

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавр

(ступінь вищої освіти)

на тему РОЗРОБКА ПРОГРАМИ ВІДЕОСПОСТЕРЕЖЕННЯ ЗА
ДОПОМОГОЮ ВЕБКАМЕРИ. SOFTWARE DEVELOPMENT FOR VIDEO
SURVEILLANCE USING A WEBCAM.

Виконав: студент(ка) 4 курсу, групи КНТ-218сп
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Комп'ютерні науки

Домашенко Є.І.

(прізвище та ініціали)

Керівник Сердюк С.М.

(прізвище та ініціали)

Рецензент Неласа Г.В.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ
 Кафедра програмних засобів
 Ступінь вищої освіти бакалавр
 Спеціальність 122 Комп'ютерні науки
(код і найменування)
 Освітня програма (спеціалізація) Комп'ютерні науки
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
 “ ” 2021 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Домашенко Єгора Ігоровича
(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка програми відеоспостереження за допомогою вебкамери. Software Development for Video Surveillance Using a Webcam.

керівник проєкту (роботи) доц., к.т.н. Сердюк Сергій Микитович,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “30” березня 2021 року
 № 103

2. Строк подання студентом проєкту (роботи) травня 2021 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Характеристика об'єкта проєктування. 2. Розробка структури програмного забезпечення. 3. Основні рішення щодо реалізації компонентів системи. 4. Керівництво програміста. 5. Керівництво оператора. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	Сердюк С.М., доцент		
Нормоконтроль	Дейнега Л.Ю., ст. викладач		

7. Дата видачі завдання 28 лютого 2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	2–3 тижні	Розділ 1
3	Розробка архітектури програми.	4 тиждень	Розділ 2
4	Розробка програми.	5–6 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми.	7 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8–9 тижні	Розділ 5, Додатки
7	Захист роботи.	10 тиждень	

Студент(ка)

(підпис)

Домашенко Є.І.

(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис)

Сердюк С.М.

(прізвище та ініціали)

(підпис)

(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:
106 с., 10 табл., 36 рис., 2 дод., 22 джерела.

ВЕБКАМЕРА, ACCORD.NET, AForge.NET, C#, ВІДЕО,
ЗАХОПЛЕННЯ ЗОБРАЖЕНЬ, WINDOWS FORMS, VISUAL STUDIO, AVI,
ДЕТЕКТУВАННЯ РУХІВ, ДЕТЕКТУВАННЯ ОБЛИЧ, INI-ФАЙЛ.

Об'єкт дослідження – процес відеозапису за допомогою вебкамери.

Предмет дослідження – методи відеозапису та відеоспостереження за допомогою вебкамери.

Мета роботи – розробка програми, що дозволяє записувати відео з вебкамери та здійснювати відеоспостереження.

Матеріали, методи та технічні засоби: об'єктно-орієнтоване програмування; мова програмування C# та середовище розробки Visual Studio 2017; програмна платформа .NET; фреймворк Accord.NET; персональний комп'ютер (ПК) з 64-розрядним процесором із тактовою частотою 1 ГГц або швидший, під управлінням операційної системи (ОС) Windows 7 або вище.

Результати. Розроблено застосунок, який забезпечує запис відео з вебкамери та здійснює відеоспостереження, захоплює зображення з вебкамери та може детектувати обличчя та рух об'єктів.

Висновки. Розроблено утиліту, яка автоматизує процес відеоспостереження.

Галузь використання – системи відеоспостереження.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 106 pages, 36 figures, 10 tables, 2 appendixes, 22 sources.

WEBCAM, ACCORD.NET, AForge.NET, C#, VIDEO, IMAGE CAPTURE, WINDOWS FORMS, VISUAL STUDIO, AVI, MOTION DETECTION, FACE DETECTION, INI FILE.

Object of study is video recording process using a webcam.

Subject of study are methods of video recording and video surveillance using a webcam.

The purpose of the work is development of a program that allows you to record video from a webcam and carry out video surveillance.

Materials, methods and technical means: object-oriented programming; Windows executive part; C# programming language and development tools Microsoft Visual Studio 2017; .NET software platform; framework Accord.NET; personal computer (PC) with a 64-bit processor with a clock speed of 1 GHz, running the operating system (OS) Windows 7 or higher.

Results. An application has been developed that provides video recording from a webcam and performs video surveillance, captures an image from a webcam and can detect faces and object movements.

Conclusions. A utility that automates the process of video surveillance has been developed.

The field of use are video surveillance systems.

ЗМІСТ

	С.
Перелік скорочень та умовних познач 8	8
Вступ 9	9
1 Характеристика об'єкта проєктування 11	11
1.1 Огляд технологій розпізнавання облич 11	11
1.2 Аналіз існуючих аналогів 18	18
1.3 Розробка технічного завдання 25	25
1.3.1 Підстава для розробки 25	25
1.3.2 Мета розробки та її призначення 25	25
1.3.3 Основні вимоги до програмного продукту 25	25
1.3.4 Вимоги до видів забезпечення і компонентів 26	26
1.3.5 Узагальнений алгоритм функціонування програми 27	27
1.3.6 Обмеження на установку і виконання програми 28	28
1.3.7 Вимоги до програмної документації 28	28
1.3.8 Вимоги до маркування та упаковки 29	29
2 Розробка структури програмного забезпечення 29	29
2.1 Визначення компонентів системи 29	29
2.2 Обґрунтування вибору мови програмування та засобів розробки 32	32
3 Основні рішення щодо реалізації компонентів системи 36	36
3.1 Розробка алгоритму роботи програми 36	36
3.2 Розробка класів застосунку 39	39
3.3 Розробка графічного інтерфейсу користувача 53	53
4 Керівництво програміста 57	57
4.1 Призначення і умови застосування програми 56	56
4.2 Характеристики програми 57	57
4.3 Початкові та вихідні дані 57	57
4.4 Повідомлення програмісту 58	58
5 Керівництво оператора 59	59
5.1 Призначення програми 59	59

5.2 Умови використання програми.....59

5.3 Виконання програми 59

5.4 Повідомлення оператору 70

Висновки.....73

Перелік джерел посилання 74

Додаток А Текст програми 77

Додаток Б Копії слайдів презентації.....98

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

BPP	– Bits Per Pixel;
FPS	– Frames Per Second;
БД	– база даних;
ГГц	– гігагерц;
Гб	– гігабайт;
МГц	– мегагерц;
ОС	– операційна система;
ПЗ	– програмне забезпечення;
ТЗ	– технічне завдання.

ВСТУП

Відеоспостереження набуває все більшого значення, оскільки організації і приватні особи прагнуть захистити свої фізичні і капітальні активи. У той же час це може привести до необхідності спостереження за великою кількістю людей, місць і інших об'єктів з метою отримання корисної інформації з відеоданих.

Камери відеоспостереження доступні в широкому діапазоні стилів і функцій і є, фактично, обов'язковим компонентом системи безпеки. В Internet-системах безпеки (або IP-камерах) мережу Internet використовують для відправки та отримання даних. Такі системи досить легко встановити і підключити, і користувач може в будь-який час переглядати прямі трансляції з камери за допомогою безкоштовних мобільних додатків для смартфонів і планшетів [1].

Відеокамери можуть бути встановлені практично в будь-якому місці і можуть контролювати ситуацію як усередині, так і зовні периметра об'єкта спостереження. Камери також можуть використовуватися для оповіщення співробітників служби безпеки про підозрілі дії або особи.

Відеоспостереження включає в себе спостереження за сценою або сценами і пошук конкретних дій, які є неналежними або можуть вказувати на появу або наявність неналежної поведінки об'єктів.

Процес відеоспостереження також включає в себе визначення проблемних областей і визначення конкретних відеокамер або груп камер, які можуть переглядати ці області. Якщо існує можливість ідентифікувати тимчасові проміжки, коли виникають тенденції до порушення безпеки, це також може бути корисним для даного процесу. Потім, переглядаючи вибрані зображення, можна визначити, чи не відбувається будь-яка непотрібна активність в певний час. Саме у сфері відеоспостереження найчастіше використовуються функції детектування рухів та облич, запису відео до файлів, ведення журналів подій.

Також варто відмітити, що крім виконання завдань моніторингу, системи відеоспостереження успішно можна застосувати в інших областях. На ці системи можна покласти функції управління промисловими процесами, забезпечити безпеку праці на виробництві, здійснювати обслуговування клієнтів – багатофункціональність системи дозволяє знайти їй застосування в багатьох побутових і виробничих областях.

Метою даної дипломної кваліфікаційної роботи бакалавра є розробка програми, що дозволяє записувати відео з вебкамери у файли, здійснювати відеоспостереження, захоплювати зображення з вебкамери, детектувати обличчя та рух.

1 ХАРАКТЕРИСТИКА ОБ'ЄКТА ПРОЄКТУВАННЯ

1.1 Огляд технологій розпізнавання облич

Розпізнавання облич, як одне з найбільш успішних застосувань програм аналізу зображень, останнім часом привернуло велику увагу. Це пов'язано з наявністю реальних технологій, в тому числі мобільних рішень. Дослідження в області автоматичного розпізнавання облич ведуться з 1960-х років, але проблема все ще залишається невирішеною. Впродовж останнього десятиріччя спостерігається значний прогрес в цій області завдяки досягненням в області моделювання і аналізу облич. Хоча системи були розроблені для виявлення та відстеження облич, надійне розпізнавання облич і в теперішній час являє собою серйозну проблему для дослідників комп'ютерного зору і розпізнавання образів. Існує кілька причин зростання останнім часом інтересу до розпізнавання облич, в тому числі зростаюча стурбованість громадськості питаннями безпеки, необхідність перевірки особи в цифровому світі, методи аналізу облич і моделювання в управлінні мультимедійними даними і комп'ютерних розвагах.

Останні досягнення в області автоматизованого аналізу облич, розпізнавання образів і машинного навчання дозволили розробити системи автоматичного розпізнавання облич для вирішення цих завдань. З одного боку, розпізнавання обличчя - природний процес, тому що люди зазвичай роблять це без особливих зусиль і без особливого усвідомлення. З іншого боку, застосування цього процесу в області комп'ютерного зору залишається складним завданням. Будучи частиною біометричної технології, автоматичне розпізнавання облич має безліч бажаних властивостей. В їх основі лежить важлива перевага – неінвазивність (відсутність проникнення через природні зовнішні бар'єри організму).

Різні біометричні методи можна розділити на фізіологічні (відбиток пальця, ДНК, обличчя) і поведінкові (натискання клавіші, голосовий відбиток) категорії. Фізіологічні підходи більш стабільні і незмінні, за винятком важкої травми у особи. Поведінкові моделі більш чутливі до загального стану людини,

такого як стрес, хвороба або втома.

Розглянемо основні методи виявлення облич з використанням ефективних методів статистичного навчання.

Основні елементи типової системи розпізнавання облич наведено на рис.1.1 [1].



Рисунок 1.1 – Основні елементи типової системи розпізнавання облич

Перший етап – це виявлення облич на отриманому зображенні в незалежності від масштабування та розміщення. На цьому етапі часто використовується розподілена процедура фільтрації, щоб визначити місцеположення, які представляють обличчя, і фільтрує їх за допомогою точних класифікаторів. Показово, що всі зміни зміщення, масштабування та повороти повинні оброблятися на етапі виявлення обличчя. Наприклад, згідно [2], вираз обличчя і зміна зачіски або усміхнене та похмуре обличчя все ще мають важливе значення на етапі розпізнавання образів.

На наступному етапі система, що заснована на наборі антропометричних

даних, прогнозує приблизне розташування основних ознак обличчя, таких як очі, ніс і рот. Зазвичай, вся процедура повторюється для прогнозування підфункцій (особливостей) щодо основних ознак та перевіряється зі статистикою розміщення, щоб відхилити будь-які неправильно розміщені ознаки.

Цей етап пов'язаний з етапом нормалізації обличчя. Першим кроком на цьому етапі є визначення положення обличчя на зображенні. Це проводиться шляхом визначення кольору шкіри. Формула перетворення [4] являє собою наступну комбінацію нормалізованого кольору моделі RGB і компонента I2 з колірної моделі I1I2I3 :

$$S1 = (RG)(GI) \text{ and } S2 = \frac{255}{\max(RB) - \min(RB)} (RB - \min(RB)), \quad (1.1)$$

$$\text{де } RB = \left| \frac{R}{R+G+B} - \frac{B}{R+G+B} \right|;$$

$$GI = 0,299R + 0,587G + 0,144B;$$

$S1, S2$ – маска кольору шкіри;

R, G, B – компоненти RGB моделі кольорів.

Другий крок нормалізації обличчя – знаходження позиції очей. Це отримується шляхом обчислення спрямованих змін інтенсивності зображення (градієнта зображення).

Виділені опорні точки генеруються в результаті геометричних комбінацій на зображенні обличчя і створюють так званий вектор (цифровий шаблон) ознак обличчя, а потім починається фактичний процес розпізнавання. Це виконується шляхом знаходження локального представлення вигляду обличчя в кожній з опорних точок (рис. 1.2). Схема представлення залежить від підходу, наприклад це порівняння частин обличчя з частинами обличчя персони, що знаходяться у базі даних (БД). Щоб впоратися з такими складнощами і з'ясувати істинний інваріант розпізнавання, дослідники

розробили різні алгоритми розпізнавання.

Наприклад, компанією Apple був розроблений сканер 3D форми особи, що встановлюється в смартфонах Iphone XR, iPhone X, Iphone XS, XS Max, і новіших моделях.

Камера TrueDepth захоплює дані особи, проектуючи на нього і аналізуючи більше 30 000 невидимих точок. Таким чином пристрій складає детальну структурну карту особи, а також його зображення в інфрачервоному спектрі. Фрагмент нейронного ядра мікропроцесорів A11, A12 Bionic, A12X Bionic і A13 Bionic, захищений модулем Secure Enclave, перетворює карту глибини і інфрачервоне зображення в математичне представлення, яке порівнюється з зареєстрованими даними особи [6].

Технологія Face ID автоматично адаптується до таких змін зовнішності, як макіяж або неголеність. При більш значних змінах (наприклад, гоління бороди) система запропонує ввести пароль, щоб підтвердити особу, і тільки після цього оновить дані про особу. Технологія Face ID розпізнає обличчя навіть при наявності капелюха, шарфа, контактних лінз та сонцезахисних окулярів. Крім того, вона працює в приміщеннях, на вулиці і навіть у повній темряві.

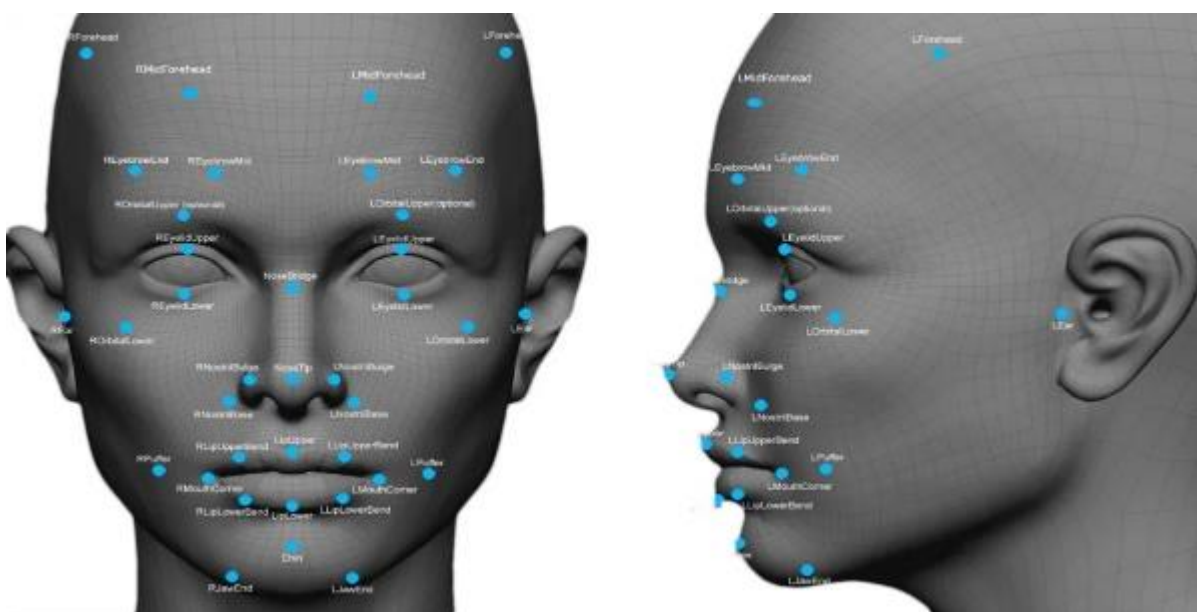


Рисунок 1.2 – Вектор ознак обличчя

Існуюча технологія розпізнавання облич – FERET має кілька обмежень. В [3] був представлений ранній еталонний тест технологій розпізнавання облич. Хоча в ідеальних умовах його продуктивність відмінна, в умовах мінливого освітлення, виразу, роздільної здатності, відстані або старіння продуктивність значно знижується. Справа в тому, що системи розпізнавання облич все ще не дуже стійкі до відхилень від ідеального зображення обличчя. Інша проблема розпізнавання – це ефективний спосіб зберігання і надання доступу до коду обличчя (або шаблону обличчя), що зберігається у вигляді набору рис обличчя, який було отримано з зображення або відео.

Таким чином, огляд представлених вище елементів складного процесу розпізнавання облич, показав, що існує ряд обмежень і недоліків. Вони вимагають уточнення або заміни новими алгоритмами, методами або навіть технологіями.

Найбільш важливі переваги і недоліки методів розпізнавання облич представлені в табл. 1.1.

Таблиця 1.1 – Порівняння методів розпізнавання облич

Метод	Переваги	Недоліки
1	2	3
1. Класичний алгоритм розпізнавання облич	Цей метод проєціює обличчя на лінійний підпростір, що охоплює зображення свого обличчя. Відстань від поверхні обличчя є перпендикуляр до площини середини зображення, яка може бути легко повернута на відстані Махаланобіса (відстань у евклідовому просторі) з певною ймовірнісною інтерпретацією.	Цей метод може неадекватно розпізнавати обличчя при великих варіаціях освітлення облич, їх виразу та інших факторах. Існує проблема вибору розміру околів або призначення їм оптимальних значень.

Продовження табл.1.1

1	2	3
2. Штучні нейронні мережі	Радіальна базисна функція штучної нейронної мережі природним чином інтегрована з розкладом невід'ємних матриць (non-negative matrix factorization). Також використовуються інші підходи до процесу спрощення щодо вбудованої функції лінеаризації нейронних мереж і прискорення обчислень. Ідеальне рішення, особливо для розпізнавання облич на зображеннях з частковим спотворенням та оклюзією.	Головний недолік цього підходу в тому, що потрібна велика кількість зразків для навчання мережі.
3 Розпізнавання обличчя на основі 3D	Розширює традиційний 2D процес захоплення і має велику точність. Інформація про глибину не залежить від пози і освітлення, що підвищує надійність рішення.	Вимагає, щоб всі елементи системи розпізнавання 3D облич були добре відкалібровані і синхронізовані з існуючими 3D даними. Обчислювально дорогий метод, не підходить для практичних застосувань.

Продовження табл. 1.1

1	2	3
4 Вейвлети Габора	Вейвлети Габора дають хороші результати в завданнях локалізації точкових особливостей на зображеннях. Вейвлети стійкі до змін яскравості і геометричних спотворень. Широко використовуються в різних біометричних застосунках для розпізнавання райдужної оболонки і відбитків пальців.	Недолік методів на основі вейвлетів Габора – велика розмірність простору функцій Габора, так як зображення обличчя згортається за допомогою банку фільтрів Габора. Даний підхід – обчислювально інтенсивний і непрактичний для застосунків реального часу. Крім того, спрощена функція Габора чутлива до швидких змін зображення.
5 Розпізнавання облич на основі відео	Головною перевагою підходу є можливість використання надмірності, яка присутня в відео для поліпшення систем з нерухомим зображенням.	Порівняно мало вивчений. Зростання кількості проблем при вимірюванні подібності двох (або більше) зображень.

Продовження табл.1.1

1	2	3
6 Методи на основі дескриптора обличчя	Основна ідея розробки дескрипторів зображень полягає у вивченні найбільш дискримінантних локальних опорних точок, які мінімізують різницю між зображеннями однієї і тієї ж людини і максимізує відмінності між зображеннями інших людей. Ці методи стійкі до змін освітлення і виразу обличчя. Вони пропонують компактний, такий що легко отримується дескриптор (набір характеристик, що описують особу незалежно від сторонніх чинників - вік, зачіска, макіяж).	Підхід вимагає великих обчислювальних ресурсів на етапі отримання дескриптора, але забезпечує простоту і продуктивність в порівнянні з онлайн-застосунками.

Методи, зазначені в табл. 1.1, ілюструють еволюцію технології розпізнавання облич. Слід підкреслити величезний потенціал методів, заснованих на дескрипторах облич, в зв'язку з тим, що ідея локального дескриптора недавно була визнана найбільш важливою проєктувальною структурою для задач ідентифікації і верифікації облич [1].

1.2 Аналіз існуючих аналогів

Перед розробкою технічного завдання (ТЗ) для програмного продукту, що розробляється необхідно розглянути вже існуючі програмні рішення. У

цьому підрозділі проведено аналіз програмних аналогів, визначено їх основні переваги та недоліки, а також зроблено висновок, які з даних особливостей повинно мати та враховувати програмне забезпечення, що розробляється.

Існує кілька десятків програм для запису відео з вебкамери. До них належать ManyCam, Movavi Screen Capture Studio, Cyberlink YouCam, Active WebCam, VirtualDub, Debut Video Capture, Altarsoft Video Capture, Webcam Surveyor та інші [7].

Було обрано та порівняно три аналоги ПЗ, що розробляється.

Першим розглянемо застосунок Movavi Screen Capture Studio [8] – програмне забезпечення, за допомогою якого можна робити захоплення відео з вебкамери. Ця програма працює з різними пристроями введення. Розмір програми – 42,3 Мб.

До переваг Movavi Screen Capture Studio належать наступні:

- підтримка запису до 60 кадрів у секунду;
- одночасний запис декількох джерел звуку (мікрофон, музика, навушники, голосові повідомлення, системні звуки і так далі);
- підтримка гарячих клавіш;
- захоплення скріншотів під час запису;
- налаштування таймеру, після закінчення якого запис автоматично зупиняється;
- містить вбудований відеоредактор, що дозволяє відредагувати записане відео;
- результат захоплення відразу зберігається в форматі MKV. Це є перевагою програми, так як деякі застосунки захоплення зберігають відео у власному форматі. В цьому випадку дисковий простір витрачається нерационально, так як відео записується спочатку у власний формат, а потім вже перетворюється в один із стандартних форматів;
- для завантаження необхідно небагато місця на жорсткому диску.

Інтерфейс Movavi Screen Capture наведено на рис. 1.3.



Рисунок 1.3 – Інтерфейс Movavi Screen Capture

Відеоредактор програми вміє редагувати як захоплені відео, так і отримані ззовні (якщо відео збережено в підтримуваному форматі). Редактор дозволяє виконувати основні операції, такі як обрізка або поворот відео, застосовувати різні відеофільтри, додавати переходи між роликами, накладати текст, в тому числі титри, а також вставляти в відео графічні стікери.

Меню «Эффекты» дозволяє задати різні ефекти: наприклад, запис натискань клавіш і курсора миші, щоб показати в відеоуроці, які саме дії виконувалися за допомогою миші і які клавіші були натиснуті.

До недоліків Movavi Screen Capture Studio належать [9]:

- у Windows XP відсутня підтримка запису звуку з декількох джерел під час зйомки монітора;
- програма ставить водяний знак на відео у безкоштовній версії;
- для запису відео потрібна відеокарта Nvidia, Intel HD, ATI AMD – з графічними прискорювачами інших брендів інструмент захоплення не працює;
- спроба установки програм «Яндекса» при інсталяції програми.

Інший аналогічний застосунок Active WebCam [10] – це спеціалізоване ПЗ, яке займається тільки відеозаписом без подальшої обробки. Програма захоплює зображення зі швидкістю до 30 кадрів в секунду з будь-якого відеопристрою. Приклад роботи з Active WebCam наведений на рис. 1.4.



Рисунок 1.4 – Приклад роботи з Active WebCam

Трансляцію можна переглядати за допомогою будь-якого Internet-браузера без установки будь-яких плагінів, тому користувачам, які відвідують вебсайт, буде простіше, тому що їм не потрібно буде завантажувати або встановлювати будь-які файли.

Програма включає помічника, який допоможе створювати вебсайт, з якого можна передавати зображення, де буде необхідно лише вибрати колір, назву та шрифт, оскільки застосунок зробить все інше автоматично [11].

Крім того користувач має можливість зробити налаштування, коли у разі виявлення руху в контрольованій зоні, програма подає сигнал тривоги,

відправляє захоплені зображення по електронній пошті, ініціює трансляцію або записує відео.

Переваги Active WebCam:

- робота з платами відеозахоплення, підтримка будь-якої вебкамери;
- дозволяє здійснювати одночасний запис одразу з необмеженої кількості камер, транслювати захоплений контент у мережу Інтернет або завантажувати відео на віддалений FTP-сервер;
- всі параметри запису і перегляду можна детально налаштувати;
- зображення швидко стискується після встановлення необхідного кодека;
- можна додавати паролі для мовлення та запису;
- програма має функції для розміщення накладень зображень і відміток дати/часу на кожному кадрі;
- можна вказати шлях, час запису кожного файлу у автоматичному режимі.

Недоліки Active WebCam:

- складний деталізований інтерфейс;
- не передбачено ніяких інструментів для обробки зображень, накладення ефектів і т.п.;
- значні системні вимоги.

Третій аналог, що розглядається – це програма Webcam Surveyor [12], тобто програмне забезпечення, яке записує матеріал (з вебкамери або з екрана) і зберігає його в відеофайл. Для захоплення зображення з вебкамери у цьому застосунку є декілька спеціальних опцій, таких як детектор руху і відкладений старт. Інтерфейс програми Webcam Surveyor наведено на рис. 1.5.

Застосунок Webcam Surveyor також підтримує сучасні алгоритми стиснення відео, такі як DivX, Mpeg4, Mjpeg і ін. За допомогою інтегрованого менеджера файлів спрощується структуризація та зберігання отриманої відеоінформації.

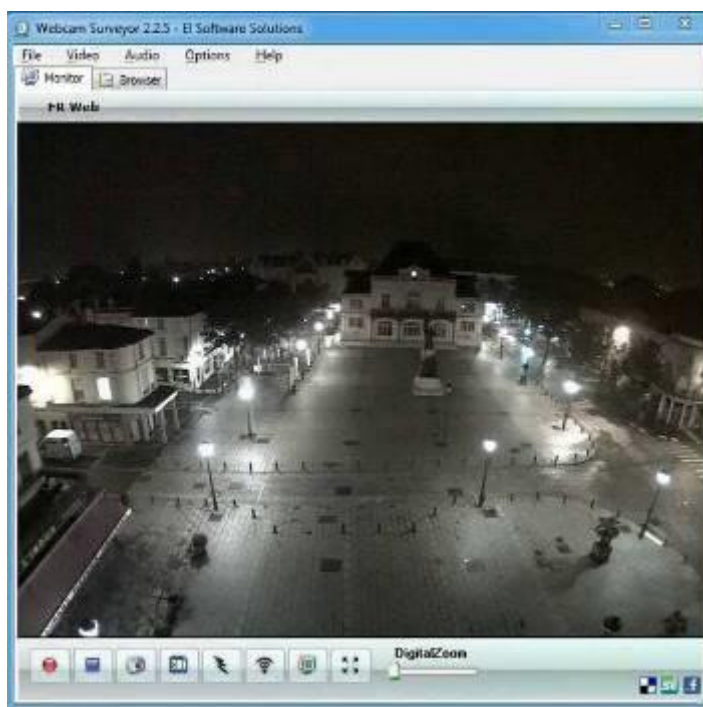


Рисунок 1.5 – Інтерфейс Webcam Surveyor

Переваги Webcam Surveyor:

- відносно потужна програма, яка має невеликий розмір (потребує 5 Мб вільного дискового простору та 20 Мб оперативної пам'яті) і не перевантажує систему;
- дозволяє транслювати потокове відео по локальній мережі та мережі Internet. За допомогою цієї функції користувач може віддалено спостерігати за своїм будинком або офісом, виробничими процесами. Для перегляду медіа потоків, можна використовувати відомий Windows Media Player, VCL-player або інші плеєри;
- дозволяє зняти відео з зовнішнього пристрою та робочого столу;
- оповіщення через електронну пошту, тобто якщо виявлено рух на спостерігаємому об'єкті, то програма надішле повідомлення з прикріпленими знімками;
- можна запустити запис з відкладеним стартом або з детектуванням руху;
- має прихований режим, тобто дає можливість сховати програму на комп'ютері користувача і керувати нею за допомогою комбінацій гарячих

клавіш. Це дозволить зробити відеозапис у присутності небажаних осіб або дізнатися, що відбувається з комп'ютером під час відсутності користувача;

— має простий, зручний інтерфейс.

Недоліком застосунку Webcam Surveyor є невелика кількість інструментів відносно його вартості.

Результати порівняння існуючих аналогів наведені у табл.1.2.

Таблиця 1.2 – Порівняння існуючих аналогів ПЗ, що розробляється

№	Порівняльні характеристики	Назва програми		
		Movavi Screen Capture Studio	Active WebCam	Webcam Surveyor
1	Запис відео з вебкамери	+	+	+
2	Захоплення зображень	+	+	+
3	Детектування руху	-	+	+
4	Запис на диск	+	-	-
5	Відкладений запис	-	+	+
6	Ліцензія	Пробна	Пробна	Пробна
7	Вартість, \$	35	29	30

Таким чином найбільш розвинений, та близький до програми, що розроблюється функціонал мають програми Active WebCam та Webcam Surveyor.

Розгляд існуючих аналогів Movavi Screen Capture, Active WebCam, Webcam Surveyor, Active WebCam показав, що основними недоліками програм для запису з вебкамери є висока вартість програмного забезпечення, лише англomовний інтерфейс, відсутність відкладеного запису.

Враховуючи недоліки і переваги аналогів, програма, що розроблюється, повинна мати такі переваги: україномовний інтерфейс, не навантажувати

систему, зберігати відео у оптимальному розмірі файлу, надавати можливість одночасного підключення відразу декількох пристроїв, мати можливість відкладеного запису, бути безкоштовною.

Проведений аналіз програмних аналогів послужив основою для розробки технічного завдання для програми відеоспостереження за допомогою вебкамери.

1.3 Розробка технічного завдання

1.3.1 Підстава для розробки

Підставою для розробки служить завдання на дипломну кваліфікаційну роботу бакалавра на тему "Розробка програми відеоспостереження за допомогою вебкамери", затверджене наказом № 103 від 30 березня 2021 р. по НУ «Запорізька політехніка».

1.3.2 Мета розробки та її призначення

Метою розробки є створення програмного продукту, що виконує функції запису відео з вебкамери. Програма призначена для відеофіксації подій у зоні дії камери та запису відеоінформації на диск. Програмне забезпечення може бути використано у випадках, коли необхідно автоматизовано відстежувати перебіг подій з мінімальним використанням людських ресурсів.

1.3.3 Основні вимоги до програмного продукту

Застосунок, що розробляється має забезпечувати виконання наступних функцій:

- записувати відео з веб-камери;
- робити фотографії (захоплювати кадри з вебкамери);
- детектувати обличчя, які знаходяться у кадрі;

- детектувати рухи на відео, які знаходяться у кадрі, у разі детектування виділяти зону рухів кольором;
- підтримувати одночасний запис з декількох вебкамер;
- виконувати запис відео згідно з заданим користувачем розкладом або у режимі безперервної зйомки;
- записувати відео в файл з заданим або каталогом за замовчуванням;
- назва файлу за замовчуванням, а також дата початку – дата кінця;
- надавати можливість задання максимального розміру файлу, у який здійснюється запис відео;
- мати історію виявлення рухів, яка буде зберігатися у бінарному файлі.

Вхідними даними є налаштування програми: необхідна вебкамера, роздільна здатність відео, максимальний розмір файлу, шлях до каталогу, де має зберігатися файл, час зйомки.

Вихідними даними є відео-файли запису з вебкамери у форматі .avi, фотографії у форматі .png, бінарний файл з історією детектування рухів.

1.3.4 Вимоги до видів забезпечення і компонентів

Програма повинна працювати під управлінням операційної системи Windows 7 або вище. Тому вимоги до апаратно-технічних засобів обумовлені вимогами до ОС.

До складу технічних засобів повинен входити персональний комп'ютер (ноутбук), який задовольняє наступним мінімальним вимогам:

- процесор або система на кристалі (SoC) з частотою 1 ГГц чи вищою;
- оперативна пам'ять 1 Гб (для 32-розрядної версії) або 2 Гб (для 64-розрядної версії);
- простір на жорсткому диску 16 Гб (для 32-розрядної версії) або 32 Гб (для 64-розрядної версії);

- відеокарта DirectX 9 або новішої версії з драйвером WDDM 1.0;
- дисплей мінімум 800x600 пікселів;
- вбудована веб-камера або/та USB веб-камера;
- комп'ютерна миша.

Вихідні коди програми повинні бути реалізовані мовою C#. У якості інтегрованого середовища розробки програми повинно бути використано середовище Visual Studio 2017.

Для реалізації необхідного функціоналу має застосовуватися фреймворк Accord.NET зі складовим Aforge.NET з необхідними доповненнями до нього, а також використовуватися інтерфейс API Windows Forms, оскільки вони містять усі необхідні складові для реалізації функціоналу і зручного інтерфейсу програми.

1.3.5 Узагальнений алгоритм функціонування програми

Програма повинна забезпечувати виконання функцій у відповідності з алгоритмом, наведеним нижче.

Крок 1. Обрання потрібної вебкамери.

Крок 2. Обрання потрібної роздільної здатності.

Крок 3. Задання частоти кадрів.

Крок 4. Обрання шляху, куди буде записуватися відеофайл, задання його назви, за необхідністю створення чи вибір каталогу за замовчуванням.

Крок 5. У додаткових налаштуваннях задання максимального розміру файлу, обрання розкладу запису.

Крок 6. Початок захоплення відео з екрану, детектування облич і рухів у кадри.

Крок 7. За необхідністю додання вебкамери за кроками 1-6.

Крок 8. При натисканні на відповідну кнопку зупинення запису.

Крок 9. При натисканні на відповідну кнопку, обрання вебкамери (якщо їх декілька), захоплення зображення і збереження фотографії.

1.3.6 Обмеження на установку і виконання програми

Обмеження на встановлення і використання програми обумовлені вимогами до апаратно-технічних засобів, описаними у п. 1.3.4.

1.3.7 Вимоги до програмної документації

Програмна документація повинна включати: керівництво програміста, керівництво оператора та вихідний код програми.

1.3.8 Умови експлуатації

Програма повинна функціонувати під управлінням ОС Windows 7 або вище.

2 РОЗРОБКА СТРУКТУРИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Визначення компонентів системи

Вихідними вимогами для розробки структури ПЗ є вимоги до функціональності застосунку, наведені у п. 1.3.3. технічного завдання.

В результаті аналізу даних вимог, запропоновані наступні способи їх вирішення:

- для забезпечення введення і зберігання даних про веб-камеру (роздільну здатність, максимальний розмір файлу, час початку і кінця запису за розкладом, кількість FPS, кількість BPP) необхідно розробити клас з полями-даними веб-камери;
- для забезпечення можливості початку і завершення запису у обраний користувачем файл необхідно застосувати клас `VideofileWriter` бібліотеки `Accord.Video.FFMPEG`;
- для забезпечення можливості захоплення зображення необхідно застосувати клас `VideoCaptureDevice` бібліотеки `Accord.Video.DirectShow`;
- для забезпечення можливості запису за розкладом необхідно використати таймер та збереження налаштувань;
- для забезпечення можливості запису в файл за замовчуванням, у випадку перевищення максимального розміру файлу необхідно робити перевірку за таймером;
- для забезпечення можливості детектування руху, виділення червоним маркером зони або границь руху у випадку детектування необхідно використати бібліотеки `Accord.Vision.Motion`;
- для забезпечення можливості детектування облич, їх виділення жовтим прямокутником у випадку детектування необхідно використати бібліотеку `Accord.Vision.Detection`;
- для забезпечення можливості збереження налаштувань у INI-файл необхідно створити клас для читання і записування у файли `.ini`;

- для забезпечення можливості збереження історії роботи з кожною вебкамерою у окремий лог-файл необхідно створити функції запису тексту у лог-файл;
- для забезпечення можливості відображення часу від початку запису необхідно використати таймер з секундним інтервалом та відповідні елементи керування;
- для забезпечення можливості відображення останньої дії у статусі та окремо результатів успішних детектувань необхідно використати елементи керування.

Система, що розробляється складається з одного Desktop-застосунку. Можна виділити його окремі частини – компоненти. Даний застосунок складається з системних (фреймворку Windows Forms) та власноруч встановлених (фреймворку Accord.NET) пакетів та класів у вигляді файлів з розширенням .cs. Класи будуть розглянуті у розділі 3.

Використані в програмі пакети Accord.NET та зв'язки між ними зображені на рис. 2.1.

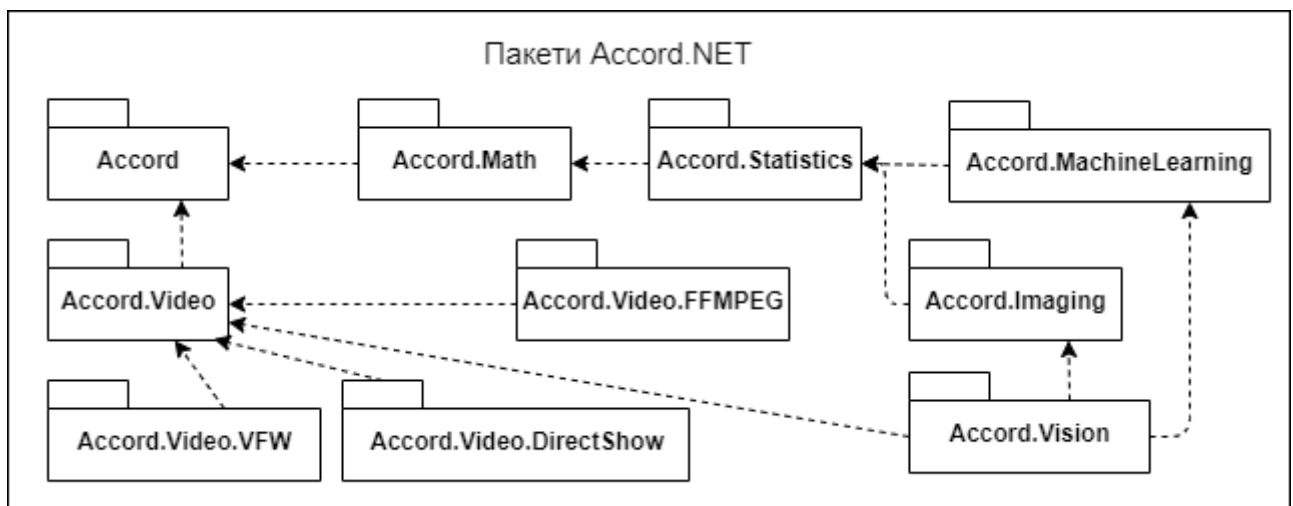


Рисунок 2.1 – Використані в програмі пакети Accord.NET

Розглянемо зв'язки між пакетами. На рис. 2.1 зображено стрілками з пунктирною лінією наступний зв'язок: від пакету, що використовує, до пакету, який є «батьківським»; кожен наступний пакет має зв'язок з пакетами

попереднього. Наприклад, пакет Accord.Imaging використовує пакети Accord, Accord Math та Accord.Statistics.

Необхідно також розглянути призначення бібліотек Accord.NET, які застосовано для реалізації функціоналу програмного забезпечення, що розроблюється.

Accord.NET містить набір бібліотек, доступних у вихідному коді на мові C#, а також за допомогою виконуваних інсталяторів і пакетів NuGet. Бібліотеки використовуються розробниками і дослідниками при вирішенні завдань із галузі комп'ютерного зору та комп'ютерної обробки звуку. До них належать наступні бібліотеки [13].

Пакет Accord.Controls.Vision – містить компоненти Windows Forms і елементи управління для відстеження рухів голови, обличчя, рук і інших завдань, пов'язаних з комп'ютерним зором.

Пакет Accord.Controls.Imaging – включає елементи управління Windows Forms для відображення і обробки зображень. Містить регулятор кольору Color Sliders, засоби вибору відтінку Hue Pickers, PictureBox з підтримкою розширеного формату, відеопрогравачі і зручний елемент управління ImageBox, який імітує традиційну поведінку MessageBox для швидкого відображення або перевірки зображень.

Пакет Accord.Imaging – містить детектори точок зацікавлення для класичних алгоритмів (наприклад, SIFT, SURF, ORB, FAST, BRISK,), фільтри зображень, методи зіставлення зображень і стикування зображень, а також екстрактори функцій, такі як гістограми орієнтованих градієнтів і описові функції Haralick.

Пакет Accord.Video – містить інтерфейси і класи для доступу до різних джерел відео, таких як IP-відеокамери (потокі MJPEG). Цей пакет створений на основі AForge.NET Framework.

Пакет Accord.Video.DirectShow – містить класи та методи обробки джерел відео за допомогою інтерфейсу DirectShow (такими, як веб-камери USB,

відеофайли, пристрої захоплення екрану). Цей пакет походить від AForge.NET Framework.

Пакет Accord.Video.FFMPEG – містить класи та методи обробки джерел відео з використанням комплексу вільних комп'ютерних програм та програмних бібліотек FFMPEG. З цієї причини цей пакет доступний лише за ліцензією GPL. Цей пакет походить від AForge.NET Framework

Пакет Accord.Video.VFW – містить класи читання та запису відеофайлів AVI за допомогою Video For Windows. Цей пакет походить від AForge.NET Framework.

Пакет Accord.Vision.Detection – містить детектори об'єктів, такі як метод Віоли-Джонса (ознаки Хаара). Ознаки Хаара повністю сумісні з визначеннями, що генеруються бібліотекою комп'ютерного зору OpenCV, а збірка поставляється з прямою підтримкою пов'язаних шаблонів обличчя і людського носа.

Пакет Accord.Vision – виявлення і відстеження особи в режимі реального часу, а також загальні методи для виявлення, відстеження і перетворення об'єктів в потоки зображень. Містить визначення каскаду Хаара, трекери Camshift і Dynamic Template Matching. Включає попередньо створені класифікатори для людських облич і деякі рис обличчя, наприклад, таких як носи.

Пакет Accord.Vision.Motion – включає алгоритми обробки руху, який підраховує окремі рухомі об'єкти та виділяє їх.

2.2 Обґрунтування вибору мови програмування та засобів розробки

Згідно п. 1.3.4. ТЗ у якості мови програмування та засобу розробки було визначено мову C# та інтегроване середовище розробки Visual Studio 2017 відповідно. Програма повинна працювати під управлінням операційної системи Windows 7 або вище. Обґрунтуємо цей вибір.

Під час вибору операційної системи, необхідно звернути увагу на статистику. За рейтингом 2020 року операційних систем в Україні, які встановлені на комп'ютерах, лідирує ОС Windows – 83.73% всіх комп'ютерів, OS X – 13.1% та Linux – 1.82% [14]. Виходячи з цього і було обрано ОС Windows під керуванням якої і буде виконуватися програмне забезпечення, що розробляється.

Одною з головних причин вибору мови C# є величезна кількість наборів інструментів і фреймворків, що підтримуються Microsoft, які отримує програміст для підтримки мови. Практично будь-який розробник визнає, що Visual Studio є однією з самих, якщо не самою багатофункціональним і потужним середовищем розробки на ринку. Платформа .NET надає сотні бібліотек для створення веб-сайтів, забезпечення безпеки, роботи з файловою системою і т. д.

Мова C# – об'єктно-орієнтована мова програмування з автоматичним керуванням пам'яттю, яка призначена для програмування на платформі .NET. Вона активно підтримується Microsoft, при цьому нові функції і синтаксичні поліпшення впроваджуються набагато швидше, ніж інші мови, такі як Java.

Синтаксис C# близький до C++ і Java. Мова має строгу статичну типізацію, підтримує поліморфізм, перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, винятки, коментарі у форматі XML [15].

Перевагами використання C# є наступні.

Згідно з даними 2021 року [16] C# знаходиться на першому місці серед мов програмування для написання десктопних застосунків (рис. 2.2). Ця властивість важлива для розробників, оскільки популярність мови прямо пропорційна тому, наскільки для неї будуть доступні онлайн-матеріали, що заощаджує час при вирішенні різних завдань в розробці.

Гнучкість та універсальність цієї мови програмування також є перевагами: до застосунків, які можуть бути розроблені за допомогою C#, .Net і Visual Studio належать застосунки для Windows, мобільні та вебзастосунки,

ігри, застосунки для Android і iOS, які розробляються за допомогою додаткових фреймворків (наприклад, Xamarin або Mono) та інші [17].

Мови програмування з розбивкою за сферами використання

Mobile Back-end Front-end Data processing Desktop System QA automation Full Stack IoT GameDev

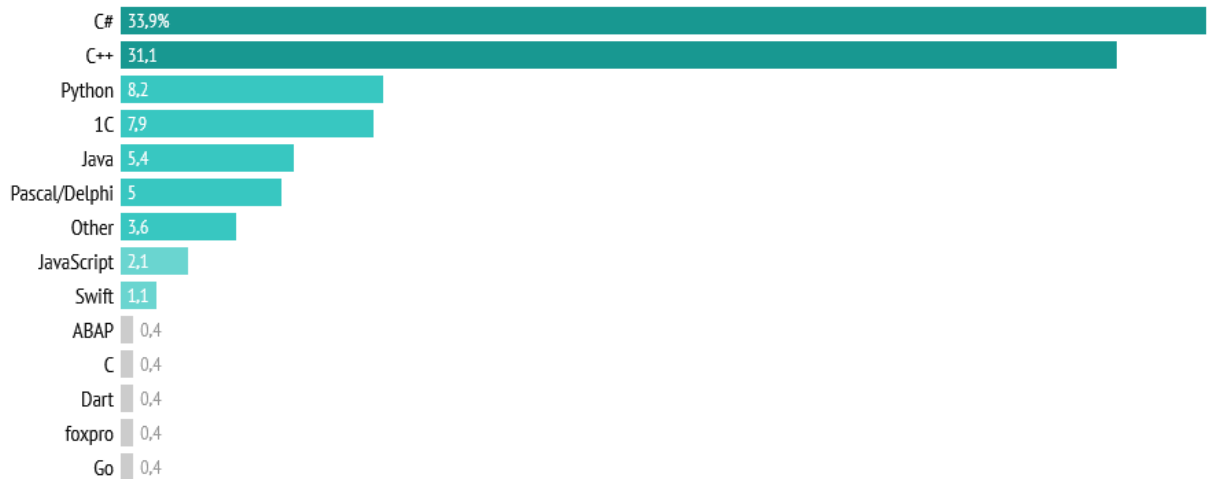


Рисунок 2.2 – Рейтинг мов програмування

Основні особливості C# наступні [18]:

- можливість визначення класів, інтерфейсів, структур, переліків та делегатів;
- підтримка всіх трьох об'єктно-орієнтованих функцій: інкапсуляції, успадкування та поліморфізму;
- тип безпеки. Об'єкти та масиви автоматично ініціалізуються нулем.

Масиви перевіряються на відсутність помилок;

- не потрібно вказівників;
- перевантаження оператора для нестандартних типів;
- автоматичне управління пам'яттю через збір сміття.

Середовище програмування Microsoft Visual Studio було обрано по наступним причинам.

Microsoft Visual Studio - це інтегроване середовище розробки від компанії Microsoft. Воно використовується для розробки комп'ютерних програм, а також вебсайтів, вебпрограм, вебсервісів та мобільних застосунків.

Visual Studio використовує платформи розробки програмного забезпечення Microsoft, такі як Windows API, Windows Forms, Windows Presentation Foundation, Windows Store та Microsoft Silverlight. Воно може створювати як власний код, так і керований код.

Функціональна структура середовища включає [19]:

- редактор вихідного коду, який включає множину додаткових функцій, таких як автодоповнення IntelliSense, рефакторінг коду та ін.;
- відлагоджувач коду;
- редактор форм, призначений для спрощеного конструювання графічних інтерфейсів;
- вебредактор;
- дизайнер класів;
- дизайнер схем баз даних.

Переваги використання Visual Studio є наступними:

- Visual Studio є рекомендованим середовищем виробника ОС Windows;
- наявність великої кількості шаблонів і бібліотек дозволяють зменшити час та зусилля на написанням коду та полегшити його розуміння;
- має єдину інфраструктуру налагодження програми;
- зручне і зрозуміле створення інтерфейсу програми та достатня кількість елементів, налаштування властивостей яких відбувається за допомогою коду або у контекстних меню;
- має зручні редактори різних типів файлів;
- наявність рефакторінгу коду та підказок для вводу імен змінних, класів і функцій.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Розробка алгоритму роботи програми

Визначення послідовності дій користувача має велике значення під час проектування графічного інтерфейсу програмного забезпечення. Це дає змогу визначити моменти інтересу користувача до тієї чи іншої активності та наділити його лише потрібною функціональністю у даний момент часу, не перевантажуючи інтерфейс. З цією метою було розроблено блок-схему алгоритму роботи програми, наведену на рис. 3.1.

Робота користувача з програмним забезпеченням поділена на три етапи: обрання вебкамери, налаштування, робота з вебкамерою. Кожен з цих етапів має бути представлений окремою формою при розробці застосунку.

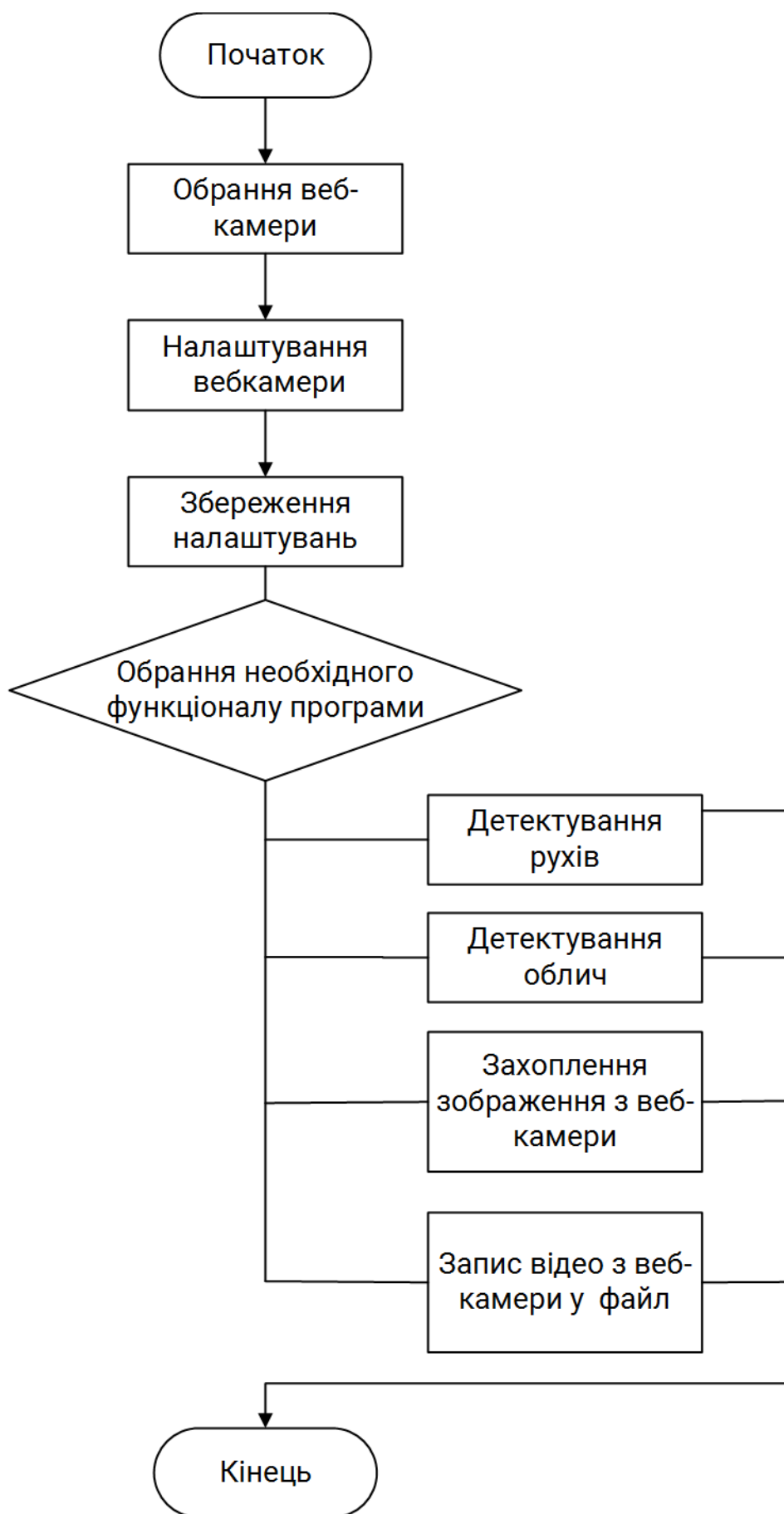


Рисунок 3.1 – Блок-схема алгоритму роботи програми

Для більш детального аналізу активностей користувача була розроблена діаграма прецедентів (рис. 3.2), що описує взаємодію користувача з програмним забезпеченням без прив'язки до часу. З цієї діаграми можна побачити зв'язаність окремих функцій системи, що також має великий інтерес при проєктуванні програмного забезпечення, наприклад, для оптимізації вибору і розміщення елементів керування на формах.

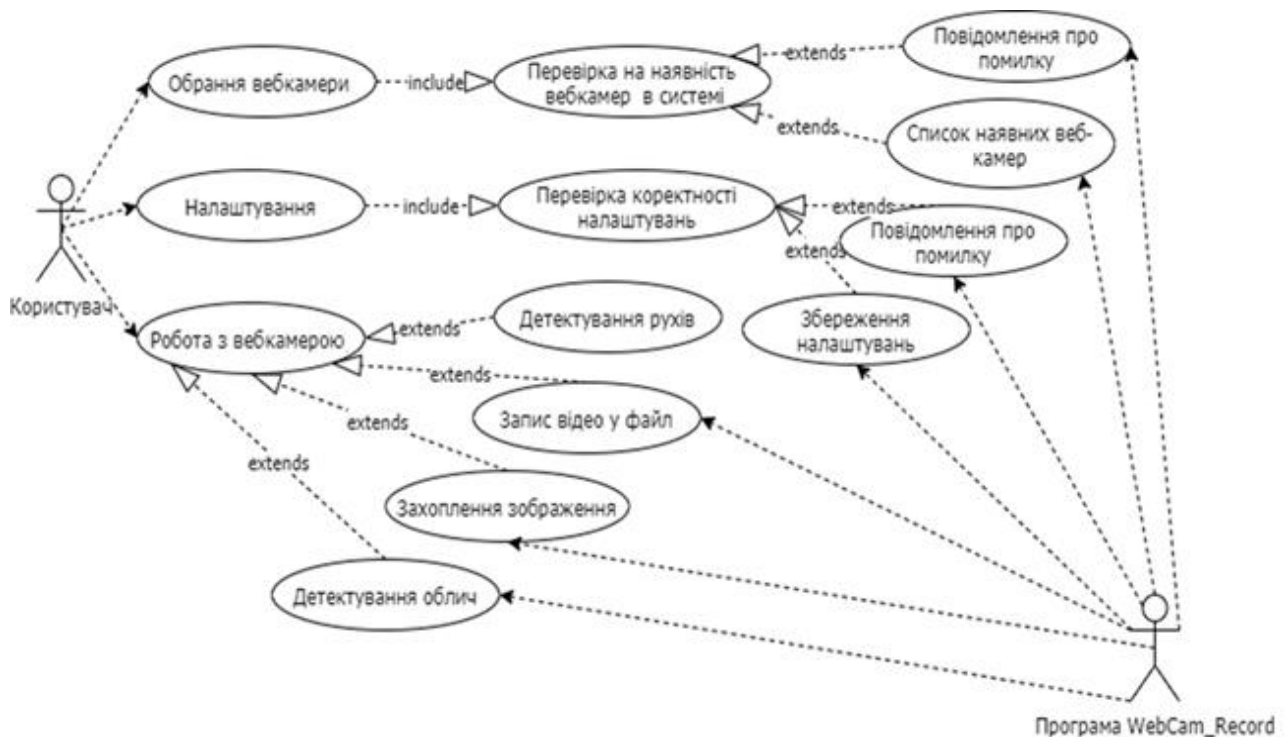


Рисунок 3.2 – Діаграма прецедентів

Дана діаграма прецедентів є моделлю в нотації мови UML, яка відображає:

- подання цілей взаємодії системи з користувачем;
- визначення і організацію функціональних вимог в системі;
- визначення контексту і вимог до системи;
- моделювання основного потоку подій в сценарії використання.

Відношення включення *include* на рис. 3.2 вказує на те, що поведінка одного прецеденту включається в деякій точці в інший прецедент в якості складового компонента. Відношення розширення *extends* відображає можливе

приєднання одного варіанту використання до іншого в певній точці (точці розширення) [20].

3.2 Розробка класів застосунку

Оскільки вказана у ТЗ мова програмування C# є виключно об'єктно-орієнтованою, то все функціонування програмного забезпечення виражається у взаємодії окремих класів, сутностей предметної області. Діаграма всіх класів застосунку «WebCam_Record» наведена на рис. 3.3.

Розроблені класи «WebCam_Record» (рис. 3.3) переважно являють собою класи, які успадковують від стандартного класу Windows Forms – Form, а саме: Main, Settings та Recording. Також стрілками з непереривною лінією зображено, у якій формі використовуються об'єкти класів, які не успадковують від Form: WebCamera та INIManager.

Об'єкти класу WebCamera містять дані про вебкамеру, тому вони використовуються у всіх формах: Main, Recording, Settings, передаючись як аргумент у конструкторі форми.

Об'єкти класу INIManager використовуються у класі Settings, бо саме при збереженні налаштувань необхідно записати налаштування у INI-файл, та у класі Main, де з нього потрібно зчитати налаштування вебкамери, якщо вони вже були задані і збережені.

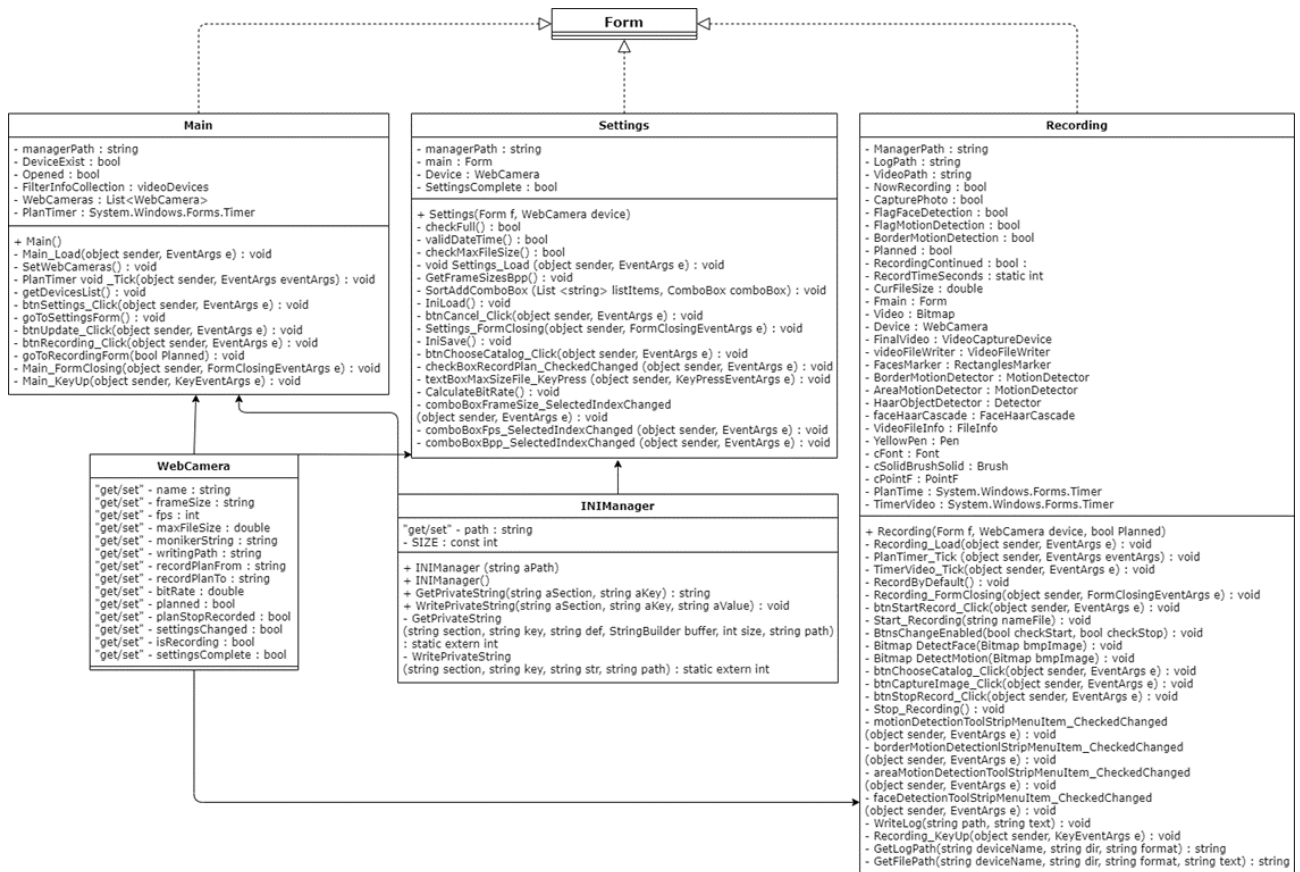


Рисунок 3.3 – Діаграма класів застосунку

Нижче описано кожний клас, що був розроблений, а також його поля і методи та їх призначення.

Форма Main є головною і містить наступні поля (табл. 3.1) і методи (табл. 3.2).

Список вебкамер отримуємо у функції SetWebCameras() за допомогою:

```
videoDevices = new FilterInfoCollection
```

```
(FilterCategory.VideoInputDevice)
```

Зберігаємо обрану вебкамеру у об'єкт класу WebCamera і якщо налаштування вже існують, заповнюємо ними, використовуючи їх у подальшому в інших формах.

Таблиця 3.1 – Поля класу-форми Main

Поле	Призначення
private	
string managerPath	Містить шлях до INI-файлу
bool DeviceExist	Показує, чи існує хоч одна вебкамера
bool Opened	Показує, чи слід відкривати вже форму запису за розкладом, або вона вже була відкрита
FilterInfoCollection videoDevices	Колекція усіх пристроїв введення відео
List<WebCamera> WebCameras	Список всіх вебкамер
System.Windows.Forms.Timer PlanTimer	Таймер для запису за розкладом

Таблиця 3.2 – Методи класу-форми Main

Метод	Призначення
1	2
public	
Main()	Конструктор класу, ініціалізує компоненти
private void	
Main_Load(object sender, EventArgs e)	Відбувається під час завантаження форми, присвоює змінним значення за замовчуванням, ініціалізує і запускає таймер
SetWebCameras()	Заповнює список вебкамер, задає їх дані з INI-файлу, якщо вони є у наявності
PlanTimer void _Tick(object sender, EventArgs eventArgs)	Відбувається кожну секунду, перевірює, чи були під час роботи програми змінені налаштування; порівнює запланований час з поточним, у разі збігу відкриває форму запису
getDevicesList()	Додає у список на формі всі вебкамери, якщо хоч одна є у наявності
btnSettings_Click(object sender, EventArgs e)	Після натискання на кнопку «Налаштування» викликає goToSettingsForm()
btnUpdate_Click(object sender, EventArgs e)	Після натискання кнопки «Оновити» оновлює список наявних вебкамер

Продовження таблиці 3.2

1	2
goToSettingsForm()	Відкриває форму налаштувань для обраної веб-камери і передає показчик на дану форму і об'єкт класу WebCamera, якщо хоч одна є у наявності
btnRecording_Click(object sender, EventArgs e)	Після натискання на кнопки налаштування викликає void goToRecordingForm(false), де false показує, що відкриття відбувається оператором, а не програмою за розкладом
goToRecordingForm(bool Planned)	Перевіряє, чи існують збережені налаштування для даної вебкамери. Якщо так, перевіряє, чи вже відкрита форма запису обраної вебкамери. Якщо ні, відкриває форму запису, передаючи показчик на дану форму і об'єкт класу WebCamera, true або false (запланований / незапланований запис)
Main_FormClosing(object sender, FormClosingEventArgs e)	Відбувається під час закриття форми, крім головної закриває всі видимі і невидимі форми
Main_KeyUp(object sender, KeyEventArgs e)	Оброблює натискання гарячих клавіш: F1 – закрити форму; F3 – оновити список наявних в системі вебкамер; F4 – перехід до форми налаштувань; F5 – перехід до форми запису.

Клас WebCamera призначений для зберігання даних про вебкамеру (табл. 3.3).

Таблиця 3.3 – Поля класу WebCamera

Поле	Призначення
1	2
private	
string name	Назва вебкамери
string frameSize	Роздільна здатність екрану
int fps	Кількість fps

Продовження табл. 3.3

1	2
double maxFileSize	Максимальний розмір відеофайлу
string monikerString	«Прізвисько» вебкамери, найменування FilterInfoCollection 3
string writingPath	Шлях до каталогу за замовчуванням
string recordPlanFrom	Час початку запланованого запису
string recordPlanTo	Час кінця запланованого запису
double bitRate	Бітова швидкість відео (бітрейт)
bool planned	Ознака запланованості
bool planStopRecorded	Показує, що запис відео за розкладом зупинено
bool settingsChanged	Показує, що налаштування були змінені
bool isRecording	Показує, чи відкрита форма запису
bool settingsComplete	Показує, що всі налаштування вебкамери збережено

До методів класу WebCamera належать наступні. Метод private WebCamera() є конструктором класу, задає значення змінних за замовчуванням. Public геттери і сеттери {get; set;} задають і повертають значення private полів класу, забезпечуючи інкапсуляцію.

Клас INIManager потрібний для роботи з INI-файлом (табл. 3.4).

Таблиця 3.4 – Поля та методи класу INIManager

Поле / Метод	Призначення
1	2
public	
INIManager (string aPath)	Конструктор, який приймає шлях до INI-файлу
INIManager()	Конструктор без аргументів (шлях до INI-файлу потрібно задати окремо)
string GetPrivateString(string aSection, string aKey)	Повертає значення з INI-файлу (за вказаною секцією і ключем)

Продовження табл. 3.4

1	2
void WritePrivateString(string aSection, string aKey, string aValue)	Записує значення у INI-файл (за вказаною секцією і ключем)
string Path	Геттер та сеттер, повертає або встановлює шлях до INI файлу
private	
const int SIZE	Максимальний розмір (для читання значення з файлу – 1024)
string path	Для зберігання шляху до INI-файлу
static extern int GetPrivateString (string section, string key, string def, StringBuilder buffer, int size, string path);	Повертає значення з INI-файлу (за вказаною секцією і ключем) з додатковими параметрами
static extern int WritePrivateString(string section, string key, string str, string path);	Записує значення до INI-файлу (за вказаною секцією і ключем) з додатковими параметрами

Форма «Settings» використовується для задання і збереження налаштувань вебкамери (табл. 3.5 – 3.6).

Таблиця 3.5 – Поля форми Settings

Поле	Призначення
private	
string managerPath	Шлях до INI-файлу
Form main	Головна форма
WebCamera Device	Обрана вебкамера
bool SettingsComplete	Ознака успішного збереження налаштувань

Бітова швидкість (бітрейт) у кбіт/с розраховується за формулою [21]:

$$\text{bitrate} = (\text{resolution} \times \text{FPS} \times \text{BPP}) / 1000, \quad (3.1)$$

де bitrate – бітова швидкість;

resolution – роздільна здатність;

FPS – кількість фреймів на секунду;

BPP – кількість бітів на піксель.

Таблиця 3.6 – Методи форми Settings

Метод	Призначення
1	2
public	
Settings(Form f, WebCamera device)	Конструктор форми, отримує вебкамеру (об'єкт класу WebCamera) та головну форму
private bool	
checkFull()	Перевіряє заповненість всіх обов'язкових полів
validDateTime()	Перевіряє правильність введеної дати
checkMaxFileSize()	Перевіряє, що максимальний розмір файлу від 1 Мб до 1 Гб
private void	
void Settings_Load (object sender, EventArgs e)	Відбувається під час завантаження форми, присвоює змінним значення за замовчуванням, ініціалізує і запускає таймер
GetFrameSizesBpp()	Отримує з вебкамери список доступних роздільної здатності та bpp
SortAddComboBox (List <string> listItems, ComboBox comboBox)	Сортує та додає у ComboBox список
IniLoad()	Завантажує налаштування вебкамери з INI-файлу, якщо вони наявні
btnCancel_Click(object sender, EventArgs e)	Після натискання кнопки «Відмінити» закриває форму, не зберігаючи зміни
Settings_FormClosing(object sender, FormClosingEventArgs e)	Відбувається під час закриття форми, відображує головну форму

Продовження табл. 3.6

1	2
IniSave()	Використовує INIManager, записує дані про веб-камеру з полів у INI-файл
btnChooseCatalog_Click(object sender, EventArgs e)	Після натискання кнопки «Обрати» відкриває діалог для вибору каталогу
checkBoxRecordPlan_CheckedChanged (object sender, EventArgs e)	В залежності від стану прапорця «За розкладом» змінює доступність полів задання дати
textBoxMaxSizeFile_KeyPress (object sender, KeyPressEventArgs e)	Обмежує ввід максимального розміру файлу клавішами цифр та комою
CalculateBitRate()	Розраховує відповідну бітову швидкість (бітрейт)
comboBoxFrameSize_SelectedIndexChanged (object sender, EventArgs e)	При зміні обраного індексу в ComboBox викликають CalculateBitRate()
comboBoxFps_SelectedIndexChanged (object sender, EventArgs e)	
comboBoxBpp_SelectedIndexChanged (object sender, EventArgs e)	

Перед переходом на форму «Recording» відбувається перевірка на відкритість форми з даною вебкамерою, потім на запланованість розкладу, яка передається у форму ознакою bool Planned разом з вебкамерою, поля якої заповнені даними з налаштувань, якщо вони були збережені. Форма відкривається на тій самій позиції лівого верхнього краю, що й головна.

```
private void goToRecordingForm(bool Planned)
{
    int ind = comboBoxDevices.SelectedIndex;
    if (WebCameras[ind].SettingsComplete)
    {
        if (!WebCameras[ind].IsRecording)
        {
            Form frmSet;
            if (Planned) frmSet = new Recording(this,
WebCameras[ind], true);
            else frmSet = new Recording(this,
WebCameras[ind], false);
            frmSet.Left = this.Left;
```

```

        frmSet.Top = this.Top;
        frmSet.Show();
        WebCameras[ind].IsRecording = true;
    }
    else MessageBox.Show("Форма з записом веб-
камери " + WebCameras[ind].Name + " вже відкрита!");
    }
    else
    {
        MessageBox.Show("Задайте налаштування веб-
камери, будь ласка!");
        goToSettingsForm();
    }
}

```

Форма «Recording» використовується для роботи з записом відео, детектуванням, захопленням зображення з вебкамери (табл. 3.7 – 3.8).

Таблиця 3.7 – Поля форми Recording

Поле	Призначення
1	2
private	
string ManagerPath	Шлях до INI-файлу
string LogPath	Шлях до лог-файлу
string VideoPath	Шлях до збереженого відео
bool NowRecording	Показує, чи відбувається запис відео у даний момент
bool CapturePhoto	Показує, чи було обрано захоплення зображення
bool FlagFaceDetection	Показує, чи було обрано детектування облич
bool FlagMotionDetection	Показує, чи було обрано детектування руху
bool BorderMotionDetection	Показує, чи було обрано детектування виділенням границь руху
bool Planned	Показує, чи був запис запланованим
bool RecordingContinued	Визначає, чи був досягнутий максимальний розмір файлу і чи потрібно почати запис у новому з назвою за замовчуванням

Продовження табл. 3.7

1	2
static int RecordTimeSeconds	Кількість секунд з початку запису
double CurFileSize	Розмір поточного файлу, у який здійснюється запис відео
Form Fmain	Головна форма
Bitmap Video	Зображення з вебкамери, необхідне для запису і відображення відео
WebCamera Device	Обрана на головній формі вебкамера
VideoCaptureDevice FinalVideo	Вебкамера для запису відео
VideoFileWriter videoFileWriter	Необхідний для запису відео у файл
RectanglesMarker FacesMarker	Маркер, що виділяє детектовані обличчя
MotionDetector BorderMotionDetector	Детектор руху меж
MotionDetector AreaMotionDetector	Детектор руху зони
HaarObjectDetector Detector	Детектор об'єктів
FaceHaarCascade faceHaarCascade	Детектор облич
Pen YellowPen	Жовта «Ручка», що необхідна для малювання прямокутників
FileInfo VideoFileInfo	Необхідний для створення нового відеофайлу за замовчуванням
Font cFont	Шрифт для тексту
SolidBrush cSolidBrush	«Пензель» помаранчевого кольору для тексту лівого верхнього кута
PointF cPointF	Координати для тексту лівого верхнього кута
System.Windows.Forms.Timer PlanTime	Таймер планування за розкладом, визначає, коли слід зупиняти відео і закривати форму
System.Windows.Forms.Timer TimerVideo	Таймер для відображення кількості часу з початку запису відео в файл

Основний функціонал пов'язаний з подією void FinalVideo_NewFrame(object sender, NewFrameEventArgs eventArgs), яка є

функцію, що створюється на подію нового фрейму об'єкту класу VideoCaptureDevice:

```
FinalVideo = new VideoCaptureDevice(Device.MonikerString);
FinalVideo.NewFrame += new
NewFrameEventHandler(FinalVideo_NewFrame);
FinalVideo.Start();
```

Таблиця 3.8 – Методи форми Recording

Метод	Призначення
1	2
public	
Recording(Form f, WebCamera device, bool Planned)	Виконується під час завантаження форми, отримує вебкамеру (об'єкт класу WebCamera), головну форму та ознаку запланованості
private void	
Recording_Load(object sender, EventArgs e)	Відбувається під час завантаження форми, присвоює змінним значення за замовчуванням, ініціалізує і запускає необхідні таймери та відображення відео з вебкамери
PlanTimer_Tick (object sender, EventArgs eventArgs)	Відбувається кожну секунду, порівнює час кінця запису за розкладом з поточним, зупиняє запис і закриває форму у випадку збігу
TimerVideo_Tick(object sender, EventArgs e)	Відбувається кожну секунду, відображує рядок с часом від початку запису
RecordByDefault()	Задає назву за замовчанням і починає запис
Recording_FormClosing(object sender, FormClosingEventArgs e)	Відбувається під час закриття форми, зупиняє таймер і запис відео, якщо не був зупинений натисканням на кнопку
btnStartRecord_Click(object sender, EventArgs e)	Викликає Start_Recording(""), у якому користувач повинен обрати ім'я файлу для запису відео
BtnsChangeEnabled(bool checkStart, bool checkStop)	Змінює доступність кнопок початку і завершення запису на необхідну.

Продовження табл. 3.8

1	2
Start_Recording(string nameFile)	Починає запис відео у файл з назвою, якщо задана: за замовчуванням, якщо ні – використовує задане оператором ім'я файлу у діалозі збереження. Для запису використовує дані з об'єкта класу WebCamera
Bitmap DetectFace(Bitmap bmpImage)	Детектує всі обличчя на зображенні, що передається
Bitmap DetectMotion(Bitmap bmpImage)	Детектує рух порівнянням двох фреймів
btnChooseCatalog_Click(object sender, EventArgs e)	Після натискання кнопки «Обрати» відкриває діалог для вибору каталогу
btnCaptureImage_Click(object sender, EventArgs e)	Після натискання кнопки «Захопити зображення» робить ознаку захоплення зображення = true
btnStopRecord_Click(object sender, EventArgs e)	Після натискання кнопки «Зупинити запис» викликає Stop_Recording() та зупиняє таймер, що рахує час від початку запису
Stop_Recording()	Зупиняє запис відео у файл
motionDetectionToolStripMenuItem_CheckedChanged (object sender, EventArgs e)	Відбувається під час зміни стану прапорця «Детектування руху», необхідний для зміни відповідних ознак детектування зони та границі руху
borderMotionDetectionToolStripMenuItem_CheckedChanged (object sender, EventArgs e)	Відбувається під час зміни стану прапорця «Виділення контурів руху» і змінює відповідні ознаки
areaMotionDetectionToolStripMenuItem_CheckedChanged (object sender, EventArgs e)	Відбувається час зміни стану прапорця «Виділення зони руху» і змінює відповідні ознаки
WriteLog(string path, string text)	Створює лог-файл, якщо його не існує, записує в нього переданий текст
faceDetectionToolStripMenuItem_CheckedChanged (object sender, EventArgs e)	Відбувається під час зміни стану прапорця «Детектування обличч» і змінює відповідні ознаки

Продовження табл. 3.8

1	2
Recording_KeyUp(object sender, KeyEventArgs e)	Оброблює натискання гарячих клавіш: F1 – закрити форму; F6 – ввімкнути / вимкнути детектування руху; F7 – ввімкнути детектування руху виділенням контуру; F8 – ввімкнути детектування руху виділенням зони руху; F9 – ввімкнути / вимкнути детектування облич; F10 – захопити зображення; F11 – почати запис відео; F12 – завершити запис відео.
private string	
GetLogPath(string deviceName, string dir, string format)	Створює і повертає шлях до лог-файлу для даної веб-камери
GetFilePath(string deviceName, string dir, string format, string text)	Створює і повертає шлях до файлу за замовчуванням

Захоплення зображення відбувається за ознакою `CapturePhoto = true`:

```
Video.Save(ImgPath, System.Drawing.Imaging.ImageFormat.Bmp),
де Video = (Bitmap)eventArgs.Frame.Clone();
```

Запис файлу та перевірку на досягнення максимального заданого розміру (`CurFileSize`) відбувається за ознакою `NowRecording = true`:

```
VideoFileInfo = new System.IO.FileInfo(VideoPath);
CurFileSize = (VideoFileInfo.Length / 1024.0 / 1024.0);
if (CurFileSize >= Device.MaxFileSize)
{
    RecordingContinued = true;
    Stop_Recording();
}
```

Детектування обличчя відбувається за ознакою `NowRecording = true`, результат присвоюється `pictureBoxVideo.Image`:

```
if (FlagFaceDetection)
{
    Bitmap bmpImage = (Bitmap)eventArgs.Frame.Clone();
```

```

        pictureBoxVideo.Image = DetectFace (bmpImage);
    }

```

Результат детектування облич на фреймі `Detector.ProcessFrame(bmpImage)` у вигляді прямокутників зберігається у масиві `faces`, за допомогою `_g.DrawRectangle(YellowPen, face)` малюється кожний з них жовтим кольором «ручкою» з заданою на початку товщиною, та малюється помаранчевим кольором рядок «Детектовано обличчя»:

```

private Bitmap DetectFace(Bitmap bmpImage)
{
    Rectangle[] faces =
Detector.ProcessFrame(bmpImage);
    if (faces.Length != 0)
    {
        Graphics _g = Graphics.FromImage(bmpImage);
        foreach (Rectangle face in faces)
        {
            _g.DrawRectangle(YellowPen, face);
            _g.DrawString("Детектовано обличчя", cFont,
cSolidBrush, cPointF);
            _g.Dispose();
        }
        return bmpImage;
    }
}

```

Виклик функції детектування руху викликається аналогічним чином до функції детектування облич. Зображення з вебкамери `bmpImage` оброблюється або за допомогою детектора меж, або детектора зони руху. У результаті в `motionDetector` зберігається числове значення, якщо воно перевищує 0.02, округлюється до трьох знаків після коми, виводиться помаранчевий рядок, що його містить, відбувається відповідний запис у лог-файл:

```

private Bitmap DetectMotion(Bitmap bmpImage)
{
    try
    {
        double motionDetector;
        if (BorderMotionDetection)
        {
            motionDetector =
BorderMotionDetector.ProcessFrame(bmpImage);
        }
        else
        {
            motionDetector =
AreaMotionDetector.ProcessFrame(bmpImage);
        }
    }
    catch { }
}

```

```

    }
    if (motionDetector > 0.02)
    {
        Graphics _g =
Graphics.FromImage(bmpImage);
        double motion =
Math.Round(motionDetector, 3);
        WriteLog(LogPath, "Зафіксовано рівень
пуху: " + motion.ToString());
        _g.DrawString("Рівень пуху:" +
motion.ToString("{0.0,00}"), cFont, cSolidBrush, cPointF);
        _g.Dispose();
    }
}
catch (Exception) { }
return bmpImage;
}

```

3.3 Розробка графічного інтерфейсу користувача

На етапі проектування системи (див. п.3.1) було зазначено, що розроблюване ПЗ повинно складатись щонайменше з трьох форм:

- головної форми, що надає доступ до інших форм;
- форми налаштувань, за допомогою якої можна налаштувати запис відео з вебкамери;
- форму запису, за допомогою якої користувач має можливість обрати режими детектування, захопити зображення та записати відео у файл з обраним ім'ям.

Вибір елементів керування проводиться відповідно до активностей користувача на формі. Таким чином, на початковій формі необхідно розмістити: дві кнопки (Button), натискаючи на котрі користувач здійснює переходи до інших форм застосунку; список для обрання необхідної для запису зображення вебкамери (ComboBox); кнопку для оновлення даних та статичний текст (Label) (рис. 3.4).

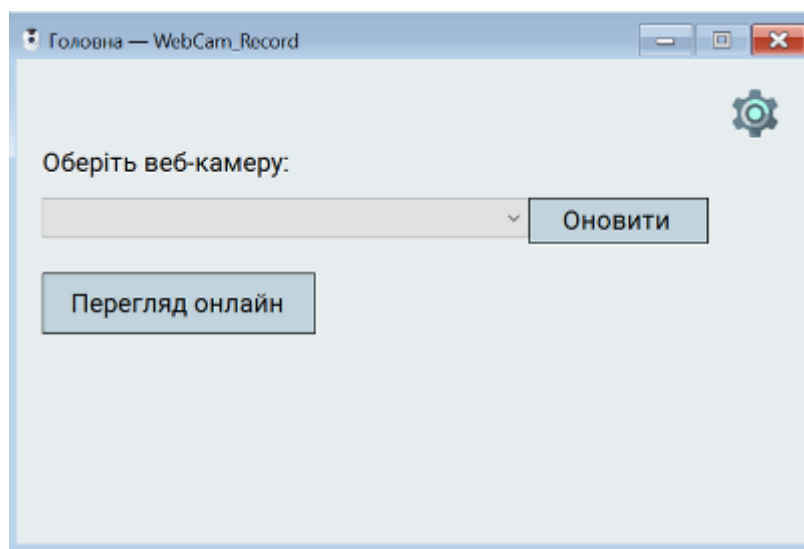


Рисунок 3.4 – Прототип інтерфейсу форми «Головна»

Форма налаштувань повинна містити всі необхідні поля, а саме: Label для статичного тексту, ComboBox (список с заданими елементами) для обрання одного варіанту зі списку, CheckBox (прапорець) для ознаки запису за розкладом, DateTimePicker (дата/час) для часу початку і кінця розкладу, TextBox (текстові поля) та Button (кнопки) для збереження і відміни (рис. 3.5).

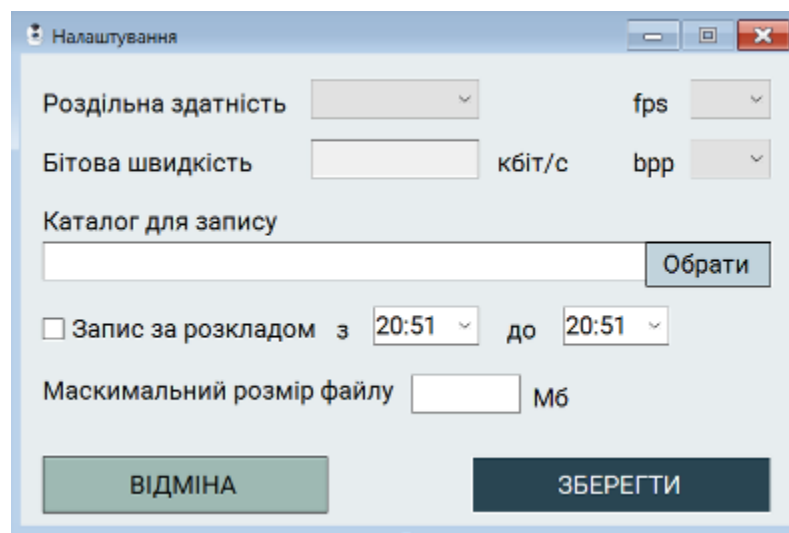


Рисунок 3.5 – Прототип інтерфейсу форми «Налаштування»

Форма запису має містити ToolStrip (верхнє меню) з його елементами, у яких можна відмічати прапорець, ToolTip (підказки при наведенні на елементи

керування), кнопки з відповідними зображеннями, Label (текст) для відображення статусу та таймеру, Panel (контейнер для інших елементів керування, необхідний для того, щоб кнопки підлаштовувались під розмір форми, який можна змінювати) та PictureBox для зображення з вебкамери (рис. 3.6).

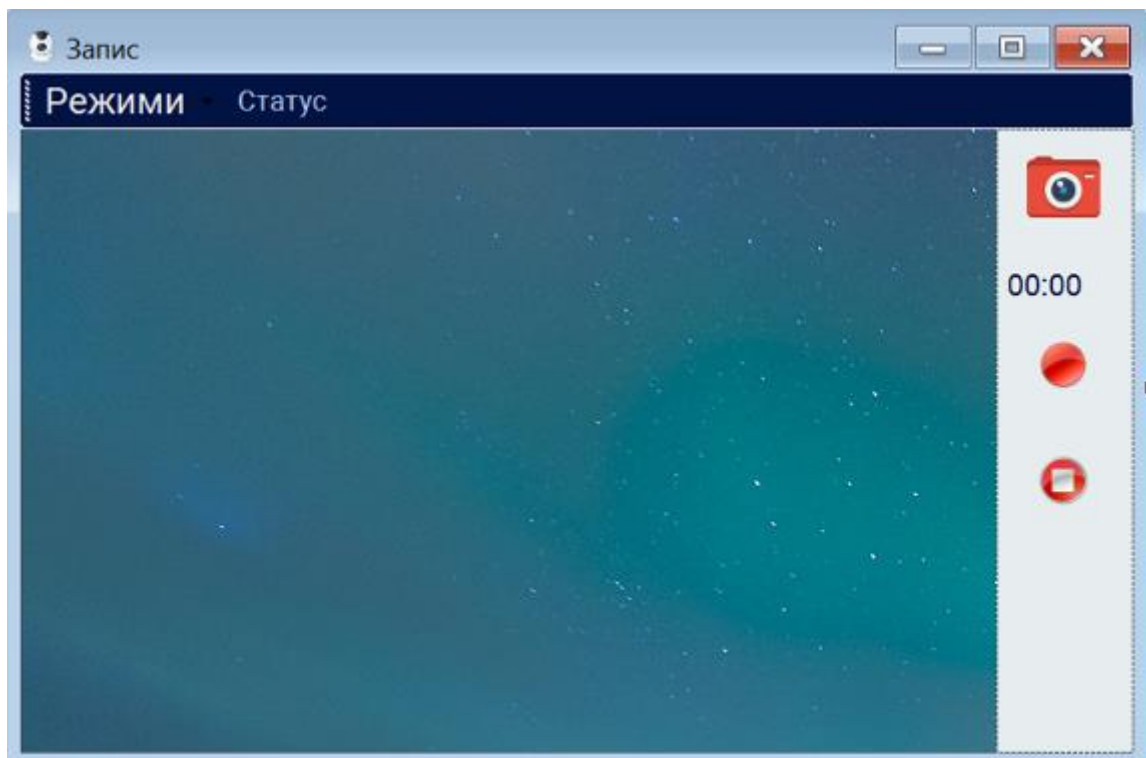


Рисунок 3.6 – Прототип інтерфейсу форми «Запис»

Обробка подій елементів керування описана у таблицях форми в п. 3.2. Оброблюються події натискання кнопок (Click), відпускання клавіші (KeyUp), зміни обраного індексу ComboBox (SelectedIndexChanged), зміни стану прапорця (CheckedChanged), кожного «тіку» таймеру з заданим інтервалом (Tick).

Призначення кожного обробника подій елементів керування було розглянуто в методах форм у попередньому пункті.

У методах форми Recording (табл. 3.8) часто використовується наступна конструкція у вигляді:

```
this.Control.Invoke((MethodInvoker)delegate {
```

```
//код, пов'язаний з властивостями елемента керування
Control});
```

Її використання дозволяє працювати з елементами керування форми, при цьому створюючи нові потоки (при створенні відео за замовчуванням, коли розмір файлу запису досяг максимального).

Делегати схожі на покажчики функцій. Вони інкапсулюють посилання на метод всередині об'єкта делегата. Потім об'єкт делегата може бути переданий в код, що викликає згадуваний метод, а метод, що викликається може бути невідомим під час компіляції. На відміну від покажчиків функцій делегати є об'єктно-орієнтованими, строго типізованими і більш безпечними [22].

Усі форми виконані з використанням одного набору кольорів, простого для читання шрифту Roboto та повторюваних геометричних форм елементів керування. Дизайн виконано у синьо-червоній та синьо-зеленій палітрі, кнопки для збереження відео відрізняються від кнопок відміни.

Завжди доступна лише кнопка або початку, або завершення запису відео, неактивність елементів передано сірим кольором:

```
private void BtnsChangeEnabled(bool checkStart, bool
checkStop)
{
this.btnStopRecord.Invoke( (MethodInvoker)delegate
{
this.btnStopRecord.Enabled = checkStop;
});
this.btnStartRecord.Invoke( (MethodInvoker)delegate
{
this.btnStartRecord.Enabled = checkStart;
});
}
```

Код програми наведено у додатку А.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення і умови застосування програми

Програмне забезпечення «WebCam_Record» призначене для запису відео з вебкамери, збереження його у пам'яті комп'ютера в заданому каталозі та здійснення відеоспостереження.

Для коректного функціонування даного ПЗ необхідна версія ОС Windows 7 та вище, а також наявність вбудованої та/або USB вебкамери, монітору та комп'ютерної миші. Більш детально про вимоги до апаратних засобів описано у п. 1.3.4.

4.2 Характеристики програми

Програмне забезпечення «відловлює» можливі помилки в роботі оператора, запобігаючи від можливих помилок в його роботі. Усі повідомлення, що отримуються оператором під час виключних ситуацій описані у п. 5.4.

Для запуску програмного забезпечення необхідно виконати файл WebCam_Record.exe. Програма не потребує та не підтримує виклик з додатковими аргументами з командного рядка.

Перелік усіх класів програми, призначення їх полів і методів, а також особливостей функцій наведено у п. 3.2.

4.3 Початкові та вихідні дані

До вхідних даних програми належать область екрану для захоплення зображення або наявне зображення у пам'яті комп'ютера, позиція курсору під час роботи з інструментами редагування, для позначення початкових та кінцевих точок фігур, параметри інструментів редагування (колір, товщина, шрифт, текст, радіус розмиття тощо).

До вихідних даних розробленого ПЗ належать:

- збережений у обраному користувачем каталозі запис відео з вебкамери у форматі .AVI;
- захоплене зображення у форматі .BMP;
- збережені налаштування вебкамер у .INI-файлі в каталозі [Шлях до проекту]\WebCam_Record\WebCam_Record\bin\Debug;
- лог-файл дій користувача та позитивних результатів детектування на формі запису у форматі .LOG для кожної вебкамери в каталозі [Шлях до проекту]\WebCam_Record\WebCam_Record\bin\Debug;.

4.4 Повідомлення програмісту

Повідомлення програмісту у програмі не передбачені.

5 КЕРІВНИЦТВО ОПЕРАТОРА

5.1 Призначення програми

Програмне забезпечення «WebCam_Record» призначене для запису відео з вебкамери, збереження його в заданому каталозі та здійснення відеоспостереження.

Для коректного функціонування програмного забезпечення необхідна версія ОС Windows 7 та вище, а також наявність вбудованої або USB вебкамери, монітору та комп'ютерної миші. Більш детально про вимоги до апаратних засобів описано у п. 1.3.4. Гарячі клавіші управління даної програми наведені у табл. 3.2 та 3.8.

5.2 Умови використання програми

Умови використання застосунку зазначені у технічному завданні (див п.п. 1.3.4, 1.3.6 та 1.3.8).

5.3 Виконання програми

Для запуску програмного забезпечення необхідно виконати файл WebCam_Record.exe. ПЗ не потребує та не підтримує виклик з додатковими аргументами з командного рядка.

Після запуску програми відкривається головна форма (рис. 5.1).

Після натискання кнопки «Оновити», програма оновлює список наявних в системі вебкамер (наприклад, якщо відключили або підключили додаткову вебкамеру). Спочатку необхідно обрати бажану вебкамеру зі списку (рис. 5.2).

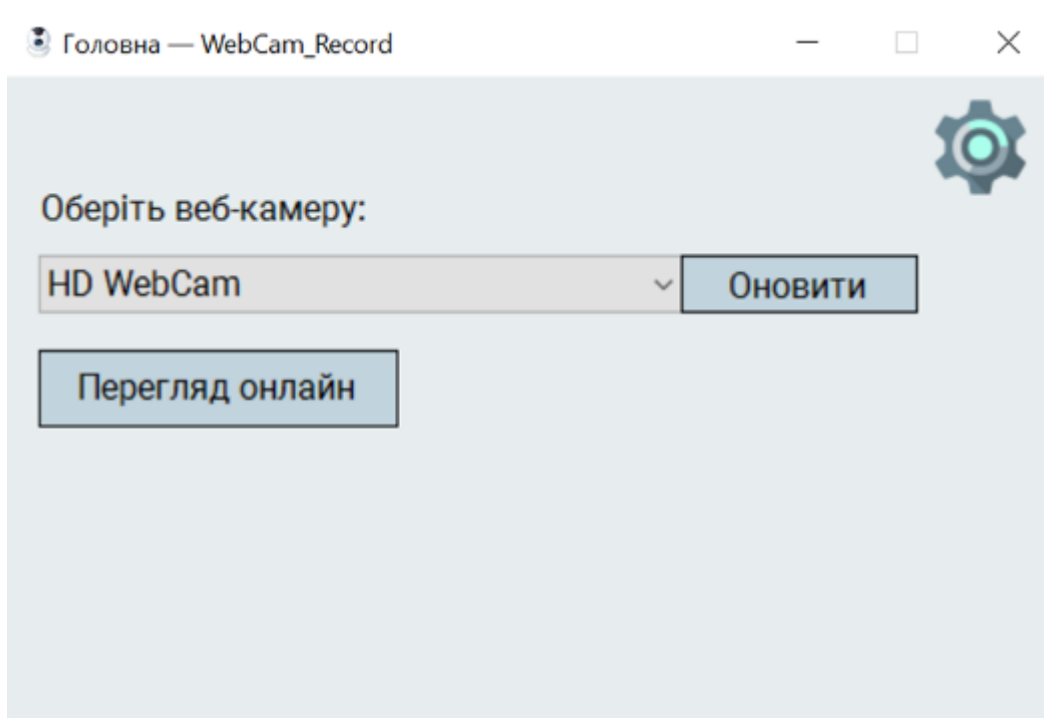


Рисунок 5.1 – Головна форма програми

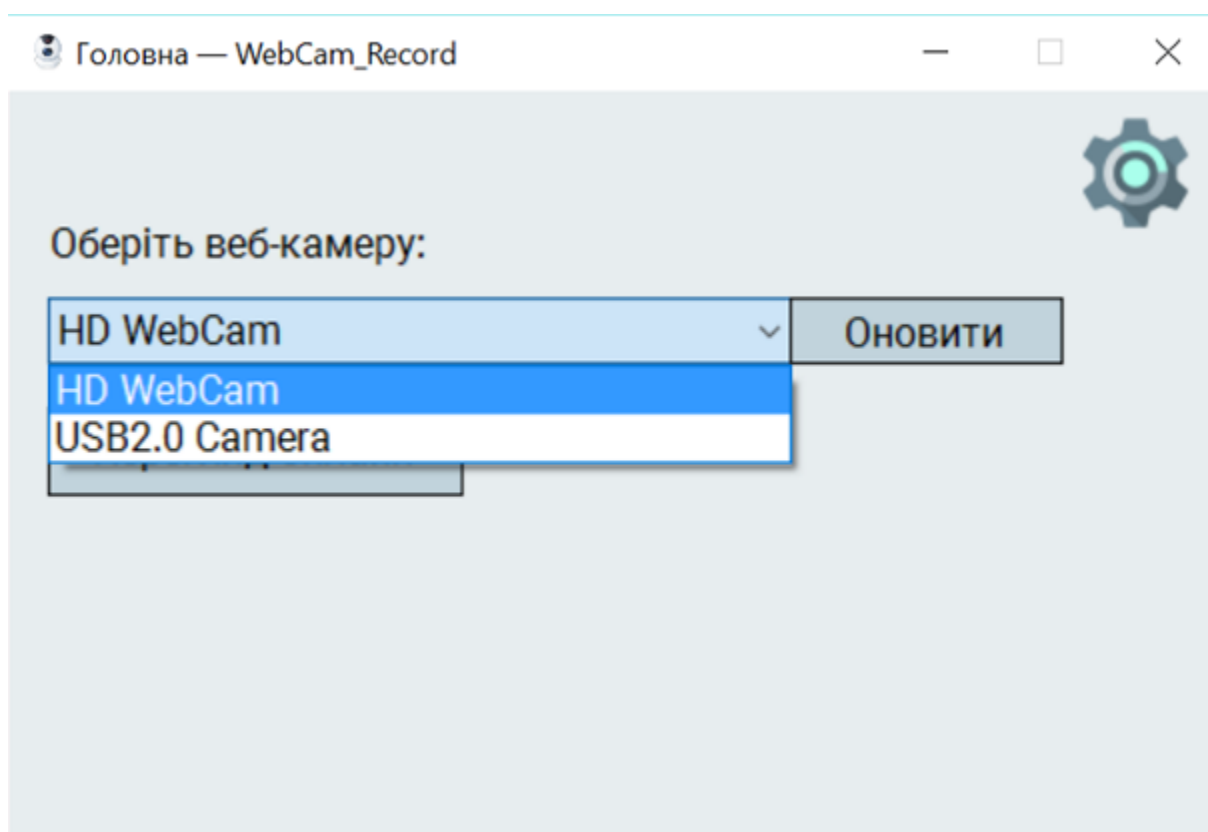


Рисунок 5.2 – Обрання вебкамери

Натиснувши верхню праву кнопку у вигляді шестерні (рис. 5.3), переходимо до форми «Налаштування» (рис. 5.4)

Необхідно обрати роздільну здатність відео з доступних для даної вебкамери.

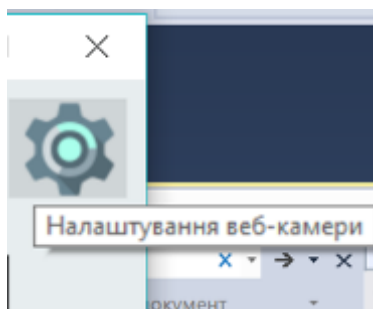


Рисунок 5.3 – Кнопка «Налаштування вебкамери»

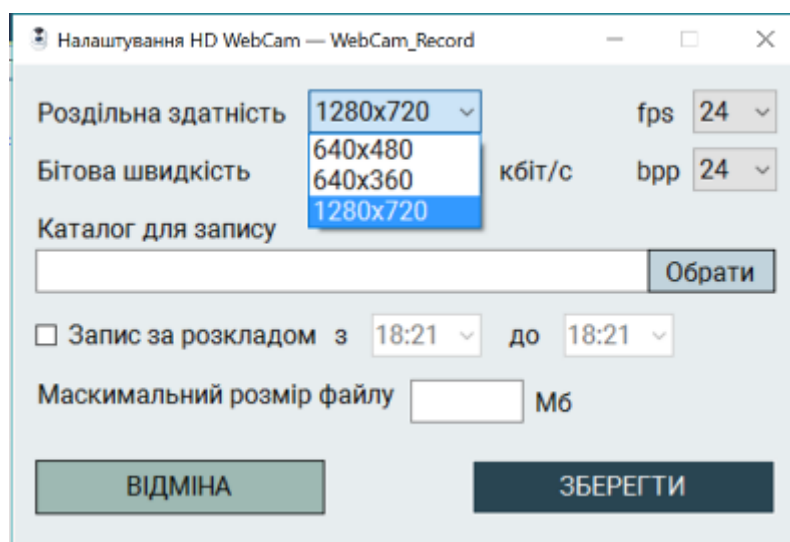


Рисунок 5.4 – Обрання роздільної здатності відео

Після цього потрібно обрати кількість FPS (Frames Per Second, кількість кадрів в секунду). За замовчуванням обрано 24 fps (рис. 5.5).

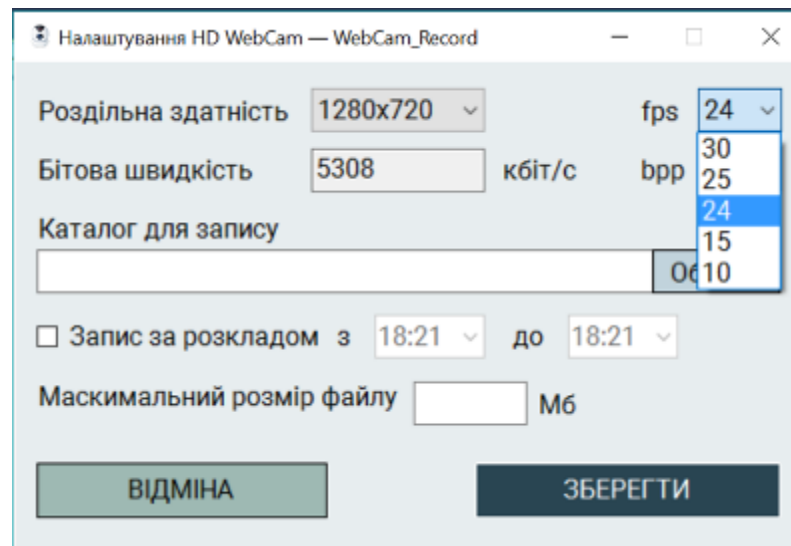


Рисунок 5.5 – Обрання fps відео

Далі необхідно обрати кількість BPP (Bits Per Pixel, кількість бітів на один піксель). За замовчуванням обрано найбільше bpp з можливих для вебкамери, що налаштовується (рис. 5.6).

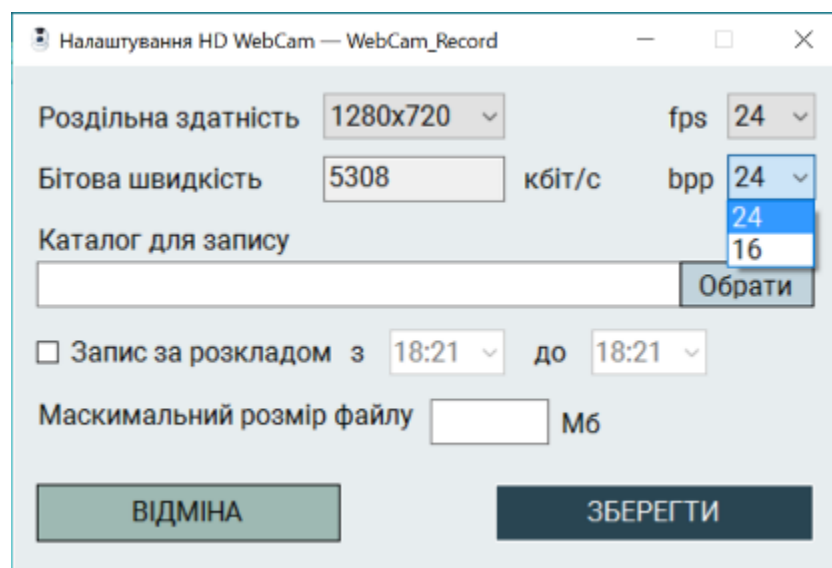


Рисунок 5.6 – Обрання bpp відео

Після зміни роздільної здатності, bpp чи fps, повторно буде розраховано оптимальну бітову швидкість (Bit Rate).

Щоб задати каталог для запису, необхідно ввести повний шлях вручну або натиснути кнопку «Обрати». Відкриється діалог обрання каталогу (рис. 5.7).

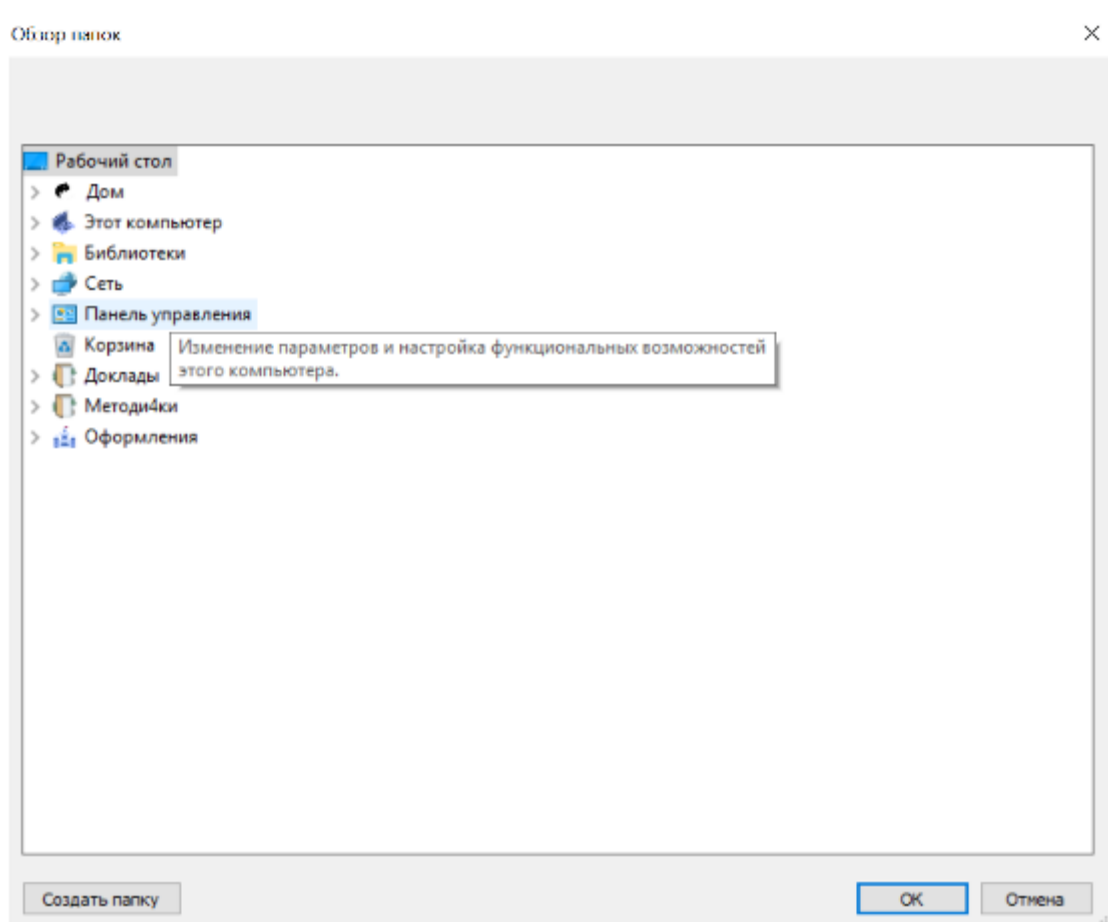


Рисунок 5.7 – Обрання каталогу, де будуть за замовчуванням зберігатись відео

Після того, як буде відмічений прапорець «Запис за розкладом» необхідно задати час запису (рис. 5.8).

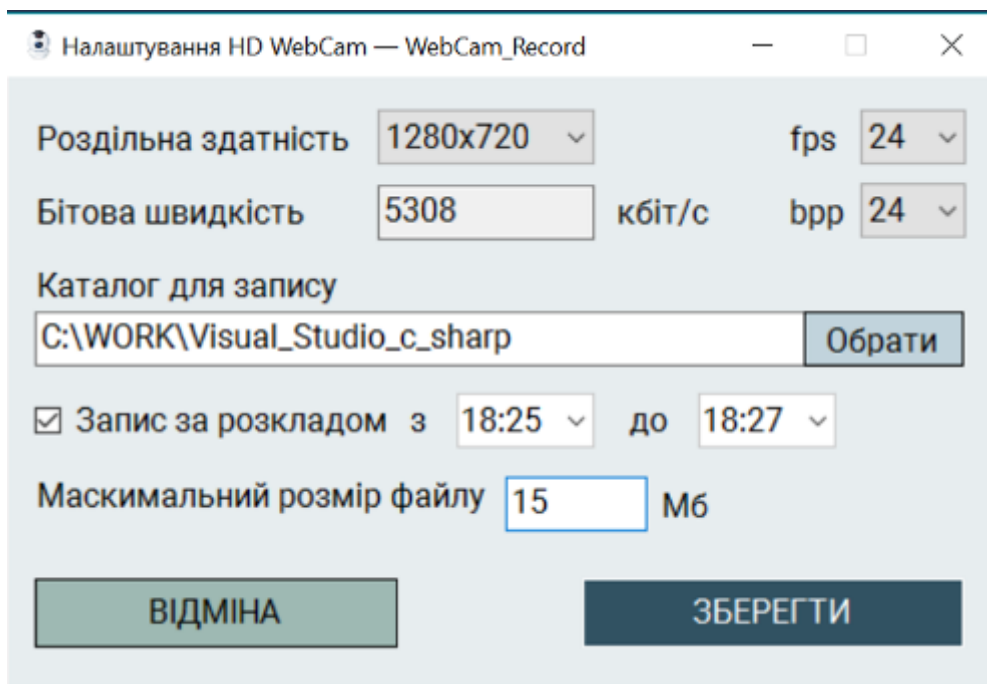
Слід зауважити, що запис буде відбуватися, якщо відкрита головна форма і форма запису необхідної вебкамери закрита.

У відповідний час початку запису відкриється форма запису та почне запис за замовчуванням, враховуючи максимальний розмір файлу. У момент кінця запису відео зупиниться і форма закриється.

Також необхідно ввести максимальний розмір файлу (від 1 до 1024 Мб).

Щоб зберегти налаштування, необхідно натиснути відповідну кнопку. У випадку натиснення «Відміна» зміни не будуть збережені.

Форма закриється і знову відкривається форма «Головна».



Налаштування HD WebCam — WebCam_Record

Роздільна здатність 1280x720 fps 24

Бітова швидкість 5308 кбіт/с bpp 24

Каталог для запису
C:\WORK\Visual_Studio_c_sharp Обрати

☒ Запис за розкладом з 18:25 до 18:27

Максимальний розмір файлу 15 Мб

ВІДМІНА ЗБЕРЕГТИ

Рисунок 5.8 – Збереження заповнених налаштувань

Аналогічним чином можна обрати іншу наявну у системі вебкамеру та задати її налаштування. Для того, щоб перейти на форму запису відео, необхідно натиснути на кнопку «Перегляд онлайн». Відкриється форма «Запис» (рис. 5.9).

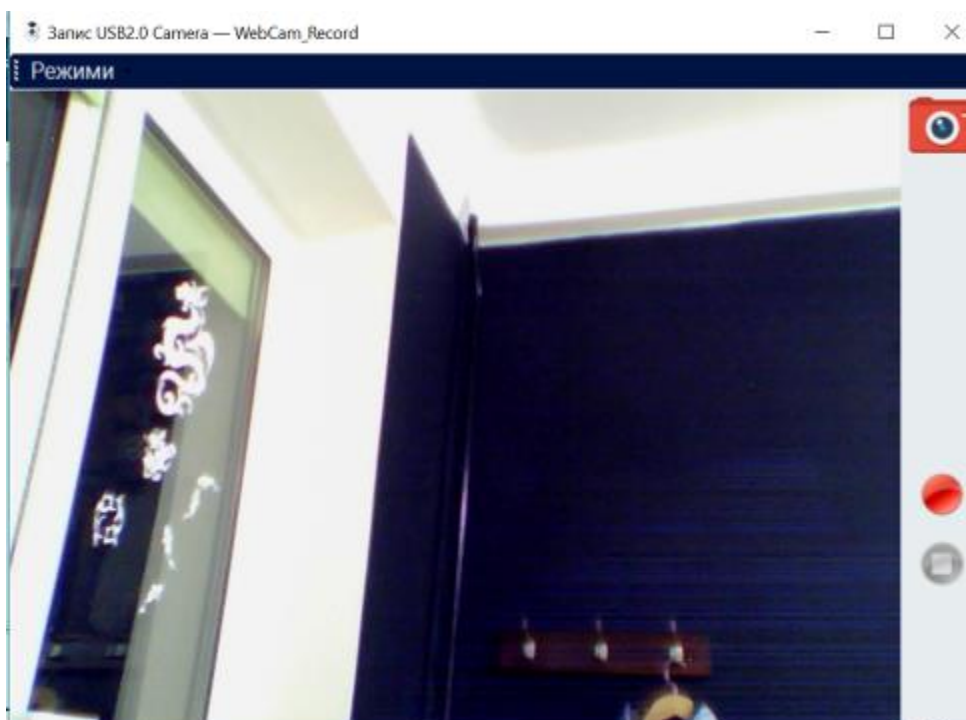


Рисунок 5.9 – Форма запису відео

Необхідно натиснути на кнопку у вигляді камери, щоб захопити зображення з вебкамери. Воно буде збережено в каталог за замовчуванням (назва – поточний час: години_хвилини_секунди). Даний шлях до зображення буде відображено у верхньому рядку статусу (рис. 5.10).

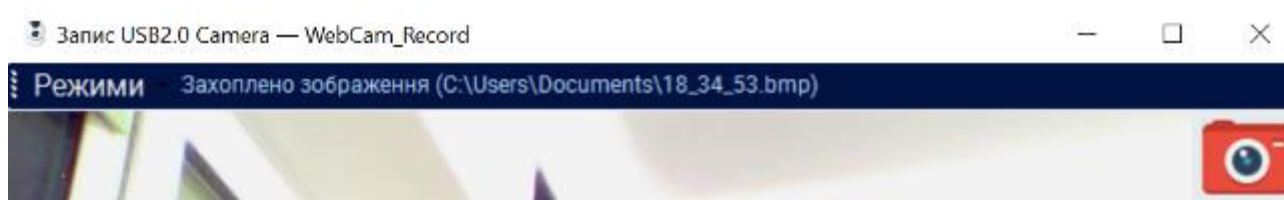


Рисунок 5.10 – Шлях до зображення у рядку статусу

Для того щоб увімкнути/вимкнути детектування рухів, потрібно натиснути Режими → Детектування рухів (рис. 5.11).

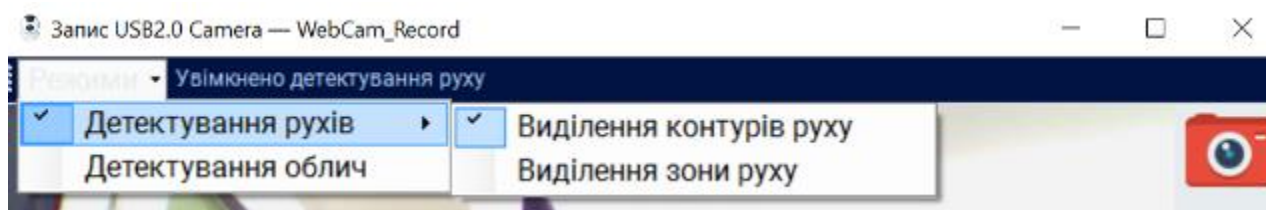


Рисунок 5.11 – Увімкнення детектування руху

За замовчуванням буде здійснюватися виділення контурів руху червоним маркером (рис. 5.12). У верхньому лівому куті при детектуванні руху >0.02 виводиться текст помаранчевого кольору з поточним рівнем руху.



Рисунок 5.12— Демонстрація детектування руху (виділення контурів)

Щоб увімкнути виділення зони руху, необхідно обрати відповідний пункт **Режими** → **Детектування рухів** → **Виділення зони руху** (рис. 5.13).

Демонстрація детектування руху (виділення зони) проводиться шляхом виділення червоним кольором меж зони переміщення об'єкта. Приклад виділення зони переміщення пальців руки наведено на рис. 5.14.

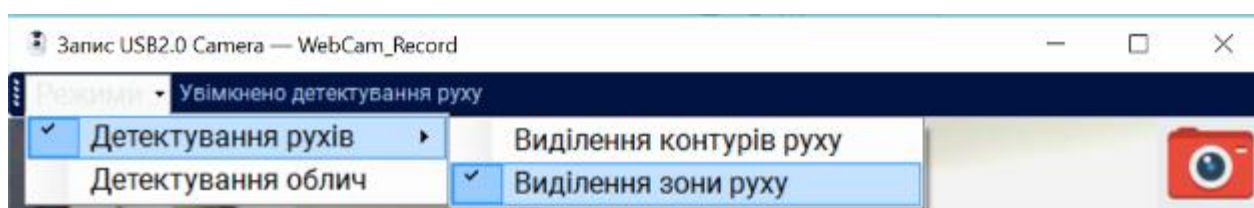


Рисунок 5.13 – Увімкнення виділення зони руху

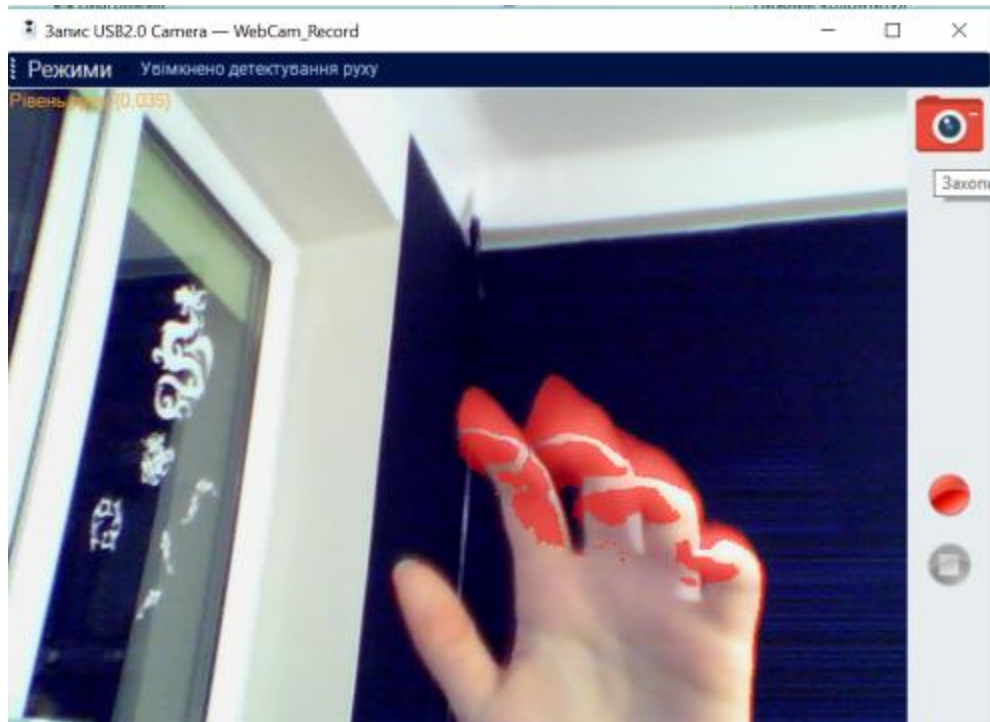


Рисунок 5.14 – Демонстрація детектування руху (виділення зони)

Щоб увімкнути/вимкнути детектування обличчя, необхідно обрати пункт меню **Режими** → **Детектування рухів**. У верхньому лівому куті, якщо буде детектовано хоч одне обличчя, виводиться текст помаранчевого кольору «Детектовано обличчя» та виділяється жовтою рамкою (рис. 5.15).

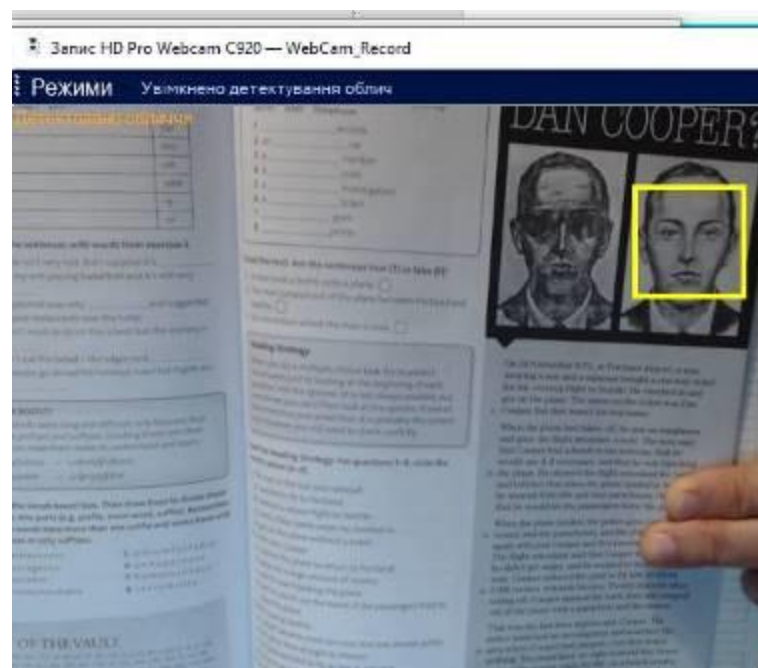


Рисунок 5.15 – Демонстрація детектування обличчя

Для того, щоб почати запис, необхідно натиснути круглу червону кнопку (рис. 5.16).



Рисунок 5.16 – Кнопка «Почати запис»

Відкриється діалогове меню задання ім'я відеофайлу з початковим каталогом, який був заданий в налаштуваннях.

Після збереження відеофайлу під заданим ім'ям кнопка запису відео стане недоступною, з'явиться таймер, що відображує час запису та відповідно зміниться рядок статусу (рис. 5.17).

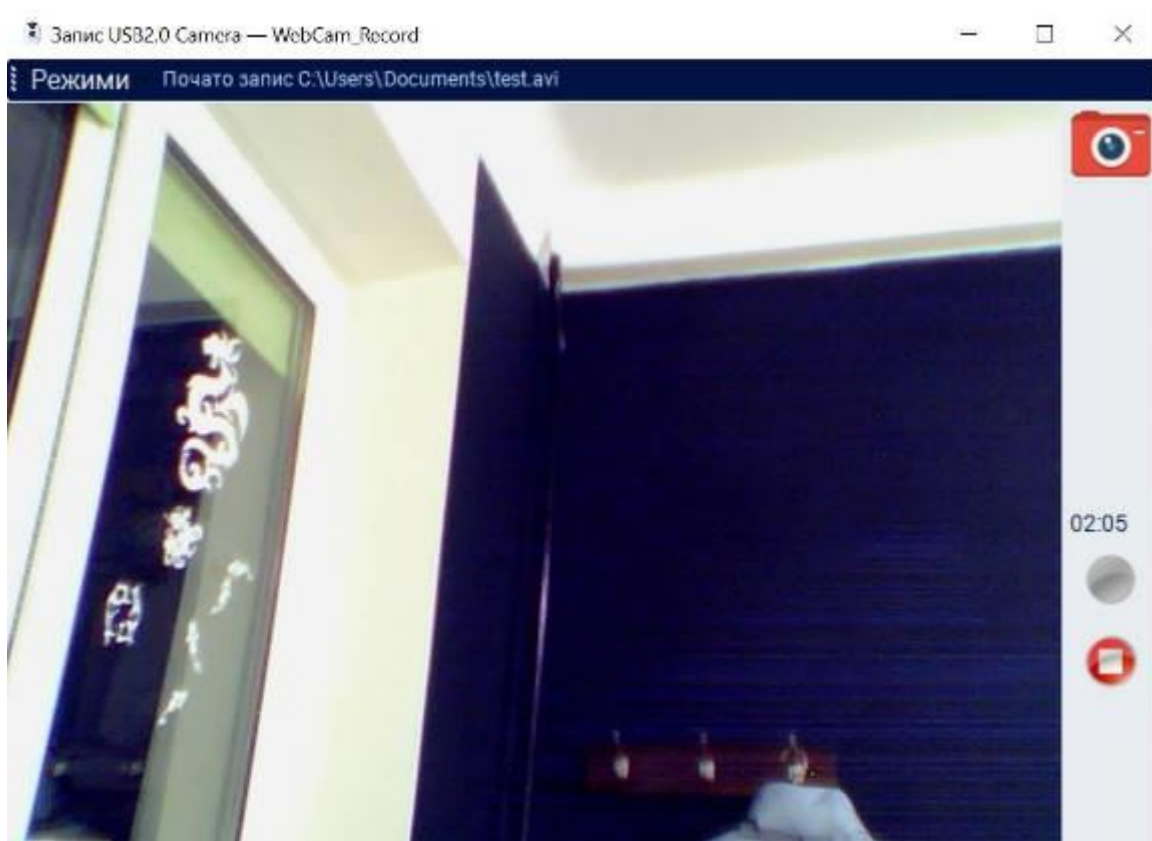


Рисунок 5.17 – Здійснення запису відео з вебкамери

Якщо розмір файлу запису буде перевищувати розмір максимального (заданий в налаштуваннях), то запис зупиниться і почнеться запис у новий відеофайл за замовчуванням. Його назва містить поточну дату, час та назву вебкамери (рис. 5.18).

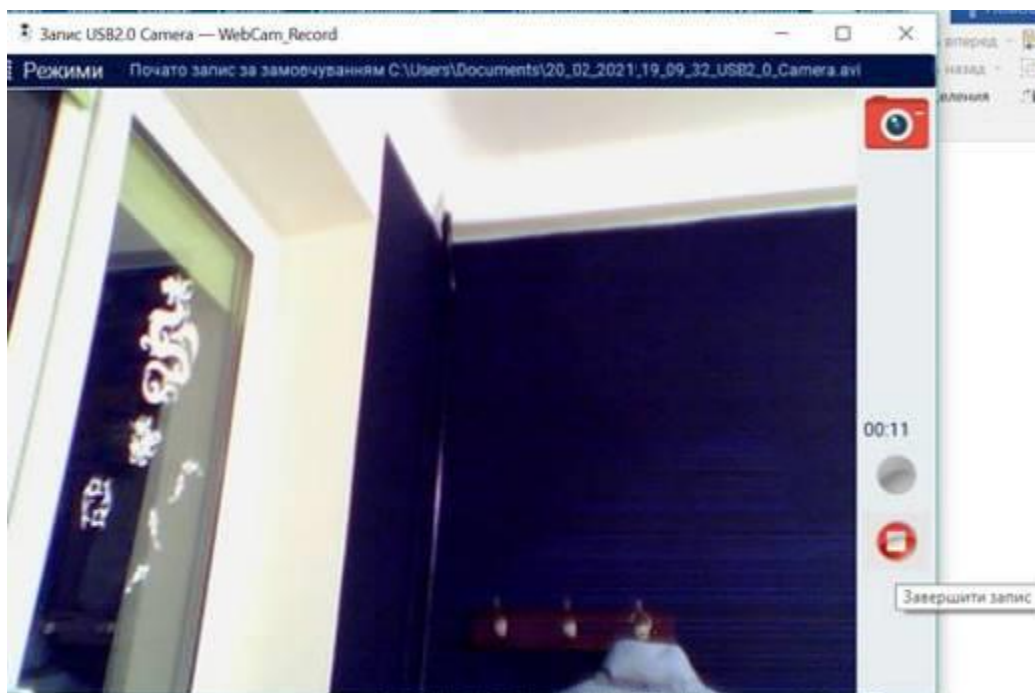


Рисунок 5.18 – Здійснення запису відео з вебкамери за замовчуванням

Завершити запис можна, натиснувши кнопку «Завершити запис».

Після завершення запису рядок статусу зміниться і буде містити шлях до збереженого відеофайлу. Також в застосунку «WebCam_Record» можна виконувати всі вище зазначені функції з декількох вебкамер, оскільки головна форма не закривається після відкриття форми запису. Можна обрати другу вебкамеру в списку і продовжувати роботу з нею, натиснувши «Перегляд онлайн» (рис. 5.19). Щоб закрити всі форми, достатньо закрити головну.

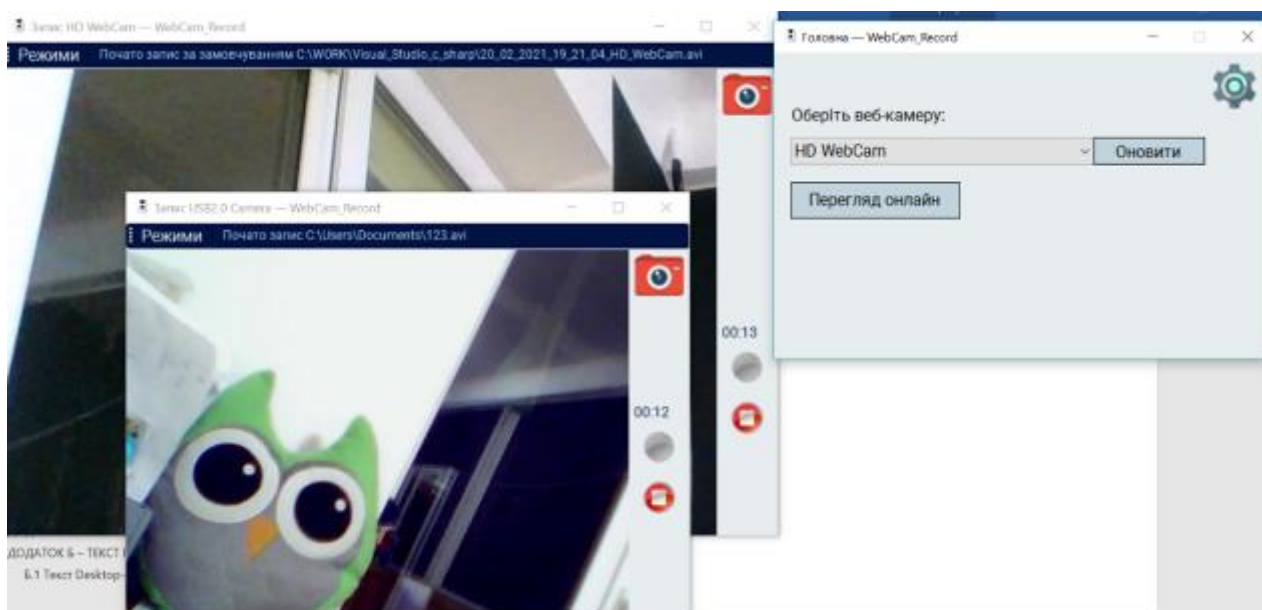


Рисунок 5.19 – Запис з декількох вебкамер: вбудованої та USB

5.4 Повідомлення оператору

Повідомлення оператору пов'язані з помилками або обмеженням вхідних даних.

До повідомлень з форми «Головна» належить повідомлення, зображене на рис. 5.20 (при цьому, здійснюється перехід на форму «Налаштування»), а також повідомлення «В системі немає наявних вебкамер», коли в системі немає вбудованих та підключених вебкамер.

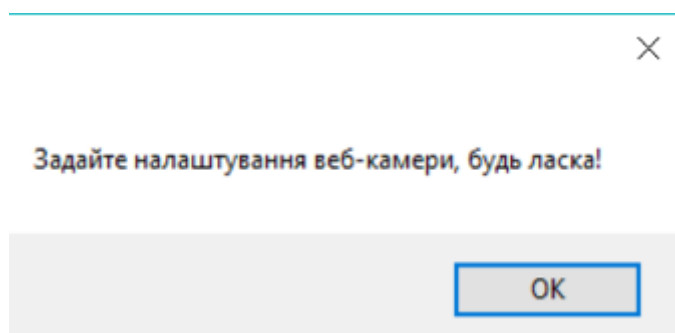


Рисунок 5.20 – Повідомлення, що виникає під час спроби переходу на форму запису без заданих налаштувань

До повідомлень з форми «Налаштування» належить повідомлення, зображені на рис. 5.21 – 5.24 які зазначають, які поля треба заповнити і якими даними. Подальшими діями є заповнення обов'язкових полів налаштування коректними даними.

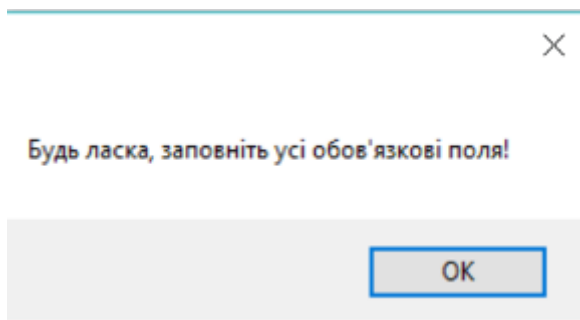


Рисунок 5.21 – Повідомлення, що виникає під час спроби збереження налаштувань без заповнення полів

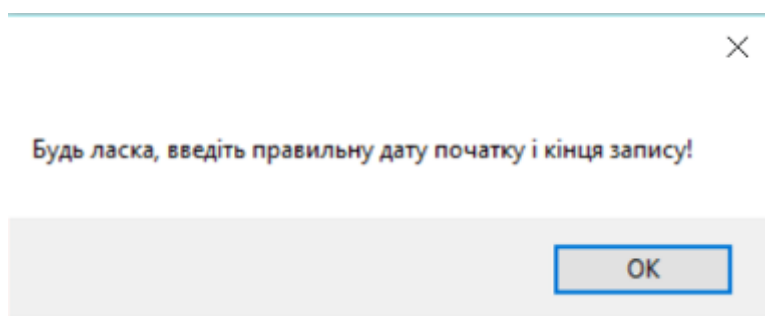


Рисунок 5.22 – Повідомлення, що виникає під час введення дати початку розкладу більшої за дату кінця

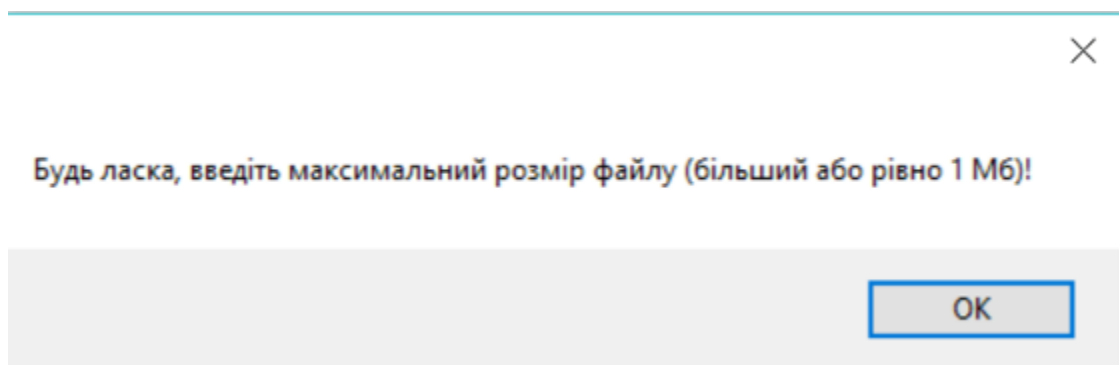


Рисунок 5.23 – Повідомлення, що виникає під час введення максимального розміру менше 1 Мб

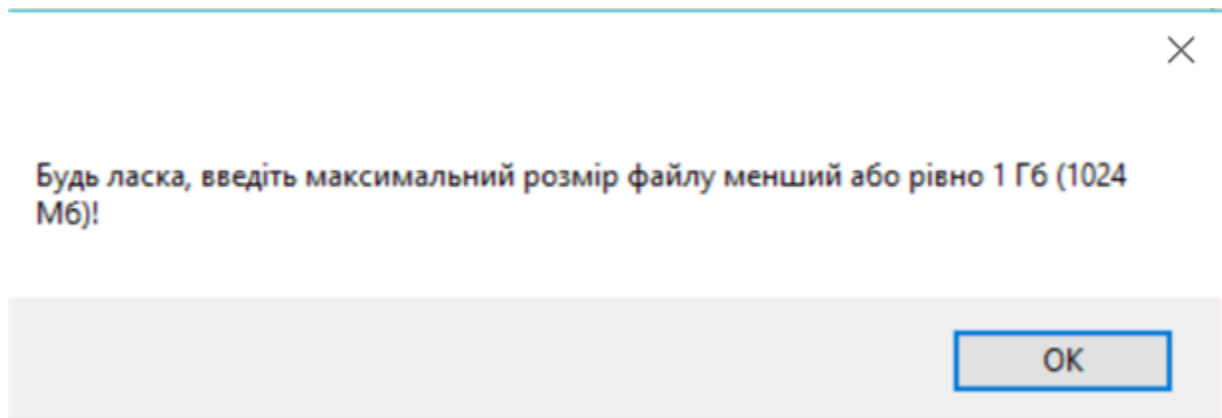


Рисунок 5.24 – Повідомлення, що виникає під час введення максимального розміру більше 1 Гб

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи бакалавра було проведено аналіз технологій розробки програмного забезпечення для ОС Windows, проведено аналіз предметної області, а також вже наявних програм-аналогів. З урахуванням результатів аналізу розроблено технічне завдання з переліком вимог до проєкту.

На етапі проєктування, для урахуванням усіх вимог, висунутих у технічному завданні, була розроблена діаграма прецедентів, що описувала послідовність дій, які виконує користувач при роботі з системою, а також форми, які забезпечують зрозумілий інтерфейс користувача. Друга модель, що була розроблена у вигляді діаграми класів, формально описує загальну структуру ієрархії класів системи.

Для реалізації такого функціоналу розробленої програми як детектування руху і облич було використано бібліотеки фреймворку Accord.NET.

Готовий програмний продукт «WebCam_Record» відповідає вимогам, поставленим у технічному завданні. Розроблене програмне забезпечення є мультифункціональним і задовольняє потреби користувачів у здійсненні відеоспостереження, запису відео, захоплення зображень з вебкамери, детектування руху і облич, ведення журналу подій.

Цей програмний продукт може використовуватись у якості найпростішої системи відеоспостереження.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Wójcik, W. Face Recognition: Issues, Methods and Alternative Applications / W. Wójcik, Gromaszek K., Junisbekov M. // Face Recognition - Semisupervised Classification, Subspace Projection and Evaluation Methods. – London: IntechOpen, 2016. – 285 p.
2. Lin, S. An introduction to Face Recognition Technology / S. Lin // Informing Science. – 2000. – Vol. 3. – P. 1-6.
3. Liao, S. A benchmark study of large-scale unconstrained face recognition / S. Liao, Z. Lei, D. Yi // Int. Joint Conf. on Biometrics (IJCIB 2014), Florida, USA, 29 September - 2 October 2014: proceedings. – Clearwater: IEEE, 2014. – P. 1–8.
4. Kurach, D. Influence of Facial Asymmetry on Human Recognition / D. Kurach, Rutkowska D. // Artificial Intelligence and Soft Computing: 11th International Conference, Zakopane, Poland, 29 April – 2 May 2012: proceedings. – Berlin: Springer, 2012. – P. 276-283.
5. Черкашин, Д. Сканер лиц в iPhone 8 будет мгновенно распознавать пользователей даже в темноте [Электрон. ресурс]. – Режим доступа: https://www.iguides.ru/main/gadgets/skaner_lits_v_iphone_8_budet_mgnovenno_raspoznavat_polzovateley_dazhe_v_temnote/.
6. Сведения о передовой технологии Face ID [Электрон. ресурс]. – Режим доступа: <https://support.apple.com/ru-ru/HT208108>.
7. Fenelly, L. J., Perry M. A. Physical Security: 150 Things You Should Know / L. J. Fenelly, M. A. Perry. – Berlin: Elsevier Inc., 2017. – 185 p.
8. Как сделать запись видео с экрана? [Электрон. ресурс]. – Режим доступа: <https://www.movavi.ru/screen-capture/>.
9. Колесниченко, Д. Как записать видео с экрана: Movavi Screen Capture Studio [Электрон. ресурс]. – Режим доступа: <https://xakep.ru/2018/06/06/movavi-screen-capture-studio/>.

10. Active Webcam [Electronic resource]. – Access mode: <https://active-webcam.ru.uptodown.com/windows>.
11. Active WebCam [Электрон. ресурс]. – Режим доступа: <https://active-webcam.ru.malavida.com/#gref>.
12. Webcam Surveyor 3.8.4 - программа для видеонаблюдения и видеозаписи [Электрон. ресурс]. – Режим доступа: <https://www.webcamsurveyor.com/ru/>.
13. Фрид, А. И. Применение библиотеки AForge.NET и ее расширения Accord.NET Framework при распознавании лиц в режиме реального времени / А. И. Фрид, С. Ф. Галеев // Молодой ученый. – 2017. – № 20 (154). – С. 85-88.
14. Найпопулярніші операційні системи світу 2020 [Електронний ресурс] – Режим доступу: <https://uaspectr.com/2020/05/06/najpopulyarnishi-operatsijni-systemy-svitu-2020/>.
15. C Sharp [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/C_Sharp.
16. Рейтинг мов програмування 2021: частка Python зменшується, а TypeScript обійшов C++ [Електронний ресурс] – Режим доступу: <https://dou.ua/lenta/articles/language-rating-jan-2021/>.
17. Reasons to Learn C# [Electronic resource]. – Access mode: <https://www.thesoftwareguild.com/blog/reasons-to-learn-c-sharp/>.
18. Psyllakis, M. Why C# Is The Best Language For The Future [Electronic resource]. – Access mode: <https://hasfocus.com/why-c-is-the-best-language-for-the-future/>.
19. Microsoft Visual Studio [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/Microsoft_Visual_Studio.
20. Основы UML — диаграммы использования (use-case) [Электрон. ресурс]. – Режим доступа: <https://pro-prof.com/archives/2594>.
21. Optimize the video bitrate for Wowza Streaming Engine [Electronic resource]. – Access mode: <https://www.wowza.com/docs/how-to-optimize-the-video-bitrate>.

22. Control.Invoke Метод [Electronic resource]. – Access mode:
[https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.control.invoke?](https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.control.invoke?view=net-5.0)
[view=net-5.0.](https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.control.invoke?view=net-5.0)

ДОДАТОК А
Текст програми

A.1 Код файла WebCamera.cs

```
namespace WebCam_Record
{
    public class WebCamera
    {
        private string name;
        private string frameSize;
        private int fps;
        private double maxFileSize;
        private string monikerString;
        private string writingPath;
        private string recordPlanFrom;
        private string recordPlanTo;
        private double bitRate;
        private bool planned;
        private bool planStopRecorded;
        private bool settingsChanged;
        private bool isRecording;
        private bool settingsComplete;
        public WebCamera()
        {
            settingsChanged = false;
            isRecording = false;
        }
        public string Name { get; set; }
        public string FrameSize { get; set; }
        public int Fps { get; set; }
        public double MaxFileSize { get; set; }
        public string WritingPath { get; set; }
        public string RecordPlanFrom { get; set; }
        public string RecordPlanTo { get; set; }
        public string MonikerString { get; set; }
        public int BitRate { get; set; }
        public bool Planned { get; set; }
        public bool PlanStopRecorded { get; set; }
        public bool SettingsChanged { get; set; }
```

```

        public bool SettingsComplete { get; set; }
        public bool IsRecording { get; set; }
    }
}

```

A.2 Код файлу INIManager.cs

```

using System.Runtime.InteropServices;
using System.Text;
namespace WebCam_Record
{
    public class INIManager
    {
        //Конструктор, який приймає шлях до INI-файлу
        public INIManager(string aPath)
        {
            path = aPath;
        }
        //Конструктор без аргументів (шлях до INI-файлу потрібно задати
окремо)

        public INIManager() : this("") { }
        //Повертає значення з INI-файлу (за вказаною секцією і ключем)
        public string GetPrivateString(string aSection, string aKey)
        {
            //Для отримання значення
            StringBuilder buffer = new StringBuilder(SIZE);
            //Отримати значення в buffer
            GetPrivateString(aSection, aKey, null, buffer, SIZE, path);
            //Повернути отримане значення
            return buffer.ToString();
        }
        //Записує значення у INI-файл (за вказаною секцією і ключем)
        public void WritePrivateString(string aSection, string aKey,
string aValue)
        {
            //Записати значення у INI-файл

```

```

        WritePrivateString(aSection, aKey, aValue, path);
    }
    //Повертає чи встановлює шлях до INI файлу
    public string Path { get { return path; } set { path = value; }
}

    //Поля класу
    private const int SIZE = 1024; //Максимальний розмір (для
читання значення з файлу)
    private string path = null; //Для зберегання шляху до INI-файлу
    //Імпорт функції GetPrivateProfileString (для читання значень) з
бібліотеки kernel32.dll
    [DllImport("kernel32.dll", EntryPoint =
"GetPrivateProfileString")]
    private static extern int GetPrivateString(string section,
string key, string def, StringBuilder buffer, int size, string path);
    //Імпорт функції WritePrivateProfileString (для запису значень)
з бібліотеки kernel32.dll
    [DllImport("kernel32.dll", EntryPoint =
"WritePrivateProfileString")]
    private static extern int WritePrivateString(string section,
string key, string str, string path);
    }
}

```

A.3 Код файлу Main.cs

```

using System;
using System.Windows.Forms;
using System.IO;
using System.Collections.Generic;
using Accord.Video.DirectShow;
namespace WebCam_Record
{
    public partial class Main : Form
    {
        private string managerPath = Directory.GetCurrentDirectory() +
@"\settingsWebCam_Record.ini";
        private bool DeviceExist;
        private bool Opened;
        private FilterInfoCollection videoDevices;
        private INIManager manager;
        private List<WebCamera> WebCameras;
        private System.Windows.Forms.Timer PlanTimer = new
System.Windows.Forms.Timer();

```

```

public Main()
{
    InitializeComponent();
}
private void Main_Load(object sender, EventArgs e)
{
    DeviceExist = false;
    Opened = true;
    btnUpdate_Click(sender, e);
    PlanTimer.Enabled = false;
    PlanTimer.Interval = 1000;
    PlanTimer.Tick += new System.EventHandler(PlanTimer_Tick);
    PlanTimer.Start();
}
private void SetWebCameras()
{
    videoDevices = new
FilterInfoCollection(FilterCategory.VideoInputDevice);
    if (videoDevices.Count != 0)
    {
        manager = new INIManager(managerPath);
        WebCameras = new List<WebCamera>();
        foreach (FilterInfo fi in videoDevices)
        {
            try
            {
                string Planned =
manager.GetPrivateString(fi.Name, "recordPlan");
                WebCamera webCamera = new WebCamera();
                WebCameras.Add(webCamera);
                webCamera.Name = fi.Name;
                if (Planned != String.Empty)
                {
                    webCamera.Planned =
Convert.ToBoolean(Planned);
                    webCamera.WritingPath =
manager.GetPrivateString(fi.Name, "writingPath");
                    webCamera.RecordPlanFrom =
manager.GetPrivateString(fi.Name, "dateTimeFrom");
                    webCamera.RecordPlanTo =
manager.GetPrivateString(fi.Name, "dateTimeTo");
                    webCamera.Fps =
Convert.ToInt32(manager.GetPrivateString(fi.Name, "fps"));
                    webCamera.FrameSize =
manager.GetPrivateString(fi.Name, "frameSize");
                    webCamera.MaxFileSize =
Convert.ToDouble(manager.GetPrivateString(fi.Name, "maxFileSize"));
                    webCamera.MonikerString = fi.MonikerString;
                    webCamera.PlanStopRecorded = false;
                    webCamera.SettingsComplete =
Convert.ToBoolean(manager.GetPrivateString(fi.Name, "settingsComplete"));
                    webCamera.BitRate =
Convert.ToInt32(manager.GetPrivateString(fi.Name, "bitRate"));
                }
            }
            catch(Exception e) { MessageBox.Show(e.ToString()); }
        }
    }
}
private void PlanTimer_Tick(object sender, EventArgs eventArgs)
{
    if (WebCameras.Count > 0)
    {

```

```

        foreach (WebCamera webCamera in WebCameras)
        {
            if (webCamera.SettingsChanged)
            {
                SetWebCameras();
                webCamera.SettingsChanged = false;
            }
            if (webCamera.Planned)
            {
                if ((webCamera.RecordPlanFrom ==
DateTime.Now.ToShortTimeString()) && Opened)
                {
                    if (!webCamera.IsRecording)
                    {
                        comboBoxDevices.Text = webCamera.Name;
                        Opened = false;
                        goToRecordingForm(true);
                    }
                }
                if (webCamera.PlanStopRecorded) Opened = true;
            }
        }
    }
    else PlanTimer.Stop();
}
private void getDevicesList()
{
    try
    {
        comboBoxDevices.Items.Clear();
        if (videoDevices.Count == 0)
            throw new ApplicationException();
        DeviceExist = true;
        foreach (FilterInfo device in videoDevices)
        {
            comboBoxDevices.Items.Add(device.Name);
        }
        comboBoxDevices.SelectedIndex = 0;
    }
    catch (ApplicationException)
    {
        DeviceExist = false;
        MessageBox.Show("В системі немає наявних веб-камер!");
    }
}
private void btnSettings_Click(object sender, EventArgs e)
{
    goToSettingsForm();
}
private void goToSettingsForm()
{
    if (DeviceExist)
    {
        int ind = comboBoxDevices.SelectedIndex;
        WebCamera webCamera = new WebCamera();
        webCamera.Name = comboBoxDevices.Text;
        webCamera.MonikerString =
videoDevices[comboBoxDevices.SelectedIndex].MonikerString;
        WebCameras.Add(webCamera);
        ind = WebCameras.Count - 1;
        Form frmSet = new Settings(this, WebCameras[ind]);
        frmSet.Left = this.Left;
        frmSet.Top = this.Top;
    }
}

```

```

        frmSet.Show();
        this.Hide();
    }
}
private void btnUpdate_Click(object sender, EventArgs e)
{
    SetWebCameras();
    getDevicesList();
}
private void btnRecording_Click(object sender, EventArgs e)
{
    goToRecordingForm(false);
}
private void goToRecordingForm(bool Planned)
{
    int ind = comboBoxDevices.SelectedIndex;
    if (WebCameras[ind].SettingsComplete)
    {
        if (!WebCameras[ind].IsRecording)
        {
            Form frmSet;
            if (Planned) frmSet = new Recording(this,
WebCameras[ind], true);
            else frmSet = new Recording(this, WebCameras[ind],
false);

            frmSet.Left = this.Left;
            frmSet.Top = this.Top;
            frmSet.Show();
            WebCameras[ind].IsRecording = true;
        }
        else MessageBox.Show("Форма з записом веб-камери " +
WebCameras[ind].Name + " вже відкрита!");
    }
    else
    {
        MessageBox.Show("Задайте налаштування веб-камери, будь
ласка!");
        goToSettingsForm();
    }
}
private void Main_FormClosing(object sender, FormClosingEventArgs
e)
{
    Application.Exit();
}
private void Main_KeyUp(object sender, KeyEventArgs e)
{
    switch (e.KeyCode.ToString())
    {
        case "F1": this.Close(); break;
        case "F3": btnUpdate_Click(sender, e); break;
        case "F4": btnSettings_Click(sender, e); break;
        case "F5": btnRecording_Click(sender, e); break;
    }
}
}
}

```

A.4 Код файлу Settings.cs

```

using System;
using System.Windows.Forms;
using System.IO;
using Accord.Video.DirectShow;
using System.Collections.Generic;
using System.Linq;
namespace WebCam_Record
{
    public partial class Settings : Form
    {
        private string managerPath = Directory.GetCurrentDirectory() +
@"./settingsWebCam_Record.ini";
        Form main;
        WebCamera Device;
        private bool SettingsComplete = false;
        public Settings(Form f, WebCamera device)
        {
            InitializeComponent();
            main = f;
            Device = device;
        }
        private void Settings_Load(object sender, EventArgs e)
        {
            this.Text += " " + Device.Name + " - WebCam_Record";
            GetFrameSizesBpp();
            comboBoxFrameSize.SelectedIndex = 0;
            comboBoxFps.SelectedIndex = 2;
            comboBoxBpp.SelectedIndex = 0;
            try
            {
                IniLoad();
            }
            catch (Exception) { }
            CalculateBitRate();
        }
        private void GetFrameSizesBpp()
        {
            VideoCaptureDevice videoCaptureDevice = new
VideoCaptureDevice(Device.MonikerString);
            List<string> tempFrameSize = new List<string>();
            List<string> tempBpp = new List<string>();
            string temp = "";
            foreach (VideoCapabilities v in
videoCaptureDevice.VideoCapabilities)
            {
                string[] words = v.ToString().Split(new char[] { ',' });
                tempFrameSize.Add(words[0]);
                temp = words[2].Replace(" bpp", "");
                tempBpp.Add(temp.Replace(" ", ""));
            }
            SortAddComboBox(tempFrameSize, comboBoxFrameSize);
            SortAddComboBox(tempBpp, comboBoxBpp);
        }
        private void SortAddComboBox(List <string> listItems, ComboBox
comboBox)
        {

```

```

        string[] arrayItems = listItems.Distinct().ToArray();
        Array.Sort(arrayItems, (x, y) => -x.CompareTo(y));
        foreach (string s in arrayItems)
        {
            comboBox.Items.Add(s.ToString());
        }
    }
    private void IniLoad()
    {
        INIManager manager = new INIManager(managerPath);
        string name = manager.GetPrivateString(Device.Name, "name");
        bool recordPlan =
Convert.ToBoolean(manager.GetPrivateString(Device.Name, "recordPlan"));
        if (recordPlan)
        {
            checkBoxRecordPlan.Checked = true;
            dateTimePickerFrom.Text = manager.GetPrivateString(Device.Name,
"dateTimeFrom");
            dateTimePickerTo.Text = manager.GetPrivateString(Device.Name,
"dateTimeTo");
        }
        textBoxMaxSizeFile.Text = manager.GetPrivateString(Device.Name,
"maxFileSize");
        textBoxWritingPath.Text = manager.GetPrivateString(Device.Name,
"writingPath");
        comboBoxFrameSize.Text = manager.GetPrivateString(Device.Name,
"frameSize");
        comboBoxFps.Text = manager.GetPrivateString(Device.Name, "fps");
        textBoxBitRate.Text = manager.GetPrivateString(Device.Name,
"bitRate");
        SettingsComplete =
Convert.ToBoolean(manager.GetPrivateString(Device.Name, "settingsComplete"));
    }
    private void btnSave_Click(object sender, EventArgs e)
    {
        if (checkFull() && validDateTime() && checkMaxFileSize())
        {
            SettingsComplete = true;
            IniSave();
            main.Show();
            Device.SettingsChanged = true;
            this.Close();
        }
        else Device.SettingsChanged = false;
    }
    private bool checkFull()
    {
        if ((textBoxMaxSizeFile.Text != String.Empty) &&
(textBoxWritingPath.Text != String.Empty)
            && (comboBoxFrameSize.Text != String.Empty) &&
(textBoxBitRate.Text != String.Empty))
        {
            return true;
        }
        else
        {

```

```

        MessageBox.Show("Будь ласка, заповніть усі обов'язкові поля!");
        return false;
    }
}
private bool validDateTime()
{
    if (Convert.ToDateTime(dateTimePickerFrom.Value) <
Convert.ToDateTime(dateTimePickerTo.Value)) return true;
    else
    {
        MessageBox.Show("Будь ласка, введіть правильну дату початку і
кінця запису!");
        return false;
    }
}
private bool checkMaxFileSize()
{
    bool check = true;
    double maxFileSize = Convert.ToDouble(textBoxMaxSizeFile.Text);
    if (maxFileSize < 1)
    {
        check = false;
        MessageBox.Show(@"Будь ласка, введіть максимальний розмір файлу
(більший або рівно 1 Мб)!");
        return check;
    }
    else if (maxFileSize > 1024)
    {
        check = false;
        MessageBox.Show(@"Будь ласка, введіть максимальний розмір файлу
менший або рівно 1 Гб (1024 Мб)!");
        return check;
    }
    else return check;
}
private void btnCancel_Click(object sender, EventArgs e)
{
    this.Close();
}
private void Settings_FormClosing(object sender, FormClosingEventArgs
e)
{
    main.Show();
}
private void IniSave()
{
    using (FileStream fileStream = new FileStream(managerPath,
FileMode.Append, FileAccess.Write)) { }
    INIManager manager = new INIManager(managerPath);
    if (!checkBoxRecordPlan.Checked)
    {
        manager.WritePrivateString(Device.Name, "recordPlan", "false");
    }
    else
    {
        manager.WritePrivateString(Device.Name, "recordPlan", "true");
    }
}

```

```

        manager.WritePrivateString(Device.Name, "dateTimeFrom",
dateTimePickerFrom.Text);
        manager.WritePrivateString(Device.Name, "dateTimeTo",
dateTimePickerTo.Text);
    }
    manager.WritePrivateString(Device.Name, "maxFileSize",
textBoxMaxSizeFile.Text);
    manager.WritePrivateString(Device.Name, "writingPath",
textBoxWritingPath.Text);
    manager.WritePrivateString(Device.Name, "frameSize",
comboBoxFrameSize.Text);
    manager.WritePrivateString(Device.Name, "fps", comboBoxFps.Text);
    manager.WritePrivateString(Device.Name, "bitRate",
textBoxBitRate.Text);
    manager.WritePrivateString(Device.Name, "settingsComplete",
SettingsComplete.ToString());
}
private void btnChooseCatalog_Click(object sender, EventArgs e)
{
    DialogResult dialogresult = folderBrowserDialog1.ShowDialog();
    folderBrowserDialog1.Description = "Обрання каталогу";
    string folderName = "";
    if (dialogresult == DialogResult.OK)
    {
        folderName = folderBrowserDialog1.SelectedPath;
    }
    textBoxWritingPath.Text = folderName;
}
private void checkBoxRecordPlan_CheckedChanged(object sender, EventArgs
e)
{
    dateTimePickerFrom.Enabled = checkBoxRecordPlan.Checked;
    dateTimePickerTo.Enabled = checkBoxRecordPlan.Checked;
}
private void textBoxMaxSizeFile_KeyPress(object sender,
KeyPressEventArgs e)
{
    //Цифри
    if ((e.KeyChar >= '0') && (e.KeyChar <= '9'))
    {
        return;
    }
    //Крапку замінюємо комою
    if (e.KeyChar == '.')
    {
        e.KeyChar = ',';
    }
    if (e.KeyChar == ',')
    {
        if (textBoxMaxSizeFile.Text.IndexOf(',') != -1)
        {
            //Вже є одна кома в textBox
            e.Handled = true;
        }
        return;
    }
}

```

```

        //Керуючі клавіші Backspace, <Enter> та інші
        if (Char.IsControl(e.KeyChar))
        {
            return;
        }
        //Інші заборонені
        e.Handled = true;
    }
    private void CalculateBitRate()
    {
        try
        {
            string[] frameSizes = comboBoxFrameSize.Text.Split(new char[] {
'e'x' });

            double fps = Convert.ToDouble(comboBoxFps.Text);
            double bpp = Convert.ToDouble(comboBoxBpp.Text) / 100;
            double BitRateKb = Convert.ToDouble(frameSizes[0]) *
Convert.ToDouble(frameSizes[1]) * fps * bpp / 1000;
            textBoxBitRate.Text = (Convert.ToInt32(BitRateKb)).ToString();
        }
        catch (Exception){}
    }
    private void comboBoxFrameSize_SelectedIndexChanged(object sender,
EventArgs e)
    {
        CalculateBitRate();
    }
    private void comboBoxFps_SelectedIndexChanged(object sender, EventArgs
e)
    {
        CalculateBitRate();
    }
    private void comboBoxBpp_SelectedIndexChanged(object sender, EventArgs
e)
    {
        CalculateBitRate();
    }
}
}

```

A.5 Код файлу Recording.cs

```

using System;
using System.IO;
using System.Drawing;
using System.Windows.Forms;
using Accord.Imaging.Filters;
using Accord.Video;
using Accord.Video.DirectShow;
using Accord.Video.FFMPEG;
using Accord.Vision.Detection;
using Accord.Vision.Detection.Cascades;
using Accord.Vision.Motion;
namespace WebCam_Record

```

```

{
    public partial class Recording : Form
    {
        private string ManagerPath = System.IO.Directory.GetCurrentDirectory()
+ @"/settingsWebCam_Record.ini";
        private string LogPath;
        private string VideoPath;
        private bool NowRecording = false, CapturePhoto = false,
FlagFaceDetection = false, FlagMotionDetection = false;
        private bool BorderMotionDetection = true;
        private bool Planned = false;
        private bool RecordingContinued = false;
        private static int RecordTimeSeconds = 0;
        private double CurFileSize = 0.0;
        private Form Fmain;
        private Bitmap Video;
        private WebCamera Device;
        private VideoCaptureDevice FinalVideo = null;
        private VideoFileWriter videoFileWriter = new VideoFileWriter();
        private RectanglesMarker FacesMarker = new RectanglesMarker(Color.Red);
        private MotionDetector BorderMotionDetector, AreaMotionDetector;
        private HaarObjectDetector Detector;
        private FaceHaarCascade faceHaarCascade;
        private FileInfo VideoFileInfo;
        private Pen YellowPen = new Pen(Brushes.Yellow);
        private Font cFont = new Font("Roboto Condensed", 10);
        private SolidBrush cSolidBrush = new SolidBrush(Color.FromArgb(254,
160, 47));
        private PointF cPointF;
        private System.Windows.Forms.Timer PlanTimer = new
System.Windows.Forms.Timer();
        private System.Windows.Forms.Timer TimerVideo = new
System.Windows.Forms.Timer();
        public Recording(Form f, WebCamera device, bool Planned)
        {
            InitializeComponent();
            Fmain = f;
            Device = device;
            if(Planned) this.Planned = Planned;
        }
        private void Recording_Load(object sender, EventArgs e)
        {
            labelStatus.Text = "";
            this.Text += " " + Device.Name + " - WebCam_Record"; ;
            LogPath = GetLogPath(Device.Name, Directory.GetCurrentDirectory(),
".log");
            btnStopRecord.Enabled = false;
            INIManager manager = new INIManager(ManagerPath);
            FinalVideo = new VideoCaptureDevice(Device.MonikerString);
            FinalVideo.NewFrame += new
NewFrameEventHandler(FinalVideo_NewFrame);
            FinalVideo.Start();
            BorderMotionDetector = new MotionDetector(new
TwoFramesDifferenceDetector(), new MotionBorderHighlighting(Color.Red));
            YellowPen.Width = 4.0F;

```

```

        AreaMotionDetector = new MotionDetector(new
TwoFramesDifferenceDetector(), new MotionAreaHighlighting(Color.Red));
        faceHaarCascade = new FaceHaarCascade();
        Detector = new HaarObjectDetector(faceHaarCascade, 30)
        {
            SearchMode = ObjectDetectorSearchMode.Average,
            ScalingMode = ObjectDetectorScalingMode.GreaterToSmaller,
            ScalingFactor = 1.5f,
            UseParallelProcessing = true,
            Suppression = 2
        };
        labelTimeRecording.Text = "";
        if (Planned)
        {
            PlanTimer.Interval = 1000;
            PlanTimer.Tick += new System.EventHandler(PlanTimer_Tick);
            PlanTimer.Start();
            RecordByDefault();
        }
        string[] FrameSizes = Device.FrameSize.Split(new char[] { 'x' });
        this.Size = new Size((Convert.ToInt32(FrameSizes[0]) +
panelBtns.Width + 16), (Convert.ToInt32(FrameSizes[1]) + toolStripModes.Height
+ 13));
        cPointF = new PointF(0.0F, toolStripModes.Height);
        TimerVideo.Interval = 1000;
        TimerVideo.Tick += new System.EventHandler(TimerVideo_Tick);
    }
    private void PlanTimer_Tick(object sender, EventArgs eventArgs)
    {
        if (Device.RecordPlanTo == DateTime.Now.ToShortTimeString())
        {
            Stop_Recording();
            PlanTimer.Stop();
            this.Close();
        }
    }
    private void TimerVideo_Tick(object sender, EventArgs e)
    {
        TimeSpan timeRecording = TimeSpan.FromSeconds(RecordTimeSeconds);
        if (RecordTimeSeconds < 3600)
        {
            labelTimeRecording.Invoke((MethodInvoker)delegate {
                labelTimeRecording.Text =
timeRecording.ToString(@"mm:ss");
            });
        }
        else
        {
            labelTimeRecording.Invoke((MethodInvoker)delegate {
                labelTimeRecording.Text =
timeRecording.ToString(@"hh:mm:ss");
            });
        }
        RecordTimeSeconds++;
    }
    private void RecordByDefault()

```

```

        {
            string datetime = DateTime.Now.ToShortDateString() + "_" +
DateTime.Now.ToLongTimeString();
            string way = GetFilePath(Device.Name, Device.WritingPath, ".avi",
datetime);
            Start_Recording(way);
        }
private void Recording_FormClosing(object sender, FormClosingEventArgs
e)
{
    TimerVideo.Stop();
    if (FinalVideo.IsRunning)
    {
        this.FinalVideo.Stop();
        videoFileWriter.Close();
    }
    Device.IsRecording = false;
}
private void btnStartRecord_Click(object sender, EventArgs e)
{
    Start_Recording("");
}
private void Start_Recording(string nameFile)
{
    string[] FrameSizes = Device.FrameSize.Split(new char[] { 'x' });
    if (nameFile == String.Empty)
    {
        SaveFileDialog saveAvi;
        saveAvi = new SaveFileDialog();
        saveAvi.Filter = "Avi Files (*.avi)|*.avi";
        saveAvi.InitialDirectory = Device.WritingPath;
        if (saveAvi.ShowDialog() ==
System.Windows.Forms.DialogResult.OK)
        {
            BtnsChangeEnabled(false, true);
            NowRecording = true;
            VideoPath = saveAvi.FileName;
            int videoBitRate = Device.BitRate * 1000;
            videoFileWriter.Open(saveAvi.FileName,
Convert.ToInt32(FrameSizes[0]), Convert.ToInt32(FrameSizes[1]), Device.Fps,
VideoCodec.Default, videoBitRate);
            WriteLog(LogPath, "Почато запис відео (" + VideoPath +
"");
            this.labelStatus.Invoke((MethodInvoker)delegate {
                this.labelStatus.Text = "Почато запис " + VideoPath;
            });
            TimerVideo.Enabled = true;
            TimerVideo.Start();
        }
    }
    else
    {
        BtnsChangeEnabled(false, true);
        NowRecording = true;
        VideoPath = nameFile;
        int videoBitRate = Convert.ToInt32(Device.BitRate * 1000);

```

```

        videoFileWriter.Open(nameFile, Convert.ToInt32(FrameSizes[0]),
Convert.ToInt32(FrameSizes[1]), Device.Fps, VideoCodec.Default, videoBitRate);
        WriteLog(LogPath, "Почато запис відео за замовчуванням " +
VideoPath);
        this.labelStatus.Invoke((MethodInvoker)delegate {
            this.labelStatus.Text = "Почато запис за замовчуванням " +
VideoPath;
        });
        TimerVideo.Enabled = true;
        TimerVideo.Start();
    }
}
private void BtnsChangeEnabled(bool checkStart, bool checkStop)
{
    this.btnStopRecord.Invoke((MethodInvoker)delegate {
        this.btnStopRecord.Enabled = checkStop;
    });
    this.btnStartRecord.Invoke((MethodInvoker)delegate {
        this.btnStartRecord.Enabled = checkStart;
    });
}
private void FinalVideo_NewFrame(object sender, NewFrameEventArgs
eventArgs)
{
    Video = (Bitmap)eventArgs.Frame.Clone();
    if (CapturePhoto)
    {
        string uniq = DateTime.Now.ToLongTimeString();
        uniq = uniq.Replace(':', '_');
        string ImgPath = Device.WritingPath + @"\" + uniq + ".bmp";
        Video.Save(ImgPath, System.Drawing.Imaging.ImageFormat.Bmp);
        this.labelStatus.Invoke((MethodInvoker) delegate {
            this.labelStatus.Text = "Захоплено зображення (" + ImgPath
+ ")";
        });
        WriteLog(LogPath, "Захоплено зображення (" + ImgPath + ")");
        CapturePhoto = false;
    }
    if (RecordingContinued)
    {
        RecordingContinued = false;
        RecordByDefault();
    }
    if (NowRecording)
    {
        try
        {
            VideoFileInfo = new System.IO.FileInfo(VideoPath);
            CurFileSize = (VideoFileInfo.Length / 1024.0 / 1024.0);
        }
        catch { }
        if (CurFileSize >= Device.MaxFileSize)
        {
            RecordingContinued = true;
            Stop_Recording();
        }
    }
}

```

```

        else
        {
            if (Video != null)
            {
                try
                {
                    videoFileWriter.WriteVideoFrame(Video);
                }
                catch (Exception) { }
            }
        }
    }
    if (FlagFaceDetection)
    {
        Bitmap bmpImage = (Bitmap)eventArgs.Frame.Clone();
        pictureBoxVideo.Image = DetectFace(bmpImage);
    }
    if (FlagMotionDetection)
    {
        Bitmap bmpFrame = (Bitmap)eventArgs.Frame.Clone();
        pictureBoxVideo.Image = DetectMotion(bmpFrame);
    }
    else if ((!FlagFaceDetection) && (!FlagMotionDetection))
    {
        pictureBoxVideo.Image = (Bitmap)eventArgs.Frame.Clone();
    }
}
private Bitmap DetectFace(Bitmap bmpImage)
{
    Rectangle[] faces = Detector.ProcessFrame(bmpImage);
    if (faces.Length != 0)
    {
        Graphics _g = Graphics.FromImage(bmpImage);
        foreach (Rectangle face in faces)
        {
            _g.DrawRectangle(YellowPen, face);
            _g.DrawString("Детектовано обличчя", cFont, cSolidBrush,
cPointF);
            _g.Dispose();
        }
        return bmpImage;
    }
}
private Bitmap DetectMotion(Bitmap bmpImage)
{
    try
    {
        double motionDetector;
        if (BorderMotionDetection)
        {
            motionDetector =
BorderMotionDetector.ProcessFrame(bmpImage);
        }
        else
        {
            motionDetector = AreaMotionDetector.ProcessFrame(bmpImage);
        }
        if (motionDetector > 0.02)

```

```

        {
            Graphics _g = Graphics.FromImage(bmpImage);
            double motion = Math.Round(motionDetector, 3);
            WriteLog(LogPath, "Зафіксовано рівень руху: " +
motion.ToString());
            _g.DrawString("Рівень руху:" + motion.ToString("{0.0,00}"),
cFont, cSolidBrush, cPointF);
            _g.Dispose();
        }
    }
    catch (Exception) { }
    return bmpImage;
}
private void btnCaptureImage_Click(object sender, EventArgs e)
{
    CapturePhoto = true;
}
private void btnStopRecord_Click(object sender, EventArgs e)
{
    Stop_Recording();
    TimerVideo.Stop();
}
private void Stop_Recording()
{
    NowRecording = false;
    BtnsChangeEnabled(true, false);
    this.labelStatus.Invoke((MethodInvoker)delegate {
        this.labelStatus.Text = "Завершено запис відео (" + VideoPath +
    ")\n";
    });
    this.labelTimeRecording.Invoke((MethodInvoker)delegate {
        WriteLog(LogPath, "Завершено запис відео (" + VideoPath + ") "
+ labelTimeRecording.Text);
        this.labelTimeRecording.Text = "";
    });
    try
    {
        videoFileWriter.Close();
    }
    catch (Exception es)
    {
        MessageBox.Show(es.Message.ToString());
    }
    if (Planned && (Device.RecordPlanFrom !=
DateTime.Now.ToShortTimeString()))
        Device.PlanStopRecorded = true;
    RecordTimeSeconds = 0;
}
private void motionDetectionToolStripMenuItem_CheckedChanged(object
sender, EventArgs e)
{
    if (motionDetectionToolStripMenuItem.Checked)
    {
        labelStatus.Text = "Увімкнено детектування руху";
        WriteLog(LogPath, "Увімкнено детектування руху");
        if (faceDetectionToolStripMenuItem.Checked)

```

```

        {
            FlagFaceDetection = false;
            faceDetectionToolStripMenuItem.Checked = false;
            borderMotionDetectionToolStripMenuItem.Checked = true;
        }
    }
    if (!borderMotionDetectionToolStripMenuItem.Checked &&
!areaMotionDetectionToolStripMenuItem.Checked)
    {
        BorderMotionDetection = true;
        borderMotionDetectionToolStripMenuItem.Checked = true;
    }
    else if (!motionDetectionToolStripMenuItem.Checked)
    {
        borderMotionDetectionToolStripMenuItem.Checked = false;
        areaMotionDetectionToolStripMenuItem.Checked = false;
        labelStatus.Text = "Вимкнено детектування руху";
        WriteLog(LogPath, "Вимкнено детектування руху");
    }
    FlagMotionDetection = motionDetectionToolStripMenuItem.Checked;
}
private void borderMotionDetectionlStripMenuItem_CheckedChanged(object
sender, EventArgs e)
{
    if (borderMotionDetectionToolStripMenuItem.Checked)
    {
        areaMotionDetectionToolStripMenuItem.Checked = false;
        BorderMotionDetection = true;
        if (!borderMotionDetectionToolStripMenuItem.Checked &&
!areaMotionDetectionToolStripMenuItem.Checked)
        {
            FlagMotionDetection = false;
            motionDetectionToolStripMenuItem.Checked = false;
        }
        else
        {
            FlagMotionDetection = true;
            motionDetectionToolStripMenuItem.Checked = true;
        }
    }
}
private void areaMotionDetectionToolStripMenuItem_CheckedChanged(object
sender, EventArgs e)
{
    if (areaMotionDetectionToolStripMenuItem.Checked)
    {
        borderMotionDetectionToolStripMenuItem.Checked = false;
        BorderMotionDetection = false;
    }
    if (!borderMotionDetectionToolStripMenuItem.Checked &&
!areaMotionDetectionToolStripMenuItem.Checked)
    {
        FlagMotionDetection = false;
        motionDetectionToolStripMenuItem.Checked = false;
    }
    else

```

```

        {
            FlagMotionDetection = true;
            motionDetectionToolStripMenuItem.Checked = true;
        }
    }
    private void faceDetectionToolStripMenuItem_CheckedChanged(object
sender, EventArgs e)
    {
        if (faceDetectionToolStripMenuItem.Checked &&
motionDetectionToolStripMenuItem.Checked)
        {
            FlagMotionDetection = false;
            motionDetectionToolStripMenuItem.Checked = false;
            faceDetectionToolStripMenuItem.Checked = true;
        }
        FlagFaceDetection = faceDetectionToolStripMenuItem.Checked;
        if (faceDetectionToolStripMenuItem.Checked)
        {
            labelStatus.Text = "Увімкнено детектування облич";
            WriteLog(LogPath, "Увімкнено детектування облич");
        }
        else
        {
            labelStatus.Text = "Вимкнено детектування облич";
            WriteLog(LogPath, "Вимкнено детектування облич");
        }
    }
    public string GetLogPath(string deviceName, string dir, string format)
    {
        deviceName = deviceName.Replace(".", "_");
        deviceName = deviceName.Replace(" ", "_");
        string path = dir + @"\\" + deviceName + format;
        return path;
    }
    public string GetFilePath(string deviceName, string dir, string format,
string text)
    {
        string path = @"\\" + text + "_" + deviceName;
        path = path.Replace(":", "_");
        path = path.Replace(".", "_");
        path = path.Replace(" ", "_");
        path = dir + path + format;
        return path;
    }
    public void WriteLog(string path, string text)
    {
        text = DateTime.Now.ToShortDateString() + " " +
DateTime.Now.ToShortTimeString() + " " + text;
        using (FileStream fileStream = new FileStream(path,
FileMode.Append, FileAccess.Write))
        using (StreamWriter streamWriter = new StreamWriter(fileStream))
        {
            streamWriter.WriteLine(text);
        }
    }
    private void Recording_KeyUp(object sender, KeyEventArgs e)

```

```

    {
        switch(e.KeyCode.ToString())
        {
            case "F1": this.Close(); break;
            case "F6": motionDetectionToolStripMenuItem.Checked =
!motionDetectionToolStripMenuItem.Checked; break;
            case "F7": borderMotionDetectionToolStripMenuItem.Checked =
!borderMotionDetectionToolStripMenuItem.Checked; break;
            case "F8": areaMotionDetectionToolStripMenuItem.Checked =
!areaMotionDetectionToolStripMenuItem.Checked; break;
            case "F9": faceDetectionToolStripMenuItem.Checked =
!faceDetectionToolStripMenuItem.Checked; break;
            case "F10": btnCaptureImage_Click(sender, e); break;
            case "F11": if(btnStartRecord.Enabled)
btnStartRecord_Click(sender, e); break;
            case "F12": if (btnStopRecord.Enabled)
btnStopRecord_Click(sender, e); break;
        }
    }
}

```

ДОДАТОК Б
Копії слайдів презентації

ДИПЛОМНА РОБОТА

РОЗРОБКА ПРОГРАМИ ВІДЕОСПОСТЕРЕЖЕННЯ ЗА ДОПОМОГОЮ ВЕБКАМЕРИ

Розробив: ст. групи КНТ-218сп
Керівник: доцент, к.т.н.

Домашенко Є.І.
Сердюк С.М.

Рисунок Б.1 – Слайд №1

Мета роботи

розробка програми, що дозволяє записувати відео з вебкамери та здійснювати відеоспостереження

Об'єкт дослідження – процес відеозапису за допомогою вебкамери

Предмет дослідження – методи відеозапису та відеоспостереження за допомогою вебкамери



Задачі, що вирішуються

- огляд технологій розпізнавання облич;
- аналіз існуючих аналогів;
- розробка технічного завдання;
- розробка моделей програмного забезпечення;
- розробка алгоритму роботи програми;
- програмна реалізація складових застосунку;
- розробка необхідної документації.



НУ "ЗП", Кафедра програмних засобів

3

Рисунок Б.3 – Слайд №3

Основні елементи типової системи розпізнавання облич



НУ "ЗП", Кафедра програмних засобів

4

Рисунок Б.4 – Слайд №4

Аналіз існуючих аналогів

№	Порівняльні характеристики	Назва програми		
		Movavi Screen Capture Studio	Active WebCam	Webcam Surveyor
1	Запис відео з вебкамери	+	+	+
2	Захоплення зображень	+	+	+
3	Детектування руху	-	+	+
4	Запис на диск	+	-	-
5	Відкладений запис	-	+	+
6	Ліцензія	Пробна	Пробна	Пробна
7	Вартість, \$	35	29	30



НУ "ЗП", Кафедра програмних засобів

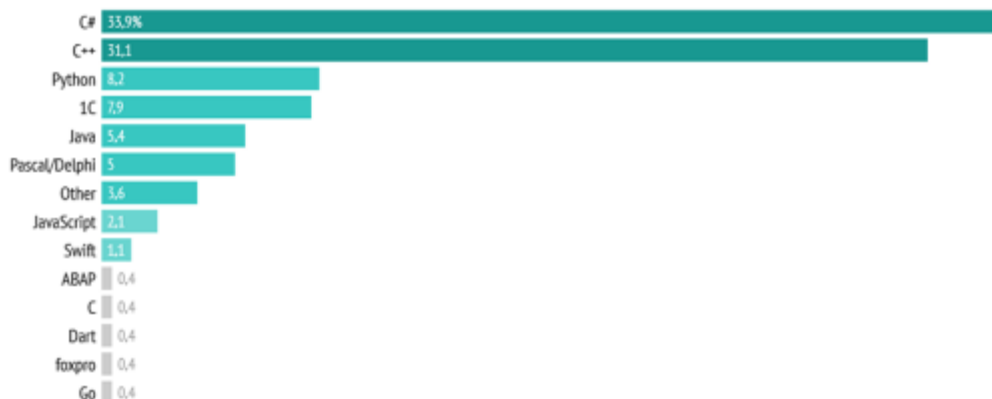
5

Рисунок Б.5 – Слайд №5

Рейтинг мов програмування

Мови програмування з розбивкою за сферами використання

[Mobile](#) [Back-end](#) [Front-end](#) [Data processing](#) [Desktop](#) [System](#) [QA automation](#) [Full Stack](#) [IoT](#) [GameDev](#)

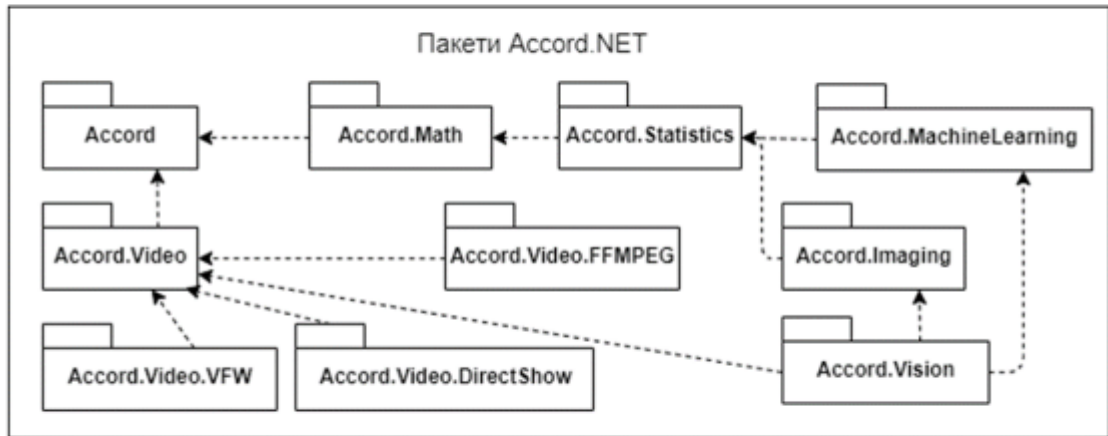


НУ "ЗП", Кафедра програмних засобів

6

Рисунок Б.6 – Слайд №6

Пакети Accord.NET використані у програмі

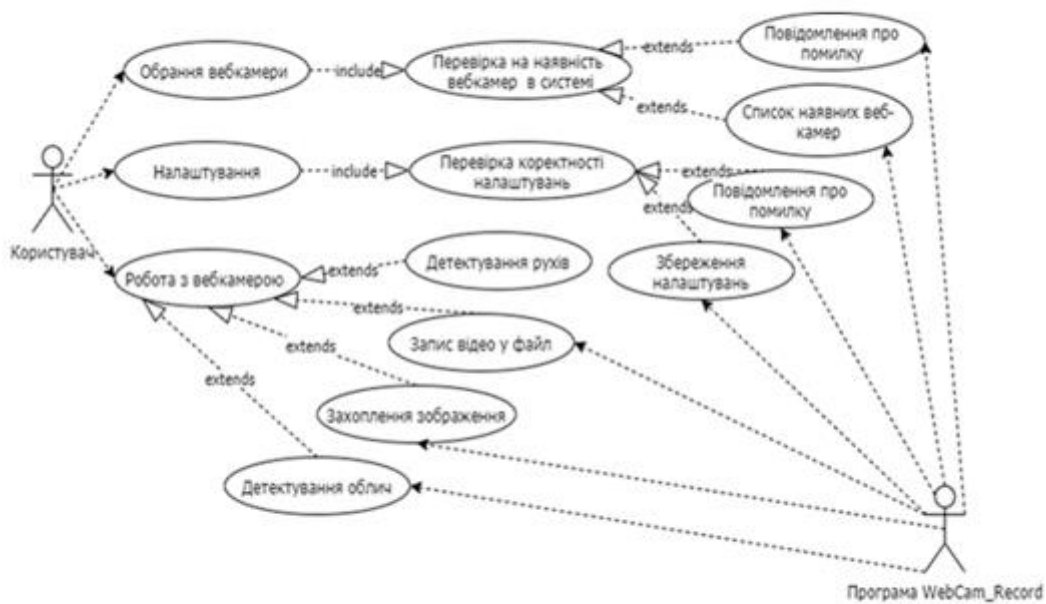


НУ "ЗП", Кафедра програмних засобів

7

Рисунок Б.7 – Слайд №7

Діаграма прецедентів



НУ "ЗП", Кафедра програмних засобів

8

Рисунок Б.8 – Слайд №8

[illegible]

Рисунок Б.9 – Слайд №9

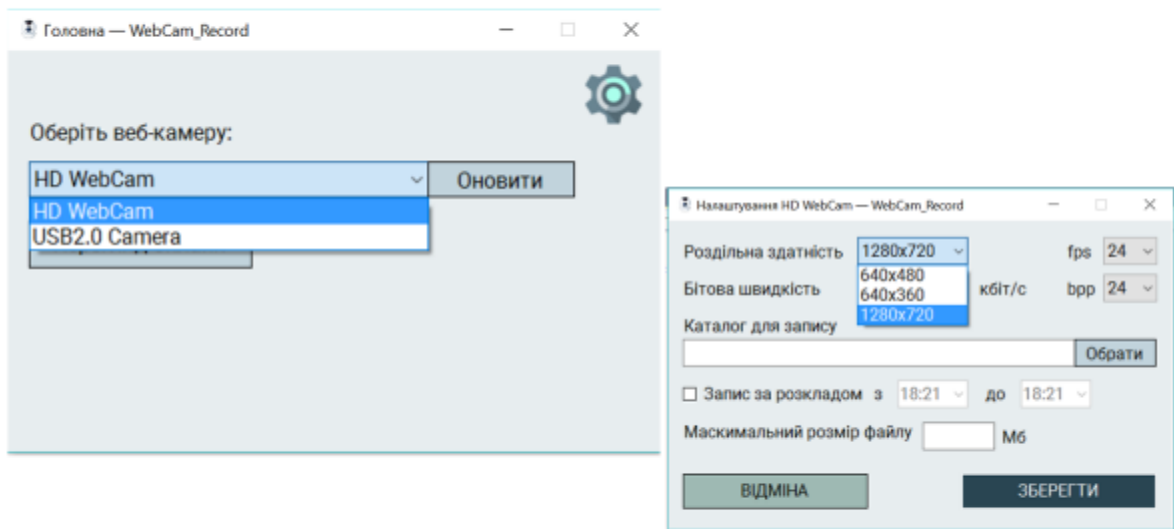
Поля класу WebCamera

Поле	Призначення
	private
string name	Назва вебкамери
string frameSize	Роздільна здатність екрану
int fps	Кількість fps
double maxFileSize	Максимальний розмір відеофайлу
string monikerString	«Прізвисько» вебкамери, найменування FilterInfoCollection
string writingPath	Шлях до каталогу за замовчуванням
string recordPlanFrom	Час початку запланованого запису
string recordPlanTo	Час кінця запланованого запису
double bitRate	Бітова швидкість відео (бітрейт)
bool planned	Ознака запланованості
bool planStopRecorded	Показує, що запис відео за розкладом зупинено
bool settingsChanged	Показує, що налаштування були змінені
bool isRecording	Показує, чи відкрита форма запису
bool settingsComplete	Показує, що всі налаштування вебкамери збережено

10

Рисунок Б.10 – Слайд №10

Обрання вебкамери та роздільної здатності

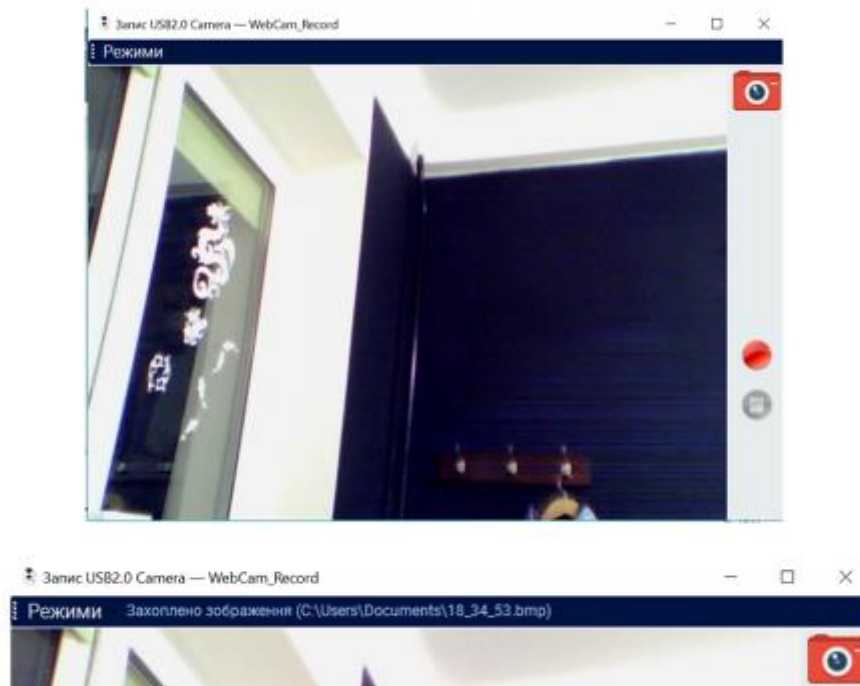


НУ "ЗП", Кафедра програмних засобів

11

Рисунок Б.11 – Слайд №11

Форма запису відео

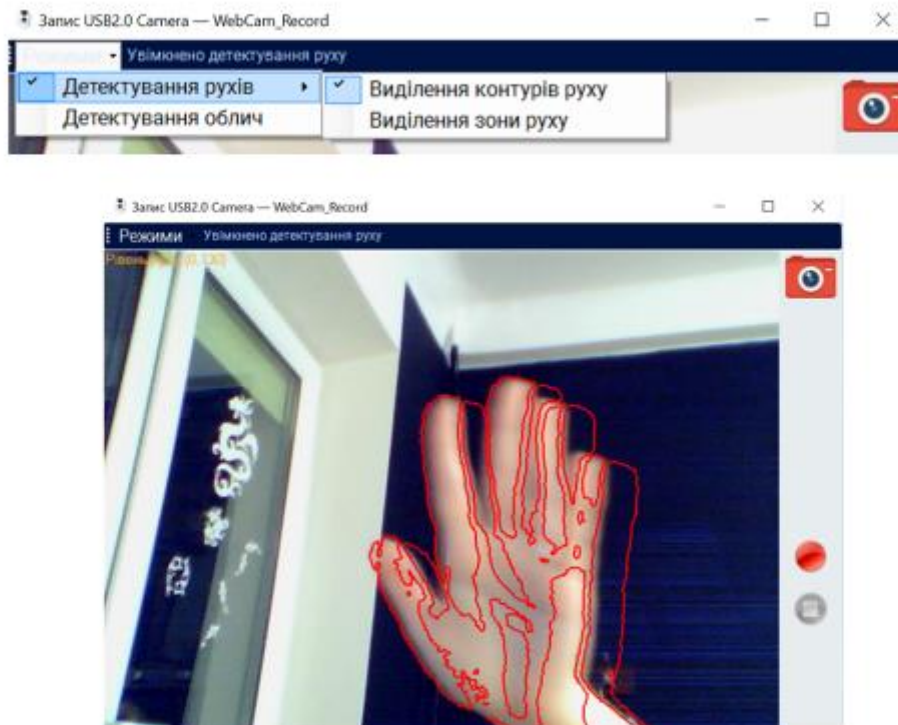


НУ "ЗП", Кафедра програмних засобів

12

Рисунок Б.12 – Слайд №12

Детектування руху

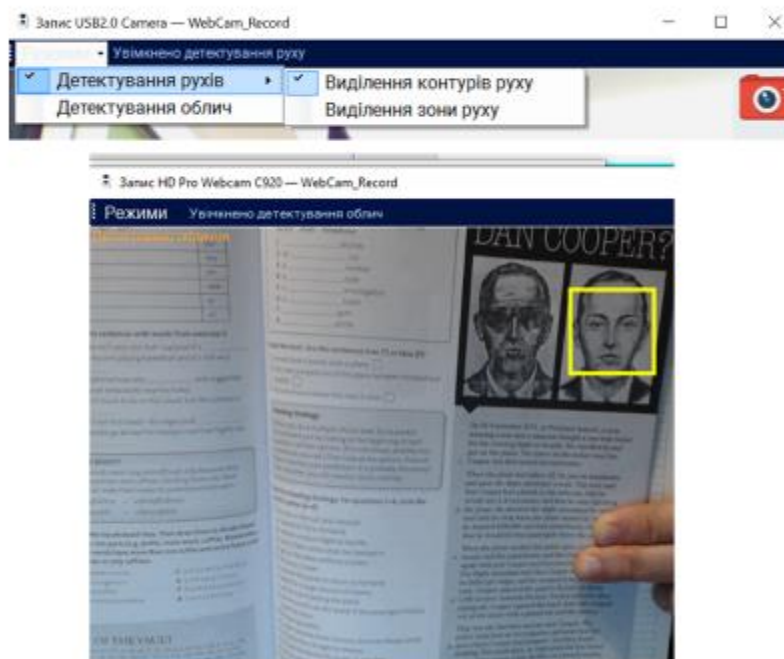


НУ "ЗП", Кафедра програмних засобів

Рисунок Б.13 – Слайд №13

13

Детектування обличчя



НУ "ЗП", Кафедра програмних засобів

Рисунок Б.14 – Слайд №14

14

Висновки

- проведено огляд технологій розпізнавання обличь;
- проведено огляд існуючих аналогів;
- розроблено технічне завдання;
- розроблено моделі програмного забезпечення;
- розроблено алгоритм роботи програми;
- програмно реалізовано усі складові частини застосунку;
- розроблено необхідну документацію.



НУ "ЗП", Кафедра програмних засобів

Рисунок Б.15 – Слайд №15

15

**Доклад закінчено,
дякую за увагу!**



НУ "ЗП", Кафедра програмних засобів

Рисунок Б.16 – Слайд №16

16