSV

cvCvtColor(image, hsv, CV\_BGR2HSV);

// разбиваем на отельные каналы

cvSplit(hsv, h\_plane, s\_plane, v\_plane, 0);

//

// определяем минимальное и максимальное значение

// у каналов HSV

double framemin = 0;

double framemax = 0;

cvMinMaxLoc(h\_plane, &framemin, &framemax);

printf("[H] %f x %f\n", framemin, framemax);

Hmin = framemin;

Hmax = framemax;

cvMinMaxLoc(s\_plane, &framemin, &framemax);

printf("[S] %f x %f\n", framemin, framemax);

Smin = framemin;

Smax = framemax;

cvMinMaxLoc(v\_plane, &framemin, &framemax);

printf("[V] %f x %f\n", framemin, framemax);

Vmin = framemin;

Vmax = framemax;

// окна для отображения картинки

cvNamedWindow("original", CV\_WINDOW\_AUTOSIZE);

cvNamedWindow("H", CV\_WINDOW\_AUTOSIZE);

cvNamedWindow("S", CV\_WINDOW\_AUTOSIZE);

cvNamedWindow("V", CV\_WINDOW\_AUTOSIZE);

cvNamedWindow("H range", CV\_WINDOW\_AUTOSIZE);

cvNamedWindow("S range", CV\_WINDOW\_AUTOSIZE);

cvNamedWindow("V range", CV\_WINDOW\_AUTOSIZE);

cvNamedWindow("hsv and", CV\_WINDOW\_AUTOSIZE);

cvCreateTrackbar("Hmin", "H range", &Hmin, HSVmax, myTrackbarHmin);

cvCreateTrackbar("Hmax", "H range", &Hmax, HSVmax, myTrackbarHmax);

cvCreateTrackbar("Smin", "S range", &Smin, HSVmax, myTrackbarSmin);

cvCreateTrackbar("Smax", "S range", &Smax, HSVmax, myTrackbarSmax);

cvCreateTrackbar("Vmin", "V range", &Vmin, HSVmax, myTrackbarVmin);

cvCreateTrackbar("Vmax", "V range", &Vmax, HSVmax, myTrackbarVmax);

//

// разместим окна по рабочему столу

//

if (image->width < 1920 / 4 && image->height < 1080 / 2) {

cvMoveWindow("original", 0, 0);

cvMoveWindow("H", image->width + 10, 0);

cvMoveWindow("S", (image->width + 10) \* 2, 0);

cvMoveWindow("V", (image->width + 10) \* 3, 0);

cvMoveWindow("hsv and", 0, image->height + 30);

cvMoveWindow("H range", image->width + 10, image->height + 30);

cvMoveWindow("S range", (image->width + 10) \* 2, image->height + 30);

cvMoveWindow("V range", (image->width + 10) \* 3, image->height + 30);

}

while (true) {

// показываем картинку

cvShowImage("original", image);

cvShowImage("H", h\_plane);

cvShowImage("S", s\_plane);

cvShowImage("V", v\_plane);

cvShowImage("H range", h\_range);

cvShowImage("S range", s\_range);

cvShowImage("V range", v\_range);

// складываем

cvAnd(h\_range, s\_range, hsv\_and);

cvAnd(hsv\_and, v\_range, hsv\_and);

cvShowImage("hsv and", hsv\_and);

char c = cvWaitKey(33);

if (c == 27) { // если нажата ESC - выходим

break;

}

}

printf("\n[i] Results:\n");

printf("[H] %d x %d\n", Hmin, Hmax);

printf("[S] %d x %d\n", Smin, Smax);

printf("[V] %d x %d\n", Vmin, Vmax);

//cvSaveImage("6.jpg", hsv\_and);

// освобождаем ресурсы

cvReleaseImage(&image);

cvReleaseImage(&hsv);

cvReleaseImage(&h\_plane);

cvReleaseImage(&s\_plane);

cvReleaseImage(&v\_plane);

cvReleaseImage(&h\_range);

cvReleaseImage(&s\_range);

cvReleaseImage(&v\_range);

cvReleaseImage(&hsv\_and);

// удаляем окна

cvDestroyAllWindows();

}