Міністерство освіти і науки України
Національний університет «Одеська політехніка»
Українсько-німецький навчально-науковий інститут
Кафедра кібербезпеки та програмного забезпечення

Іларіонова Ольга Вікторівна,
студентка групи НРЗ-181

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

Розробка алгоритму виявлення клонування кадрів у цифрових відеопослідовностях

Спеціальність:
125 Кібербезпека

Спеціалізація, освітня програма:
Кібербезпека

Керівник:
Лебедєва Олена Юріївна,
к.т.н., доцент

Одеса – 2022

Міністерство освіти і науки України
Національний університет «Одеська політехніка»
Українсько-німецький інститут
Кафедра кібербезпеки та програмного забезпечення
Рівень вищої освіти перший (бакалаврський)
Спеціальність 125 – Кібербезпека
Освітня програма – Кібербезпека

ЗАТВЕРДЖУЮ
Завідувач кафедри КБПЗ

_____
д.т.н.,проф. А.А.Кобозєва
____ _____202_р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Іларіоновій Ользі Вікторівні

1.Тема роботи: *Розробка алгоритму виявлення клонування кадрів у цифрових відеопослідовностях*,

керівник роботи *Лебедєва Олена Юріївна, к. ф.-м. н., доцент,*

затверджені наказом ректора від „17” 05. 2022р. №168-в .

2.Зміст роботи*: аналіз проблемної області, постановка задачі, розробка алгоритму виявлення клонування кадрів у цифрових відеопослідовностях, програмна реалізація, проведення експериментів, охорона праці*.

3. Перелік ілюстративного матеріалу: *скріншоти інтерфейсу програми, блок-схема алгоритму програмного продукту, слайди презентації*.

4. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
Охорона праці	Ярова І.А.	Завдання видав	Завдання прийняв
		9.06.2022	14.06.2022

5. Дата видачі завдання 9.06.2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання	Примітка
1	*Аналіз джерел з теми випускної кваліфікаційної роботи*	*20.11.2021*	*виконано*
2	*Розробка алгоритму виявлення фальсифікацій в цифрових відеопослідовностях*	*25-01-2022*	*виконано*
3	*Розроблення програмного комплексу*	*28-03-2022*	*виконано*
4	*Написання розділу з охорони праці*	*14-06-2022*	*виконано*
5	*Підготовка тексту роботи*	*15-06-2022*	*виконано*
6	*Підготовка презентації та доповіді*	*01-06-2022*	*виконано*
7	*Попередній захист*	*03-06-2022*	*виконано*
8	*Нормоконтроль, рецензування*	*20-06-2022*	*виконано*

Здобувач вищої освіти _____ *Іларіонова О.В.*

Керівник роботи _____ *Лебедєва О.Ю.*

ЗАВДАННЯ

на розробку розділу «Охорона праці» у кваліфікаційній роботі бакалавра

студентці *Іларіоновій Ользі Вікторівні*

Інститут Українсько-німецький навчально-науковий інститут

Кафедра Кафедра кібербезпеки та програмного забезпечення

Дата отримання завдання 09.06.2022

Консультації 09.06.2022, 14.06.2022

Дата закінчення розділу 14.06.2022

Тема роботи «Розробка алгоритму виявлення клонування кадрів у цифрових відеопослідовностях»

Зміст розділу

1. Аналіз умов праці і вибір основних заходів виробничої безпеки
2. Аналіз пожежної безпеки і вибір заходів і засобів пожежної безпеки

Керівник дипломної роботи

_____ Лебедєва О.Ю.
(підпис)

« __ » _____2022 р.

Консультант з охорони праці

_____ Ярова І. А.
(підпис)

« __ » _____ 2022 р.

АНОТАЦІЯ

Кваліфікаційна робота на тему «Розробка алгоритму виявлення клонування кадрів у цифрових відеопослідовностях» на здобуття першого (бакалаврського) рівня вищої освіти за спеціальністю 125 − Кібербезпека, спеціалізація, освітня програма: Кібербезпека, містить 9 рисунків, 3 таблиці, 1 додаток, 26 літературних джерел за переліком посилань. Робота виконана на 60 сторінках загального тексту і 44 сторінках основного тексту.

Метою даної роботи є розробка алгоритму виявлення відеопослідовностей, що були сфальсифіковані, шляхом пошуку клонованих кадрів.

У цій роботі пропонується знаходити дубльовані кадри, порівнюючи послідовно кожен кадр досліджуваного відео з кожним. В алгоритмі використовується перетворення зображення з колірного простору RGB на YUV, для аналізу використовується компонента Y, тому що вона зберігає всю потрібну для досліджень інформацію про зображення. Для порівняння значень компонент Y відеокадрів використовуються різні метрики оцінки якості, такі як коефіцієнт кореляції Пірсона, MSE, NMSE, SNR, AAD, Image Fidelity. Дослідження проводилися на сфальсифікованих заздалегідь відео, які в процесі фальшування не піддавались стисканню. Була проведена оцінка точності виявлення як поодиноких сфальсифікованих кадрів, так і їх послідовностей, а також порівняно час, за який програма опрацьовувала відео за допомогою різних метрик.

ДУБЛЮВАННЯ ВІДЕОКАДРІВ, ВИЯВЛЕННЯ ФАЛЬСИФІКАЦІЙ У ВІДЕОФАЙЛАХ, МЕТРИКИ ОЦІНКИ ЯКОСТІ ЗОБРАЖЕННЯ, КОЛІРНИЙ ПРОСТІР YUV

ANNOTATION

Qualification work entitled as «Development of an algorithm for detecting frame cloning in digital video sequences» to obtain the first (bachelor's) level of higher education in the specialty 125 – Cybersecurity, specialization, educational program: Cybersecurity, has 9 pictures, 2 tables, 1 appendix, 26 literature sources according to the list of references. The work is performed on 59 pages of general text and 43 pages of main text.

The purpose of this work is to develop an algorithm for detecting video sequences that have been falsified by searching for cloned frames.

In this paper, it is proposed to find duplicate frames, comparing each frame of the studied video with each. The algorithm uses the conversion of the image from the RGB color space to YUV, the Y component is used for analysis, because it stores all the necessary information for research about the image. To compare the values of the Y components of video frames, various quality assessment metrics are used, such as Pearson's correlation coefficient, MSE, NMSE, SNR, AAD, Image Fidelity. The research was conducted on falsified videos in advance, which were not compressed during the falsification process. The accuracy of detection of both single counterfeit frames and their sequences was evaluated, as well as a comparison of the time for which the program processed video using different metrics.

DUPLICATION OF VIDEOS, DETECTION OF FALSIFICATIONS IN VIDEO FILES, METRICS OF EVALUATION QUALITY ASSESSMENT, COLOR SPACE YUV

ЗМІСТ

ВСТУП

Розвиток технологій та цифровізації в сучасному світі йде швидкими темпами, багато сфер життя людини оцифровуються і переводяться в смартфон або комп'ютер: таксі, доставка, сімейні архіви фотографій. Фото- та відеоматеріали вже складають звичну та необхідну нам частину оточення. Ми відсилаємо один одному фотографії з вихідних, дивимося відео на відеохостингах в автобусі по дорозі на роботу, увечері дивимося новини по телевізору, здійснюємо відеодзвінки, навчаємось за допомогою відеоуроків. Будь-який смартфон може знімати, обробляти та зберігати фото- та відеоконтент, не кажучи вже про персональний комп'ютер.

Ми настільки звикли споживати та виробляти візуальний контент, так міцно увійшла в наше життя цифровізація, що багато складних сфер людського життя тепер нерозривно пов'язані з роботою з фото- та відеофайлами, такі як, наприклад, криміналістика або засоби масової інформації. Скрізь розташовані камери відеоспостереження, майже кожен у кишені має смартфон. Того чи іншого виду цифрові докази є в більшості випадків злочинів. Відомо багато випадків справ, розкритих лише за допомогою записів з камер відеоспостереження. Новини також поширюються дуже швидко, і часом їхнє джерело втрачається в інформаційному шумі.

З часом винаходяться і розповсюджуються, стають масово доступними та більш дружелюбними до користувача не тільки нові інструменти зйомки та зберігання фото- та відеоконтенту, але також інструменти модифікації отриманої інформації. Не потрібно мати спеціальну освіту або вузькопрофільне обладнання, що фальсифікувати зображення або відео, вистачить персонального комп’ютера та певної кількості часу, витраченого на вивчення будь-якого графічного редактора. Можливості реалізувати різноманітні фальсифікації дуже високої якості, зміни в яких непомітні ні оку звичайної людини, ні експерта, з кожним днем розширюються.

У криміналістиці дуже важлива справжність поданих доказів, щоб суд був дійсно справедливим, щоб виходячи з підроблених доказів, зробивши неправильні висновки, не посадити за ґрати невинного і не відпустити волю винного. Також підробки відеофайлів можуть використовуватись для шантажу.

В інформаційному просторі, який зараз є як ніколи відкритим і масштабним, неправдиві новини або фейки, засновані на сфальшованих відеоматеріалах, здатні розлетітися дуже швидко, а слід їхнього творця може навіки загубитися. У багатьох людей немає ні часу, ні бажання довго думати над тим, чи достовірна інформація, яку вони щойно прочитали на сайті новин або каналі; вони приймають на віру те, що першим побачать і що не виглядає як зовсім відверта брехня. Однак інформація, яку могли приховати або додати відео, здатна навіть змінити його сенс на протилежний. І вже точно фальсифікація надовго відкладеться в головах якщо не всіх, то багатьох людей, які побачили підроблену новину, і може сильно вплинути на їхню думку та рішення надалі.

Для запобігання таким неприємностям потрібно постійно створювати та розвивати методи та інструменти, що дозволяють аналізувати відеофайли та виявляти в них фальшування за наявності. Тому тема даної роботи є актуальною.

Метою даної роботи є розробка алгоритму виявлення відеопослідовностей, що були сфальсифіковані, шляхом пошуку клонованих кадрів.

Для досягнення даної мети були поставлені наступні задачі:

- виконати огляд сучасного стану проблеми фальсифікації відео;
- розробити алгоритм виявлення фальсифікації у відеопослідовності;
- реалізувати програмними засобами розроблений алгоритм виявлення фальсифікації у відео послідовності та протестувати з різними метриками.

Об’єктом дослідження є фальсифікація відеопослідовностей.

Предметом дослідження є методи виявлення фальсифікації у відеопослідовності.

Практична цінність роботи полягає у розробці алгоритму виявлення фальсифікації у відеопослідовності, та його програмна реалізація, яка може бути використана з виявлення фальсифікації у відео.

1 ОГЛЯД СУЧАСНОГО СТАНУ ПРОБЛЕМИ ФАЛЬСИФІКАЦІЇ ВІДЕО

1.1 Огляд можливих фальсифікацій у відео

Нещодавно, завдяки доступності недорогих та легких мультимедійних пристроїв, цифрова технологія зазнала величезного вдосконалення. Підробка відео стала набагато простішою і, як наслідок, будь-який медіа продукт тепер повинен підлягати перевірки.

Усі існуючі методи фальсифікації відео можна розділити на категорії, виходячи зі схожості алгоритмів:

- видалення частини відеопослідовності або заміна її на інший елемент або уривок відео, з цього ж відео або з іншого, або на окреме зображення або кілька;
- редагування або обрізання кадру відео;
- включення побічної інформації у кадр, зокрема віртуальних об'єктів;
- анімація статичних об'єктів;
- накладання побічних зображень або частин на вихідне відео.

Розроблені програмні засоби оцінки фото- та відеоматеріалів здатні виявити:

- різкі зрушення окремих елементів;
- відмінності у освітленні, розподілі світла та тіні на сусідніх кадрах;
- відмінності у відтінках кольору, зернистості, оптичної густини, насиченості на подібних або сусідніх зображеннях;
- свідчення застосування графічних редакторів;
- диспропорцію окремих елементів, відсутність композиційної єдності;
- впроваджені штучні заповнення окремих віддалених елементів [1].

В [2] також описуються випадки фальсифікації способом міжвідеозміни, яке видаляє частини відео, наприклад об'єкти або людей.

Недоліком цих методів є неминуча наявність похибок.

Також всі підробки відео можна розділити на міжкадрові та внутрішньокадрові [3]. Міжкадрові підробки – це будь-які зміни, внесені до

послідовності кадрів, наприклад, видалення або вставка кадру або набору кадрів із відеопослідовності або до неї. Сюди можна віднести дублювання або клонування кадрів, а також склейку, коли кадри з двох або більше відео інтерполюються для створення нового відео. При внутрішньокадрових підробках змінюється вміст окремих кадрів, наприклад, копіювання та вставка окремих фрагментів, кадрування тощо.

1.2 Огляд методів фальсифікації відео

Високий відсотком виявлення міжкадрових підробок має метод, описаний у [4]. Він призначений для виявлення фальсифікацій у записах із камер відеоспостереження. Фрагменти відеозаписів з камер відеоспостереження часто вважаються більш переконливими доказами, що перевіряються в суді, ніж статичні зображення, однак вони так само легко піддаються підробкам. Описаний метод отримує двовимірну фазову конгруентність кадру і знаходить кореляцію між сусідніми кадрами. Алгоритм складається з виділення ознак та локалізації аномальних точок з використанням алгоритму кластеризації на другому етапі. Недолік методу в тому, що при випадку видалення кадру на початку або в кінці відео застосування методу виявлення неможливо. Також алгоритм краще виявляє вставку кадру, ніж видалення кадру.

Цифрова CCD камера вносить шуми у кожен кадр, який вона записує, і який виявляється під час зчитування. Було з'ясовано, що ці шуми в послідовності кадрів слідують деякому шаблону. Було припущено [5], що у виявленні підробок може допомогти будь-яка знайдена зміна у цьому шаблоні. Для виявлення підробки автори пропонують вирахувати різницю між середнім рівнем шуму для всіх кадрів відео та шумом у конкретному кадрі. Кадри з високим відхиленням від середнього значення вважаються сфальшованими. Також використовувалося обчислення кореляції між шумом у підблоці кадру та загальним шумом кадру, і будь-які невідповідності давали можливість припускати, що кадр міг бути підроблений. У метода є недоліки: неможливість роботи зі стислими відео та

використання жорсткої системи порогів, які визначаються емпіричним шляхом, а, отже, у них відсутня гнучкість.

У [6] описаний підхід, що використовує дактилоскопію. Для початку за допомогою концепції SPN (Sensor Pattern Noise) визначається, чи всі кадри даного відео були записані однією камерою. Автори вважають, що кожна відеокамера має унікальну шумову картину, ці структури виявлялися на початку відеоряду і використовувалися для знаходження підробок в іншій частині відео. Кореляція між SPN досліджуваного кадру та еталонним SPN розраховується та порівнюється з емпіричним порогом для виявлення вставки та реплікації кадру, а також вставки об'єкта у кадр. Метод працює надійно у випадку з несжатими відео, але показує погану ефективність, коли потрібно досліджувати вже стиснуті відео. У джерелі стверджується, що цей метод надійний для відео, кадри якого зазнали інтерполяції, але не був визнаний дуже точним, якщо SPN містив високочастотні компоненти.

Інший підхід, заснований також на використанні властивостей камери, був запропонований в [7]. Метод забезпечує аутентифікацію цифрового відео на рівні пікселів, а також здатний локалізувати будь-які міжкадрові підробки. Підозрілими областями у досліджуваному відео вважаються кадри, записані за допомогою іншої камери та вставлені в бажане місце у заданому відео. Вони виявляються за допомогою використання невідповідностей у фотонному дробовому шумі, які вносяться різними пристроями збору даних. Метод тестували на нестиснутих відеороликах, записаних у приміщенні та на вулиці, внаслідок чого було виявлено 97% підроблених пікселів. Однак, як і в минулому методі, зі стиснутими відео ефективність перевірки різко знижується, у випадку з окремими кодеками іноді до 6% виявлених фальшивих пікселів. Сучасні кодеки видаляють більшу частину шумових характеристик з відео, і продуктивність програми падає зі зменшенням кількості шуму в тестовому відео, тому до стисненого відео метод або не застосовується, або застосовується з обережністю. Також цей метод застосовується лише до відео зі статичними сценами, наприклад, записаним стаціонарними камерами спостереження. Крім того, оскільки метод був

протестований на невеликій кількості тестових послідовностей, його широкомасштабна корисність не може бути визначена з упевненістю.

Підхід, запропонований в [8], заснований на технології моделювання руху між сусідніми кадрами відеопослідовності за допомогою використання марківських моделей. Ознаки руху було вилучено шляхом моделювання руху між сусідніми кадрами відеопослідовності з використанням марківських моделей. У центрі кожного кадру розташовувалося невелике вікно, і до всіх кадрів усередині цього вікна застосовувалася функція усереднення захопленної інформації про рух між цими кадрами. Залишок руху був отриманий з різниці між передбачуваним та фактичним рухом, і до цього залишку було застосовано марківську модель. Середня точність виявлення становить близько 87%, проте ефективність методу не була підтверджена.

В [9] був запропонований метод виявлення міжкадрової підробки, заснований на виявленні невідповідностей оптичного потоку, що виникають в результаті додавання/видалення кадрів. Автори спромоглися виявити сильно неузгоджені значення оптичних потоків в результаті своїх досліджень. Вони використовували невелике вікно, що рухається, розраховували оптичний потік між першим і останнім кадрами всередині вікна, щоб виявити вставку кадрів, і між кожною парою сусідніх кадрів, щоб виявити видалення кадрів. Для оцінки продуктивності вони створили одну тестову базу даних відео з використанням сценаріїв TRECVID Content Based Copy Detection (CBCD), дослідження на якій показали точність 95,3%, і дві бази даних з використанням бібліотеки функцій OpenCV, точність виявлення підробок у яких склала 97,9%. Ефективність падала зі зменшенням кількості кадрів, що вставляються, так само як і видалених кадрів. Чим менше кадрів було видалено або вставлено, тим менш ефективним був метод. Також на ефективність могла негативно вплинути відсутність значного або помітного руху на відео. Крім того, розрахунок значень оптичного потоку є дуже складним та ресурсомістким процесом. Також він схильний до помилок, що може поставити під загрозу роботу всього методу.

Ще одна схема виявлення міжкадрової підробки була запропонована в [10]. Використовується узгодженість коефіцієнтів кореляції значень сірого (CCCoGV) як криміналістична ознака. За основу береться твердження, що хоча для вихідних відео CCCoGV залишається постійним, будь-які порушення поствиробництва в послідовності кадрів змушують це значення демонструвати аномалії. Для остаточної класифікації використовується SVM. Для кількісного аналізу свого підходу автори використовували п'ять наборів даних — один, що містить оригінальні відео, і чотири, що містять підробки — кожен із яких складався з 598 відео з нерухомим фоном і невеликим рухом камери. Повідомляється, що середня точність виявлення становить 99% (для виявлення вставки кадру) та 95% (для виявлення видалення кадру). Провівши базовий аналіз функціональності методу, можна дійти невтішного висновку, що сильне стиснення так само, як і у вищеописаних методах, призведе до значного зниження точності виявлення підробок.

У [11] було запропоновано підхід виявлення дублювання кадрів, досліджуючи аномалії на рівні пікселів. Автори запропонували процес «від грубого до точного» для визначення подібності між заданим кліпом-запитом і кліпом-кандидатом, щоб визначити наявність кліпів, що повторюються, і локалізувати їх положення. Першим кроком було обчислення у часовій області різниці у гістограмах послідовних кадрів у колірному просторі RGB. Потім були розраховані фактичні просторові кореляції між відповідними кадрами кліпу-запиту та кліпу-кандидата, щоб переконатися, що ці кадри справді ідентичні, а не просто помилки, спричинені шумом. Експерименти проводились на чотирьох тестових відео. Метод не викликав помилкових спрацьовувань, але в одному відео не всі підробки виявились. Цей метод також заснований на жорстких порогах, що емпірично визначаються, що обмежує сферу його застосування. Крім того, було виявлено, що цей метод не зможе виявити будь-яку підробку, якщо кадри будуть перемішані перед вставкою в інше місце у відео.

Метод виявлення інтерполяції, запропонований в [12], працює на тому принципі, що кожного разу, коли зловмисник намагався виконати інтерполяцію

кадрів, одночасно намагаючись мінімізувати тимчасові артефакти, що виникають, це робилося за допомогою інтерполяції з компенсацією руху. Проте інтерполяція з компенсацією руху залишила характерні та цілком помітні сліди. Автори змогли запропонувати систему, яка працювала для стиснених і злегка стиснених, наприклад, за допомогою H.264, відео, таких як телевізійні трансляції, і давала багатообіцяючі результати навіть при використанні лише на підмножині кадрів. Точні кількісні результати не повідомлялися, але автори визнали, що продуктивність їхньої системи була незадовільною для відео, закодованого у форматі MPEG-2, оскільки таке стиснення було набагато більшим, ніж кодування H.264/AVC. Більш того, система добре функціонувала на просторових вікнах невеликого розміру, що дозволило використовувати цей детектор як можливий інструмент для виявлення атак підробки копіпасти. Однак кількість спостережуваних інтерпольованих кадрів має бути достатньо великою, щоб система могла успішно виявляти підробки.

У [13] автори помітили, що більшість методів підвищення частоти кадрів вносять візуально помітні періодичні артефакти в текстурні області порушених кадрів. Це спостереження призвело авторів розробки двоетапного методу сліпого виявлення, заснованого на аналізі лише на рівні кадру функції, названої середньої варіацією текстури (ATV). На першому етапі методу обчислювалися значення ATV кожного кадру, щоб отримати криву ATV відео-кандидата. На другому етапі кожна крива ATV додатково оброблялася для виявлення періодичних артефактів. Цей метод може локалізувати становище інтерпольованих кадрів, і навіть допомогти оцінити вихідну частоту кадрів відео. Повідомляється, що середня точність виявлення становить 96%.

У [14] описані методи та класифікації, які використовують у криміналістиці для встановлення справжності цифрових доказів. Відповідно до [15], сьогодні 80-90% випадків злочинів мають той чи інший вид цифрових доказів. У [16] описані різні способи ідентифікації цифрових камер, з яких був зроблений знімок або відеозйомка, за допомогою дослідження особливості стиснення JPEG, масиву колірних фільтрів, дефектів сенсора і так далі. Більшість публікацій покладається

на SPN — шаблонний шум сенсора, так як окремі пікселі виявляють різну чутливість до світла через неоднорідність кремнієвих пластин, проте цей метод має недоліки, такі, наприклад, як забруднення SPN деталями сцен.

Типи датчиків, що використовуються у цифрових фотоапаратах та відеокамерах, аналогічні. Ось чому в цій галузі можуть бути застосовані підходи, що враховують патерн шуму датчика. Велика кількість кадрів у відеоданих призводить до кращих результатів для ідентифікації джерела в порівнянні з ідентифікацією вихідної цифрової камери. У [17] показали, що успішна ідентифікація можлива навіть після завантаження відео з різними параметрами якості та невідомими параметрами на платформу YouTube.

У [18] описаний метод перевірки цілісності відеофайлу, що використовує структуру відеоконтенту в реєстраторі даних відеоподій (VEDR). Більшість систем відеоспостереження не забезпечені належними вбудованими функціями захисту цілісності. Автори вважають, що систематичні підходи до судової перевірки цілісності необхідні встановлення істини чи хибності. використовує структуру відеоконтенту у реєстраторі даних відеоподій (VEDR). Пропонований метод визначає різницю в полях індексу кадру між підробленим файлом та вихідним файлом. Експерименти, проведені з використанням реальних VEDR на ринку та відеофайлів, підроблених за допомогою інструмента для редагування відео, показують, що запропонована схема перевірки цілісності може виявляти порушення цілісності відеоконтенту.

Вузькоспеціалізованих методів визначення різноманітних фальсифікацій у відео є велика кількість. На жаль, більша частина з них має різноманітні недоліки: не працює або погано працює зі стислими відео, вимагає величезної кількості ресурсів. Деякі мають відносно великий відсоток помилки.

Як ми можемо бачити, область виявлення фальсифікацій у відео потребує додаткових досліджень та експериментів.

2 РОЗРОБКА АЛГОРИТМУ ВИЯВЛЕННЯ КЛОНУВАННЯ КАДРІВ У ЦИФРОВИХ ВІДЕОПОСЛІДОВНОСТЯХ

2.1 Поняття та характеристики відео

З моменту появи в 1992 році технологій, що дозволяють оцифровувати аналогові відео, сфери, пов'язані зі створенням та обробкою цифрових відео, значно розвинулися. З'явилися як нові неймовірні можливості, які в тому ж 1992 важко було навіть уявити, так і нові проблеми. Чим складніша система, тим більше вона має вразливостей.

Розвиток та розповсюдження інструментів, заточених на модифікацію відео, доступність їх для кожного, хто володіє спеціальним устаткуванням, сприяло появі різних фальсифікацій. Якість та різноманітність цих інструментів здатні забезпечити високу якість самої підробки та ускладнити її виявлення. Часто буває, що без спеціальних інструментів не можна відрізнити фальшивку від оригіналу, або навіть знайти «вхідні точки» модифікацій.

Відео – образотворча, візуальна інформація, що включає нерухливі зображення та зображення, що змінюються в часі. Нерухливе зображення є просторовим розподілом інтенсивності, постійним в часі. Ознакою зображення, що змінюється у часі є те, що просторова картина інтенсивності змінюється з часом. Іншим часто використовуваним терміном для відео є послідовність зображень [20].

Цифрове відео – це візуальний або аудіовізуальний контент, що пройшов аналогово-цифрове перетворення і представлений у вигляді бітової послідовності.

Цифрове відео має основні характеристики.

Частота кадрів відображає кількість кадрів, яка входить у секунду відео. Найчастіше використовується частота 30 кадрів за секунду. У кіновиробництві прийнято за стандарт частоту 24 кадри в секунду.

Колірна роздільна здатність або глибина кольору – це кількість кольорів, представлена в бітах, які можуть бути задіяні у формуванні кадру відео.

Роздільна здатність екрана – це кількість пікселів по вертикалі та горизонталі відеокадрів, з яких складається відео.

Ширина відеопотоку або бітрейт – це кількість бітів відеоінформації, яка обробляється за секунду часу.

Якість відеозображення – сукупність значень попередніх характеристик, тобто глибини кольору, частоти кадрів, екранної роздільної здатності та бітрейту, які використовуються для аналізу та оцінки якості обробленого відео в порівнянні з оригіналом.

Для того, щоб зменшити обсяг пам'яті, який займає відео, та полегшити його передачу каналами передачі інформації, застосовуються алгоритми стиснення відео. Аудіо- та відеоінформація в несжатому вигляді займала б величезну кількість місця на носії інформації, тому в більшості випадків аудіо та відео зберігаються в стислому вигляді. До популярних стандартів стиснення відео відносяться DV, MPEG-2, VCD, MPEG-4. Загалом їх існує кілька десятків.

Вибравши один алгоритм стиснення, можна скористатися різними інструментами та отримати різні результати. Кодек – це програма, що реалізує кодування оригінального відеоматеріалу, яка включає не тільки можливість кодування, але і можливість декодування вихідної інформації. Застосування цих програм і тягне за собою підсумкові відмінності в звуку та якості картинки, а також поширенні та частоті похибок. Відповідно, існують різні кодеки, найпопулярнішими є MPEG-4 Part 2 ASP, MPEG-4 AVC, VC-1.

Також для зберігання відео необхідний медіаконтейнер, який визначатиме, який кодек використовувати, скільки аудіодоріжок та каналів субтитрів може бути у відеофайлі, з якими медіаплеєрами він сумісний тощо.

Існує кілька найпоширеніших медіаконтейнерів:

- AVI – підтримує поєднання різних кодеків, однак його можна вважати застарілим, оскільки він не підтримує безліч нових функцій, такі як, наприклад, позначки часу або можливість зберігати альтернативні аудіодоріжки;

- MKV – сучасний формат, що може зберігати безліч потоків аудіо, відео та субтитрів, також забезпечує швидку навігацію по файлу за допомогою можливості реалізації екранного меню та поділу вмісту файлу на розділи, а також швидкого перемикання між доріжками відео, аудіо та субтитрів; проте він сприймається невеликою кількістю пристроїв, переважно лише персональними комп'ютерами;
- MP4 – має практично ті самі можливості, що й MKV, але підходить для відтворення майже на всіх пристроях та цифрових платформах, і має обмеження за кількістю кодеків та форматів звуку. Забезпечує високу якість відео при відносно невеликих розмірах;
- MOV – формат відеофайлів, розроблений Apple, розроблений для підтримки програвача QuickTime. Файли MOV містять відео, аудіо, субтитри, часові коди та інші типи мультимедіа. Це відеоформат дуже високої якості, файли займають велику кількість місця у пам'яті носія;
- WMV – розроблений Microsoft і широко використовується у медіаплеєрах Windows. Формат WMV забезпечує невеликі розміри файлів із кращим стисненням, ніж MP4.

Відео складається з відеокадрів, тобто із зображень.

2.2 Поняття зображення. Колірні простори RGB та YUV

Цифрове зображення – це відображення реального зображення у вигляді набору чисел, який може зберігатися та оброблятися комп'ютером. Щоб перевести зображення в цифровий формат, спочатку воно ділиться на області, які називаються пікселями, опис певних властивостей кожної області, таких, як яскравість або колір, обробний пристрій перетворює на число чи деяку кількість чисел. Числа потім зберігаються в масиві рядків і стовпців, що відповідає вертикальному та горизонтальному розташуванню пікселів у зображенні.

Зображення можна визначити як двовимірну функцію f(x, y), де x і y – координати в просторі (конкретно на площині) і значення f якої у будь-якій точці,

що задається парою координат (x, y), називається інтенсивністю або рівнем сірого, або ж рівнем яскравості зображення у цій точці [21].

Цифрові зображення мають ряд основних характеристик: роздільна здатність, тип, колірний простір.

Роздільна здатність зображення – це величина, яка означає, скільки пікселів вміщується у дюймі зображення.

Зображення можуть бути векторними та растровими. Векторне зображення подається у вигляді геометричних примітивів, растрове – за допомогою пікселів, у яких зберігається інформація про колір. Растрові зображення поділяються на:

- бінарні, коли пікселі можуть зберігати тільки значення «0», що відповідає за чорний колір, або «1», що відповідає за білий колір;
- напівтонові, які складаються з пікселів, кожен з яких набуває будь-якого значення інтенсивності одного тону;
- палітрові, у яких значення кожного пікселя є посиланням на палітру – двомірний масив, у якому розташовані значення інтенсивності одного кольору;
- повнокольорові, у пікселях яких безпосередньо зберігається інформація про яскравість кольору.

Колірні простори можна уявити як математичні діаграми основних кольорів, які у поєднанні дають будь-яку кількість кольорів з різною насиченістю і яскравістю. Колірні простори необхідні для створення барвистих зображень та відео.

Колірні простори RGB та YUV є найбільш широко використовуваними.

Ідея колірного простору RGB заснована на концепції, що оскільки людське око має кольорочутливі рецептори тільки для червоного, синього та зеленого кольорів, теоретично можна розкласти кожен колір на ці три «основні» кольори. Сучасні дисплеї здатні відображати мільйони кольорів, використовуючи та комбінуючи лише ці три кольори. Для позначення глибини кольору використовуються значення від 0 до 255, якщо всі три колірні канали мають нульове значення, то результуючий колір вийде чорним, якщо 255 - білим.

З RGB працюють всі екрани, цей колірний простір є найвідомішим і найпоширенішим, проте він не оперує окремо такими поняттями, як, наприклад, насиченість, яскравість та колірний тон.

Колірний простір YUV переважно використовується в аналоговому телебаченні. Колірний простір YUV був особливо корисним, коли колір вперше з'явився на телебаченні, оскільки він міг декодувати і відображати як кольорове, так і чорно-біле зображення, і допомагав передавати кольорове зображення каналами, створеними для монохромних сигналів. Система YUV, як і будь-який інший колірний простір, є системою математичного кодування, і використовує математичні формули для представлення кольорів, але зберігає в собі як яскравість, так і колір.

YUV найбільш корисний для створення зображень у градаціях сірого, оскільки все, що потрібно зробити користувачеві, це позбутися елементів U і V рівняння.

До кожного компоненту простору YUV потрібно ставитися по-різному, тому що вони виконують дуже різні функції. Компонента Y зберігає інформацію про яскравість, у той час як в U і V зберігається інформація про синій та червоний колірний компонент відповідно.

Для перетворення компонентів RGB в компоненти YUV існують формули:

$$Y = 0.299R + 0.587G + 0.114B$$

$$U = 0.492\ (B-Y)$$

$$V = 0.877\ (R-Y)$$

Також їх можна представити як:

$$Y = 0.299R + 0.587G + 0.114B$$

$$U = -0.147R - 0.289G + 0.436B$$

$$V = 0.615R - 0.515G - 0.100B$$

Цифрове зображення може зберігатися у файлах різних форматів.

JPEG(JPG) – один із найпопулярніших форматів, зокрема для цифрових фотокамер. Це формат стиснення з втратами, тобто якість зображення знижується зі зменшенням розміру файлу, хоча зазвичай це майже непомітно. Він має гнучку

систему стиснення, що дозволяє контролювати ступінь стиснення, можна як зберегти зображення в максимальній якості, так і зменшити до мінімального розміру для передачі по мережі, якщо використовується програма для редагування зображень. Файли мають розширення .jpg або .jpeg;

TIFF – файли зберігаються без стиснення і без втрат в якості, хоча є варіанти і зі стисненням, використовується у сферах, де важлива висока якість зображення. Файли мають розширення .tif, .tiff;

PNG – зберігає дані у стислому вигляді, але без втрат. Спочатку розроблений для покращення або заміни формату GIF. PNG підтримує наявність альфа-каналу, що дозволяє створювати ефект прозорості у будь-якій частині зображення. Файли мають розширення .png;

BMP – у файлах цього формату відсутнє стиснення, а значить, і втрата інформації. Це дозволяє зображенням мати дуже високу якість, але є недолік - дуже великий розмір файлів. Файли мають розширення .bmp;

GIF – файли цього формату мають обмеження в 256 кольорів, підтримують наявність альфа-каналу та можуть бути анімовані. Зазвичай, мають невеликий розмір. Використовується стиск без втрати якості. Файли мають розширення .gif

EPS – поширений формат векторних зображень, збереження відбувається без стиснення. Файли мають розширення .eps;

RAW – необроблені зображення, створені камерою або сканером. Зображення у форматі RAW – це еквівалент цифрового негативу, вони містять багато інформації про зображення, але їх все одно потрібно обробляти в якомусь редакторі. Файли мають розширення .raw, .nef, .orf, .cr2, .sr2 та інші.

2.3 Метрики оцінки якості

Для пошуку подібних блоків в алгоритмах можуть використовувати метрики оцінки якості зображення. Вони були створені для того, щоб оцінювати вплив на зображення різних спотворень, отриманих внаслідок стиснення або обробки, таких як шум, артефакти, розмиття тощо. Часто оцінка, отримана за допомогою такої метрики, корелює із суб'єктивним сприйняттям людини, також

вони допомагають обчислювати непомітні оку або первинній перевірці помилки. Метрики оцінки якості можуть бути використані для порівняння зображень або для порівняння алгоритмів обробки зображень.У цій роботі для знаходження дубльованих кадрів використовувалися такі метрики: коефіцієнт кореляції Пірсона, MSE, NMSE, Average Absolute Difference, Signal to Noise Ratio, Image Fidelity.

Коефіцієнт кореляції Пірсона використовується для виявлення взаємозв'язку між двома змінними, визначає, чи пропорційна їх мінливість, коли при зміні одного показника змінюється і другий. Кореляція Пірсона є лінійною. У математичній статистиці значення коефіцієнта кореляції Пірсона може бути від +1 до -1, де +1 свідчить про наявність повного позитивного лінійного зв'язку, а -1 — про наявність повного негативного лінійного зв'язку. При повному вигляді значення коефіцієнта кореляції Пірсона дорівнює 1.

$$\frac{\sum(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum(X_i - \bar{X})^2 \sum(Y_i - \bar{Y})^2}}$$

MSE (Mean Square Error) — середньоквадратична помилка, для всіх точок даних вимірює середню суму квадратної різниці між фактичним значенням та прогнозованим значенням. Оскільки виконується зведення у другу ступінь, негативні значення не нівелюються позитивними, і вплив помилок посилюється. Чим менше значення MSE, тим вище подібність. При повному вигляді значення MSE дорівнює 0.

$$\frac{1}{MN}\sum_{m,n}(X_{m,n} - Y_{m,n})^2$$

NMSE – це MSE, нормалізована за потужністю сигналу. При повному подобі значення NMSE також дорівнює 0.

$$\frac{\sum_{m,n}(X_{m,n} - Y_{m,n})^2}{\sum_{m,n} X_{m,n}^2}$$

Average Absolute Difference – це середнє абсолютне відхилення від центральної точки, зведена статистика статистичної дисперсії чи мінливості. При повному вигляді значення AAD дорівнює 0.

$$\frac{1}{MN}\sum_{m,n}\left|X_{m,n}-Y_{m,n}\right|$$

SNR (Signal to Noise Ratio) – за допомогою цієї метрики порівнюється рівень корисного сигналу з рівнем фонового шуму. При повній подобі значення SNR прагне +∞.

$$\frac{\sum_{m,n}X_{m,n}^2}{\sum_{m,n}(X_{m,n}-Y_{m,n})^2}$$

Image Fidelity – це міра визначення точності зображення, тобто здатність процесу точно відображати зображення без будь-яких видимих спотворень та втрати інформації. При повному вигляді значення Image Fidelity дорівнює 1.

$$1-\frac{\sum_{m,n}(X_{m,n}-Y_{m,n})^2}{\sum_{m,n}X_{m,n}^2}$$

2.4 Алгоритм виявлення фальсифікації у відеопослідовності

Будемо вважати, що маємо відеопослідовність $P=\{p_1,p_2,\dots,p_s\}$ розміру $M\times N$. Алгоритм виявлення фальсифікації у відеопослідовності складається з наступних основних кроків:

Крок 1. Зчитується послідовність кадрів $P=\{p_1,p_2,\dots,p_s\}$ розміру $M\times N$. Отримати матриці R^k, G^k, B^k для кожного кадру, k = 1,…,s.

Крок 2. Отримати матриці Y^k, U^k, V^k для кожного кадру, k = 1,…,s.

Крок 3. Зчитується p_i кадр, i = 1,…,s. Для кожного кадру p_i виконуємо наступні кроки:

Крок 3.1 Зчитується p_j кадр, j = 1,…,s, $i\neq j$. Для кожної пари кадрів p_i та p_j вирахувати метрику подібності блоків $metric=SimilarityMetric(p_i,p_j)$.

Крок 3.2 Якщо $metric\geq\delta$, то кадри p_i та p_j являються оригінальним та клонованим, запам’ятовуються їх номера, $res=res\cup\ i\ \cup\ j$. Інакше розглянути наступну пару кадрів.

Крок 4. Вивести знайдені номери кадрів res.

В якості метрик подібності в роботі використовувались: коефіцієнт кореляції Пірсона, MSE, NMSE, Average Absolute Difference, Signal to Noise Ratio, Image Fidelity.

Значення порогового коефіцієнту δ визначається в результаті експериментів.

3 ОПИС ПРОГРАМНОГО ПРОДУКТУ

3.1 Відомості про мову програмування C#

Програмний продукт, про який йде мова в нашій роботі, був написаний мовою програмування C#.

C# – це об'єктно-орієнтована мова програмування, розроблена Microsoft на початку 2000-х років під керівництвом Андерса Хейлсберга. Є частиною платформи .Net і призначений для використання як проста мова програмування загального призначення. Його можна використовувати для розробки різних типів програм, включаючи консольні, віконні, веб-програми та мобільні програми [22].

Компілятор C# компілює вихідний код, вказаний у вигляді набору файлів з розширенням .cs, у збірку, яка є одиницею упаковки та розгортання у .NET. Збірка може бути як додатком, так і бібліотекою, з тією різницею, що програма має точку входу – метод «Main», а бібліотека – ні. Бібліотека призначена для виклику або посилання програмою або іншими бібліотеками. Середовище виконання .NET Core (і .NET Framework) складається із набору бібліотек [23].

Без інструментів, що оперують даними, від програми ніякої користі. У C# існують різні типи даних, такі як bool, integer, string, double, decimal тощо [24]. Типи даних мають фіксований розмір, що спрощує роботу. C# підтримує інкапсуляцію, успадкування, поліморфізм та перевантаження операторів та багато чого ще[25]. У C# реалізований механізм автоматичного збору сміття, що дозволяє здебільшого не прописувати спеціальні інструменти під це завдання.

C# розроблений на основі мов програмування Java, C++, Basic, від яких він багато успадкував і з якими має багато чого спільного, що полегшує його вивчення при знанні будь-якої з мов-«батьків». Основною перевагою C# є універсальність і кросплатформеність, на ньому можна розробити додаток практично під будь-який пристрій, хоча пріоритетно він все ж таки орієнтований на платформу Windows. Також плюсом є величезна кількість бібліотек, заготовок, шаблонів, що дозволяють не винаходити те, що вже тисячі разів придумано, кожного разу кожному програмісту, а також, що важливо, безкоштовність і

доступність більшості цих інструментів. Перевагами мови C#, крім іншого, є хороша читаність коду, висока швидкість розробки та велике ком'юніті. Він підтримує абсолютну більшість продуктів Microsoft. C# постійно займає високі позиції у рейтингах популярності, його використовують для розробки у багатьох великих компаніях та невеликих проектах, він стабільно розвивається.

Крім переваг, C#, зрозуміло, має і недоліки. Таким є, наприклад, недостатня захищеність коду від дизассемблізації. Також у деяких програмістів виникають проблеми з автоматичним складанням сміття, яке може діяти некоректно. Також у деяких джерелах називається недоліком називається деяка перевантаженість C# синтаксичними конструкціями, які можуть не дозволяти використовувати багато рішень найефективнішим чином.

У роботі використовуються дві бібліотеки C# для роботи з відео: Accord.Video.FFMPEG та OpenCvSharp. Accord.Video.FFMPEG належить до Accord.NET, що забезпечує статистичний аналіз, машинне навчання, обробку зображень та методи комп'ютерного зору для програм .NET. Колись вона була розширенням колишньої платформи AForge.NET Framework, а потім розширилася, включила Aforge.NET і доповнила її новими функціями, створивши більш повне середовище для наукових обчислень в .NET [26].

OpenCvSharp відноситься до OpenCV, яка є бібліотекою з відкритим вихідним кодом, що включає кілька сотень алгоритмів комп'ютерного зору і має модульну структуру.

Accord.Video.FFMPEG та OpenCvSharp містять синтаксичні конструкції, що дозволяють працювати з відеопослідовностями, читати та витягувати кадри для подальшої роботи з ними, зберігати та перезаписувати відео.

3.2 Опис програмного продукту

Щоб проаналізувати відео, слід відкрити файл videoframes.exe. З'являється вікно такого вигляду (рисунок 3.1):

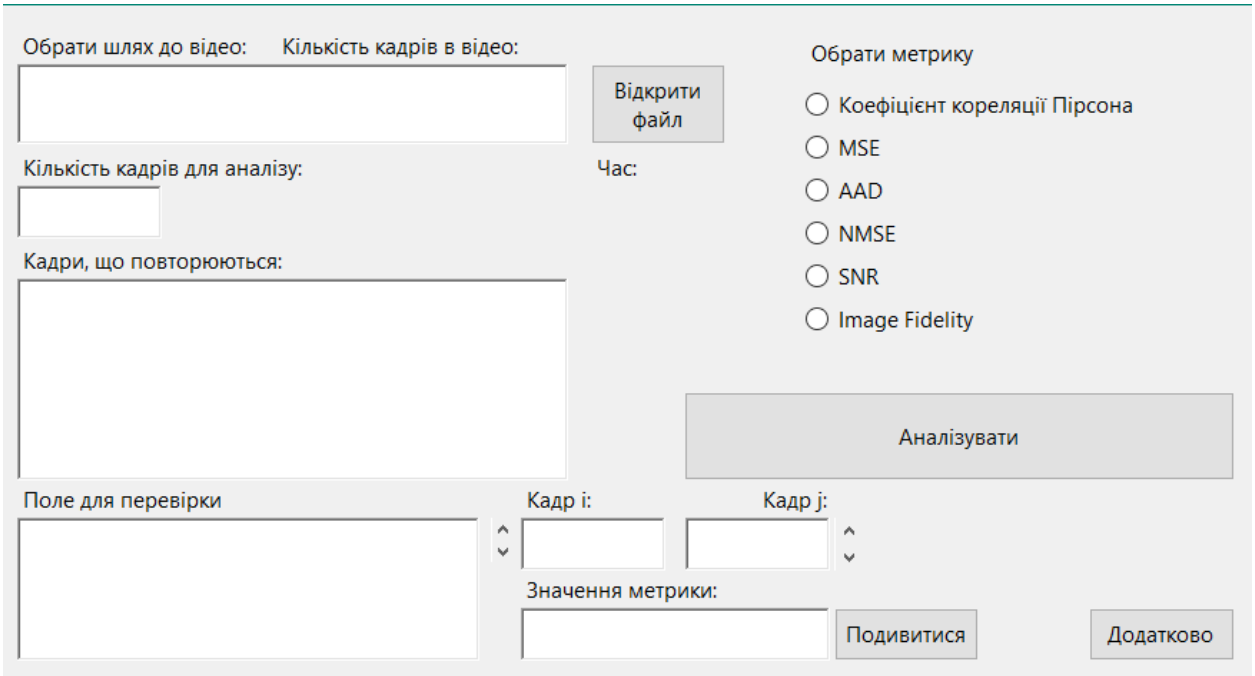

Рисунок 3.1 – Початок роботи з програмою

Потрібно вибрати шлях до файлу, натиснувши кнопку «Відкрити файл». Програма порахує кількість кадрів, що можна побачити над полем вибору шляху до файлу у рядку «Кількість кадрів». Наприклад, у даному випадку їх 17. У полі «Кількість кадрів для аналізу» потрібно вписати кількість кадрів, яку потрібно проаналізувати, якщо стоїть завдання проаналізувати все відео, то вписати потрібну повну кількість. У списку поруч користувач вибирає бажану метрику оцінки якості. Аналіз розпочнеться із натисканням кнопки «Аналізувати» (рисунок 3.2):

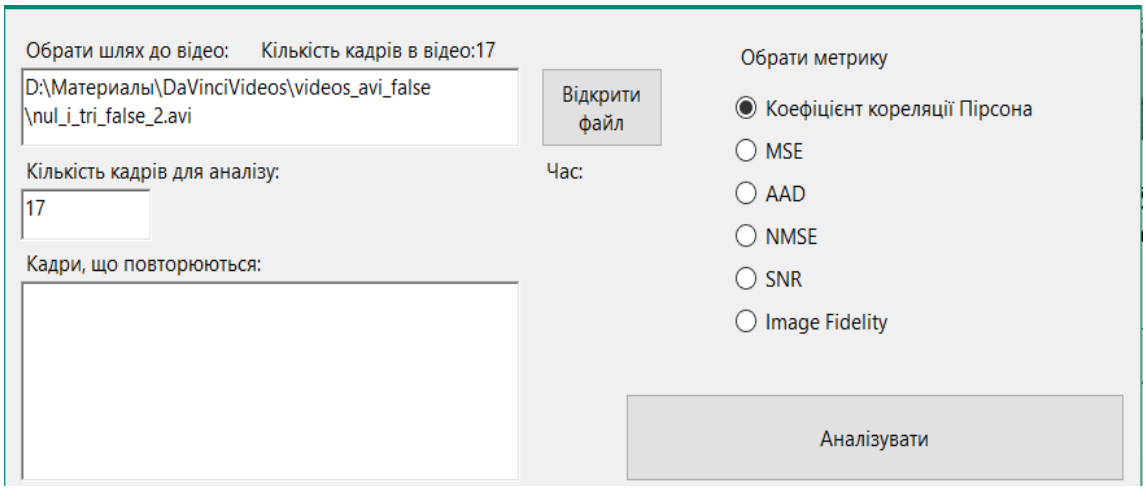

Рисунок 3.2 – початок аналізу

Коли аналіз завершиться, у полі «Кадри, що повторюються», можна буде переглянути його результати – номери кадрів, які програма вважає дубльованими. Також у рядку нижче кнопки «Відкрити файл» під ім'ям «Час» можна побачити кількість часу в годинах, хвилинах, секундах та мілісекундах, скільки зайняв аналіз (рисунок 3.3):

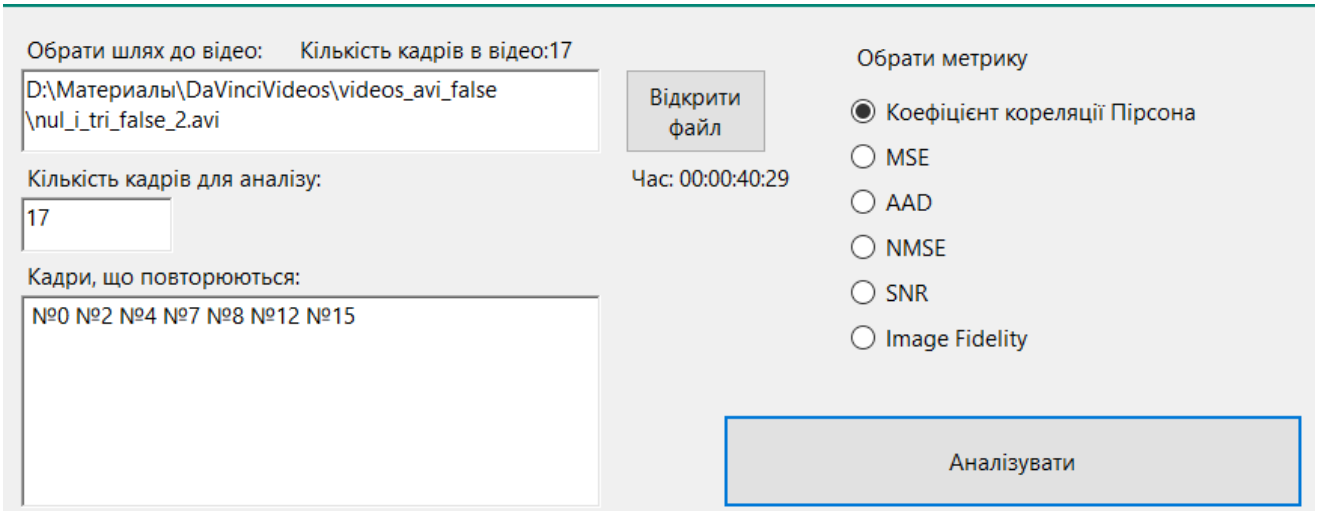

Рисунок 3.3 – результати аналізу

Оскільки програма працює ще й у тестовому форматі, то в полі з ім'ям «Поле для перевірки» можна переглянути список значень метрики кожної пари досліджуваних кадрів. Також поруч розташований інструмент, що дозволяє дізнатися значення метрики для двох кадрів, номери яких можна вибрати за допомогою стрілок, що знаходяться поруч(рисунок 3.4):

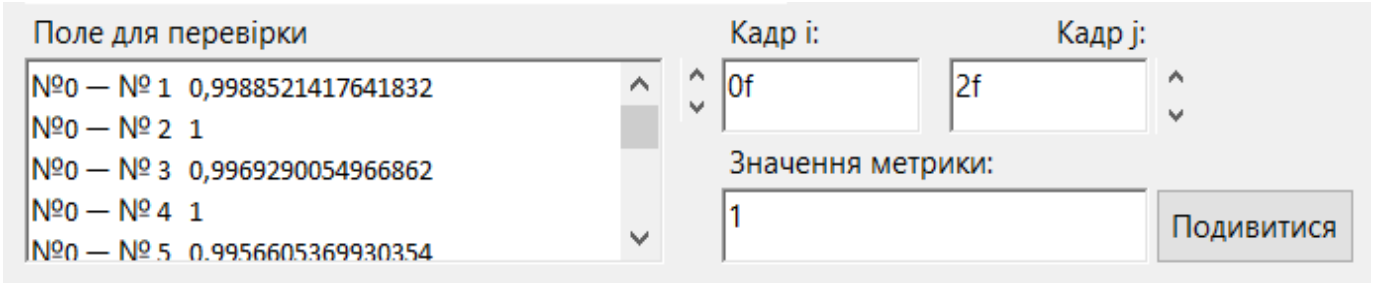

Рисунок 3.4 – перегляд значень метрики

3.3 Результати досліджень

Для фальсифікації досліджуваних відео використовувався метод фальшування без застосування алгоритмів стиснення.

Було розроблено доповнення до основної програми, яким можна скористатися, натиснувши кнопку «Додатково» у вікні основної програми. Відкриється таке вікно (рисунок 3.5):

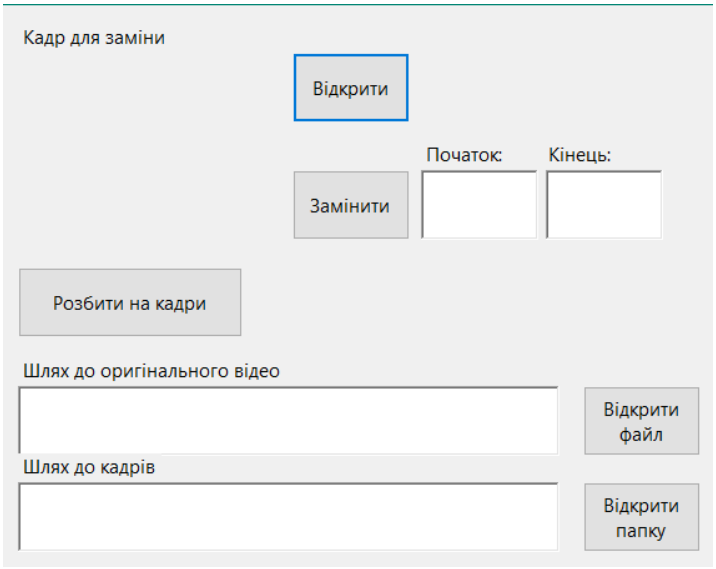

Рисунок 3.5 – інтерфейс доповнення

Спочатку вибирається шлях до відео, яке буде фальсифікуватися: потрібно натиснути кнопку «Відкрити файл» і вибрати необхідне відео у вікні. Далі потрібно вибрати шлях до папки, в якій зберігатимуться сфальшовані кадри, натиснувши на кнопку «Відкрити папку». Далі потрібно натиснути кнопку «Розбити на кадри»(рисунок 3.6):

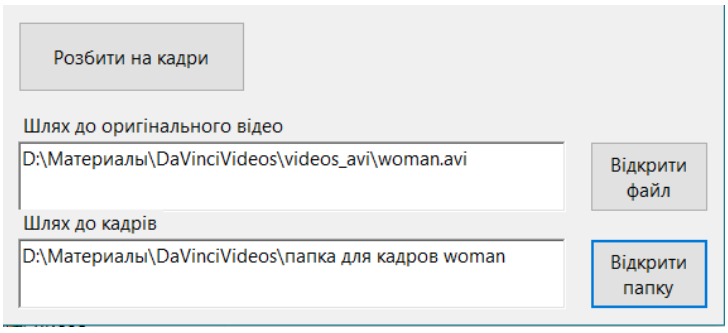

Рисунок 3.6 – вибір шляхів та розбиття

Результат розбиття вказано на рисунку 3.7:

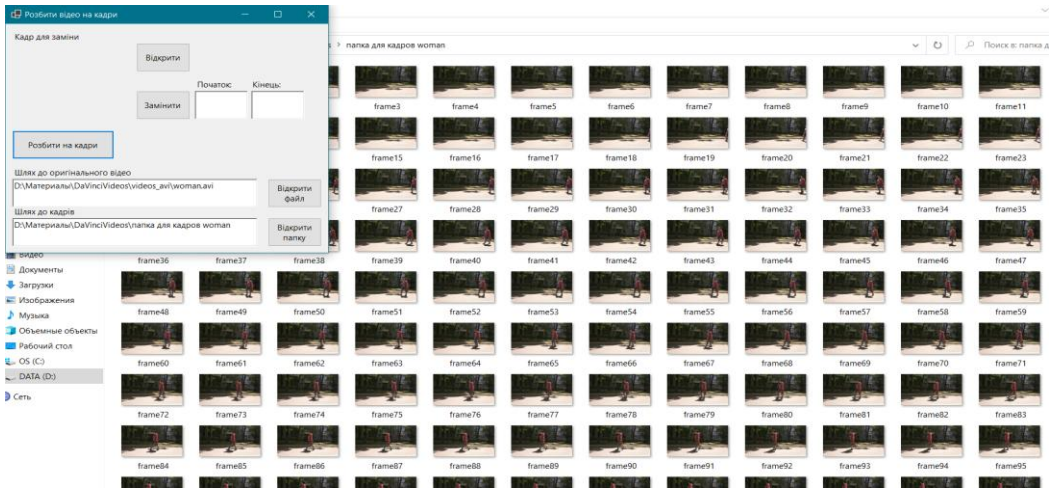

Рисунок 3.7 – результат розбиття

Потрібно переглянути кадри в папці та вибрати, які будуть замінені. У полях «Початок» та «Кінець» потрібно вписати номери кадрів, з якого і по який відповідно здійснюватиметься заміна, а також за допомогою кнопки «Відкрити» вибрати кадр або зображення, на яке здійснюватиметься заміна, а потім натиснути «Замінити»(рисунок 3.8):

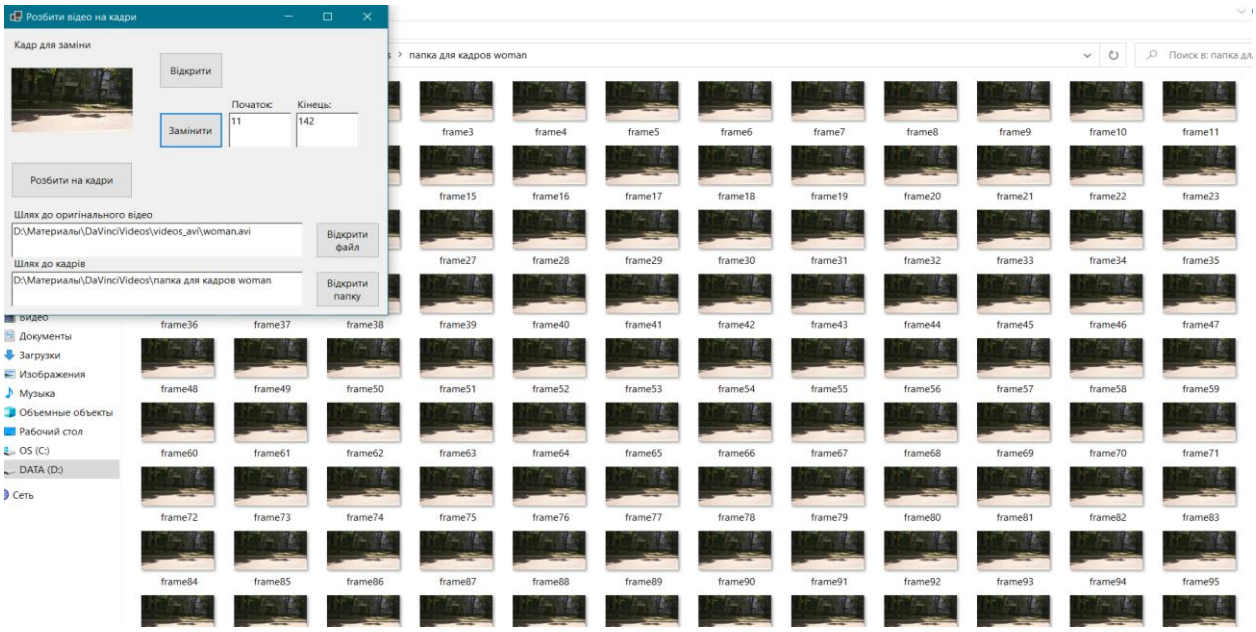

Рисунок 3.8 – результат заміни кадрів

Щоб зібрати сфальсифіковані кадри у відео, було використано програму Adobe After Effects 2022:

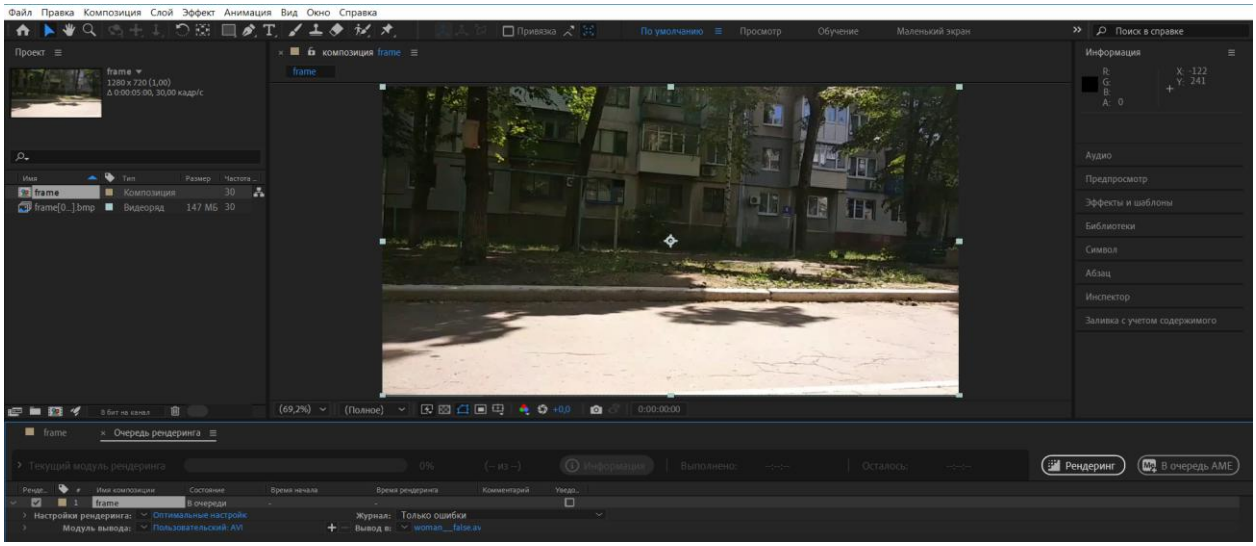

Рисунок 3.9 – процес відтворення відеопослідовності

Треба враховувати, що фальсифікація у такий спосіб відбувається без стискання, що дозволяє використовувати порогові значення 0 та 1 для MSE та коефіцієнту кореляції Пірсона відповідно, так само і з іншими метриками. Але структура програми дозволяє після деяких досліджень внести коригування у встановлені порогові значення, щоб аналізувати і відео із застосуванням різного ступеня стиснення теж.

Було проведено низку експериментів, метою яких було з'ясувати, з яким ступенем точності відпрацьовує алгоритм із різними метриками, а також порівняти час, який займає аналіз за допомогою кожної метрики.

Щоб з'явилася можливість порівняти час, витрачений на оцінку та аналіз, було експериментально взято значення кількості кадрів 25 і 50, а також кількість кадрів повного відео.

Результати досліджень 25 та 50 кадрів відео представлені в таблицях 3.1, 3.2.

Таблиця 3.1 – Дослідження 25 кадрів файлу birds_false.avi

Ім'я файлу	Метрика	Номери дубльованих кадрів	Час
birds_false.avi	Коефіцієнт кореляції Пірсона	№0, 19, 20, 21, 22, 23, 24	00:01:43:09
	MSE	№0, 19, 20, 21, 22, 23, 24	00:01:08:34
	AAD	№0, 19, 20, 21, 22, 23, 24	00:01:00:33
	NMSE	№0, 19, 20, 21, 22, 23, 24	00:01:11:42
	SNR	№0, 19, 20, 21, 22, 23, 24	00:01:10:45
	Image Fidelity	№0, 19, 20, 21, 22, 23, 24	00:01:10:12

Таблиця 3.2 – Дослідження 50 кадрів файлу birds_false.avi

Ім'я файлу	Метрика	Номери дубльованих кадрів	Час
birds_false.avi	Коефіцієнт кореляції Пірсона	№0, 19 — 49	00:10:58:66
	MSE	№0, 19 — 49	00:08:33:86
	AAD	№0, 19 — 49	00:08:17:50
	NMSE	№0, 19 — 49	00:09:18:87
	SNR	№0, 19 — 49	00:09:16:70
	Image Fidelity	№0, 19 — 49	00:09:06:69

Таблиця 3.2 – Дослідження 141 кадрів(всіх) файлу birds_false.avi

Ім'я файлу	Метрика	Номери дубльованих кадрів	Час
birds_false.avi	Коефіцієнт кореляції Пірсона	№0, 19 — 114	06:43:10:94
	MSE	№0, 19 — 114	05:57:12:04
	AAD	№0, 19 — 114	05:47:32:16
	NMSE	№0, 19 — 114	06:08:35:19
	SNR	№0, 19 — 114	06:04:03:78

	Image Fidelity	№0, 19 — 114	06:05:56:41

Як можна бачити з досліджень, через відсутність стиску алгоритм знаходить всі замінені кадри за допомогою кожної з метрик. Спостерігається різниця у часі роботи метрик. Метрикою, аналіз за допомогою якої займає максимальну кількість часу, є коефіцієнт кореляції Пірсона. Метрики, робота яких займає найменше часу - AAD і MSE. Приблизно однакова кількість часу займає аналіз за допомогою метрик SNR, NMSE та Image Fidelity.

Таким чином, було проведено низку експериментів, для точності опрацьовано кілька десятків відео, на яких метрики показали пропорціональні результати. Для аналізу відео із стисканням без втрат можна рекомендувати використовувати метрики AAD і MSE як найбільш швидкі.

4. ОХОРОНА ПРАЦІ

Середньостатистична людина проводить на роботі значну частину свого життя. Турбота про безпечні умови праці для працівників є однією із цілей сучасного цивілізованого суспільства. У процесі роботи працівник обов'язково стикається з небезпечними шкідливими факторами, які супроводжують робочий процес. Система охорони праці була створена, щоб забезпечити належну безпеку праці на підприємствах та турботу про здоров'я працюючого. Практика показує, що витрати на організацію та підтримання добре функціонуючої системи охорони праці менші у кілька разів, ніж витрати на покриття збитків у разі нехтування цими правилами. Також факт піклування про гідні умови праці робить підприємство привабливішим для поточних та потенційних працівників, які можуть бути набагато більше впевнені у безпеці для себе. Турбота про здоров'я працівників також може підвищити прибуток роботодавця — чим краще почувається людина, тим продуктивніше вона працює і тим якісніша її праця. Державою призначено санітарні норми та стандарти, яких повинні дотримуватися підприємства та організації, щоб виключити або мінімізувати ризик для життя і здоров’я людини.

В якості спроектованого місця роботи розглядається робоче місце експерта-криміналіста з персональним комп’ютером. Робоче місце — стіл —обладнане монітором персонального комп’ютера, системним блоком, клавіатурою, мишею, сканером та принтером.

В процесі праці за робочим місцем, що є об’єктом проектування, на працівника можуть впливати такі небезпечні та шкідливі фактори:

- недостатня освітленість;
- підвищений рівень шуму;
- недостатня вологість повітря;
- підвищений рівень електромагнітного випромінювання;
- можливість ураження електричним струмом.

При роботі навіть у такому відносно безпечному місці, як офіс, працівник може зазнати впливу шкідливих або небезпечних факторів, що навіть при ретельній турботі про дотримання належно продуманих правил безпеки важко виключити. Однак існує також безліч способів мінімізувати ризики нещасних випадків, травм та негативного впливу на здоров'я людського організму, що виходять з цих факторів.

Правильно спроектоване освітлення робочого місця – один із стовпів продуктивної роботи та гарного самопочуття. При нестачі освітлення люди швидше втомлюються, у них псується настрій, їхня здатність концентруватися на роботі погіршується, і якість їхньої праці знижується. Можуть розвиватися головний біль і втома центральної нервової системи. Також тривала робота в приміщенні з недостатнім освітленням може завдати серйозної шкоди зоровій системі людини, аж до розвитку специфічних хвороб органів зору. Фактичне значення освітленості в офісі, де розміщено спроектований об’єкт, становить 450лк.

Необхідні умови освітлення приміщення прописані у документі ДБН В.2.5-28-2018. В залежності від розряду зорових робіт, що виконуються працівниками, встановлюється значення мінімальної освітленості. Для IV розряду зорових робіт вона становить 300-500лк.

Для того, щоб освітлення в офісі було оптимальним, потрібно провести перевірку та виправити недоліки, якщо вони знайдені:

- робоче місце не повинно бути розміщене обличчям до вікна, якщо вікно виходить на сонячну сторону і на ньому немає фіранок, штор або жалюзі;
- освітлювальні лампи не повинні бути розташовані на рівні очей;
- світло не повинно бути надмірно тьмяним або надмірно яскравим;
- колірна температура всіх джерел освітлення має бути приблизно однаковою.

Організація розсіяного освітлення – найкращий вибір для офісу. Кількість та вид світильників визначається на підставі категорії роботи, яку виконує персонал. Штучне освітлення здійснюється завдяки загальній системі освітлення.

Якщо використовуються світильники місцевого освітлення, слід подбати про те, щоб вони не створювали дратуючих зір відблисків від монітору або поверхні стола. Також слід потурбуватися про наявність джерела природного освітлення, оскільки ніякий високотехнологічний світильник не зрівняється за показниками зі світлом сонця. Без природного освітлення людина швидше втомлюється, і швидкість її реакції падає.

Шум – це сукупність коливань середовища, що сприймаються людським вухом, яка негативно впливає на організм людини та викликає неприємні відчуття.

Коли робочим оточенням продукується зайвий рівень шуму, у працівників може виникати фізичний та психологічний стрес, нервозність, дратівливість, погіршується концентрація та продуктивність. Вплив занадто гучного шуму протягом тривалого часу може призвести до часткової або повної втрати слуху, що серйозно погіршить життя людини і може зробити її непрацездатною. Шум сильно відволікає увагу і може завадити сприйняти попереджувальні сигнали про будь-яку небезпеку, що може призвести до нещасних випадків та травм. Небезпечним є не тільки довгий рівномірний вплив зайвого шуму, але й короткі його періоди. До цих пір проводяться різноманітні дослідження, покликані визначити ефект впливу шуму на органі людини, і деякі з них говорять про те, що шум негативно впливає також на рівень кров'яного тиску і на рівень холестерину в організмі людини, з чого можна зробити висновок, що нехтувати методами мінімізації впливу шуму на працівників не варто.

Фактичне значення шуму в робочому приміщенні становить 30дБА.

Заходи безпеки та нормування щодо рівня шуму проводяться згідно з ДСН 3.3.6.037–99. Шум на робочому місці експерта-криміналіста не повинен перевищувати 50 дБА. Для нормалізації рівня шуму на робочому місці та зниження його впливу на організм працівника слід зробити деякі кроки щодо вдосконалення матеріального обладнання офісу, такі як заміна машин на більш тихі та збільшення частоти техобслуговувань. Також можна відмежувати працівників або ізолювати джерело шуму і організувати зони відпочинку, в яких

забезпечити тишу, щоб слух людей міг відновитися. Потрібно скласти розклад так, щоб кількість часу, який люди проводять на небезпечній відстані від шумної машини, була обмежена.

Також роботодавець повинен забезпечити доступ до засобів індивідуального захисту від шуму, таких як вкладиші, втулки та навушники.

Мікроклімат – це кліматичні умови внутрішнього середовища робочого приміщення, що впливають на теплообмін працівників з навколишнім середовищем.

Мікроклімат впливає на працездатність і здоров'я людини, тому що за оптимальних мікрокліматичних умов знижується розповсюдження інфекційних захворювань і ймовірність виникнення теплових ударів та переохолоджень. Організм людини почувається добре та задіює мінімальні зусилля для підтримки комфортної температури тіла, що дозволяє розподілити більше енергії на вирішення робочих задач.

Температура повітря – найважливіший показник. При надмірно високій температурі у приміщенні у працівників може виникати зневоднення, порушення водно-сольового балансу. Також знижується ясність свідомості. При постійно зниженій температурі повітря люди частіше хворіють, а також їхній організм більше працює на те, щоб зігрітися, а не для того, щоб якісніше виконувати свою роботу, і у працівників може спостерігатися сонливість і млявість. Вологість повітря залежить від температури. Коли повітря сухе, люди наражаються на небезпеку підхопити вірус, тому що слизова оболонка пересихає і стає вразливою. Підвищена ж вологість шкідлива для людей, хворих на астму, чи тих, що страждають на захворювання серцево-судинної системи. Також, якщо підвищена вологість зберігається протягом достатньо довгого періоду часу, це може сприяти розвитку цвілі та грибків, чиї спори згубно впливають на дихальну систему людини. Вплив швидкості руху повітря на організм залежить від температури повітря.

Оптимальні та допустимі мікрокліматичні умови регулюються документом ДСН 3.3.6.042-99 з урахуванням категорії важкості виконуваної роботи та періоду року.

Легка Іа – категорія роботи, що виконується експертом-криміналістом. Так як працівник знаходиться на робочому місці більш 2-х годин, то робоче місце вважається постійним. В таблиці 4.1 вказано оптимальні умови для категорії роботи легка Іа в різні періоди року.

Таблиця 4.1 — Оптимальні значення мікроклімату для категорії легка Іа

Період року	Температура повітря, °С	Відносна вологість, %	Швидкість руху повітря, м/сек
Холодний	22-24	60-40	0,1
Теплий	23-25	60-40	0,1

Щоб дотримуватися встановлених норм мікрокліматичних умов, роботодавець зобов'язаний обладнати захист робочих місць від джерел випромінювання тепла, забезпечувати справну роботу систем опалення, кондиціювання повітря та вентиляції. Для подолання проблеми недостатньої вологості слід встановити у приміщенні систему зволоження повітря.

Електромагнітні поля оточують людину, але вона їх не відчуває. Є два види джерел електромагнітних полів: антропогенні та природні. Офісна електротехніка відноситься до антропогенних.

Частота та інтенсивність електромагнітного випромінювання, а також тривалість його дії визначають вплив на організм людини. Електромагнітне випромінювання здатне надавати теплову дію на організм, яку не реєструють органи чуття, може призводити до порушення функціональності деяких систем організму, а також впливати на репродуктивну систему людини.

В офісному приміщенні є можливість отримати ураження електричним струмом. Воно загрожує можливістю порушення роботи чи зупинки серця чи дихання, ураження тканин мозку, виникнення опіків чи розривів м'язів. Травми, отримані в офісі, часто пов'язані з ураженням струмом. Причиною виникнення

загрози може бути використання неякісного чи пошкодженого обладнання чи ізоляції.

Правила електробезпеки - це першооснова загальної безпеки людини, і їм має бути приділена найпильніша увага. Якщо інші фактори діють протягом тривалого часу, і, навіть якщо деякі моменти були втрачені, їх вплив можна регулювати в довгій перспективі, то удар струмом діє швидко і часом назавжди.

Щоб мінімізувати ризики ураження струмом або виникнення пожежі, а також зменшити вплив електромагнітного поля на працівників слід дотримуватися правил:

- робоче обладнання має бути якісним та справним, регулярно піддаватися перевіркам та техогляду;
- дроти бажано закріпити в одному місці, це не дозволяє їм заплутатися, адже тоді легше пропустити пошкодження на дроті, і виключає можливість піддати їх механічним пошкодженням внаслідок необачного розташування їх, наприклад, посеред проходу;
- не можна користуватися електрошнурами, у яких пошкоджено ізоляцію;
- не можна накривати електрошнури килимами або покриттям для підлоги;
- не можна перевантажувати розетки, і слід утриматися від використання трійників;
- бажано використовувати електричну мережу, обладнану автоматичними вимикачами та пристроями захисного відключення;
- не можна використовувати обігрівачі, які перевантажують електричну мережу, а також не можна встановлювати їх поблизу легкозаймистих горючих матеріалів та експлуатувати в приміщенні з високою вологістю;
- слід вимикати прилади з розеток після закінчення використання, щоб унеможливити ризик нещасних випадків, коли працівники чекають цього найменше;
- не залишати електрообладнання без нагляду;
- віддалити робоче місце від джерела електромагнітного випромінювання;
- організувати проведення лікувально-профілактичних заходів для

персоналу.

Також доцільним буде планування та організація періодичних коротких тренінгів для персоналу, щоб нагадати та закріпити правила електробезпеки.

Відповідно до ДСТУ Б В.1.1-36:2016 було визначено категорію пожежної небезпеки "В" для приміщення, де може знаходитися спроектований об'єкт, оскільки горіти можуть електрообладнання та тверді речовини.

Джерелами потенційної небезпеки виникнення пожежі на робочому місці є персональний комп'ютер, периферія, електропроводка, електрошнури та розетки, а також обігрівачі чи кондиціонери у холодну чи теплу пору року відповідно. Пожежа може виникнути через перевантаження електромережі, коротке замикання, перегрів, що може бути викликане як випадковістю, так і недотриманням правил електробезпеки.

Для того, щоб зменшити ризик виникнення пожежі і, якщо раптом вона все ж таки трапиться, подолати наслідки з мінімальними втратами, слід реалізувати на робочому місці комплекс заходів з пожежної безпеки.

Електроустаткування в офісі має бути в робочому стані, проходити перевірки працездатності та негайно ремонтуватися у разі виявлення несправностей. При ремонті не слід забувати відключати прилади від електромережі.

Потрібно підготувати плани евакуації, розвісити їх на видних місцях та ознайомити всіх працівників із ними. Також слід подбати про те, щоб передбачуваній евакуації не завадили фізичні перешкоди, наприклад меблі або обладнання. Варто продумати цей момент наперед. Також можна написати великими літерами на плані евакуації номер пожежної служби, бо багато людей, впадаючи в паніку, можуть забувати навіть елементарні речі.

Для гасіння пожежі в приміщенні, де є електрообладнання, підходять порошкові або вуглекислотні вогнегасники. Потрібно стежити за справністю та належними умовами зберігання вогнегасників, часом їх перезаряджання. Потрібно знайти місце в офісі, в якому вогнегасник буде видно, де доступ до

нього буде легкий, і де він буде захищений від потенційних механічних пошкоджень. Відстань між вогнегасником та робочим місцем не повинна перевищувати 20 метрів.

Щоб якомога раніше дізнатися про виникнення вогнища, слід також забезпечити встановлення в офісі необхідної кількості детекторів диму, і переконатися, що вони справні і мають джерело живлення.

Важливо регулярно перевіряти всі засоби пожежної безпеки, встановлені в офісі, та ремонтувати або замінювати їх у разі виходу з ладу.

У багатьох людей є сильна залежність від куріння, і роботодавець повинен врахувати цей момент і відвести належне приміщення, де курити дозволено, тому що курити люди все одно будуть, і краще, щоб вони не робили цього на робочому місці і не викидали бички в кошик для сміття для паперових відходів, що також є надзвичайно пожежонебезпечним. Слід донести до співробітників, що курити в офісі суворо заборонено, та запровадити суворі штрафні санкції за порушення цієї заборони.

Слід призначити людину зі співробітників, яка буде відповідальна за пожежну безпеку в офісі. Інформацію про план евакуації, поведінку у разі пожежі, правила використання вогнегасників потрібно регулярно оновлювати в головах співробітників на спеціальних інформаційних інструктажах. Факт проходження працівником такого інструктажу слід зазначати у спеціальному журналі.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було:

- проведено аналіз літератури на тему роботи, існуючих та описаних методів фальсифікації та виявлення фальсифікацій у відео;

- знайдено та опрацьовано необхідні теоретичні відомості щодо об'єктів дослідження — зображень та відеоматеріалів;

- розроблено алгоритм виявлення клонування кадрів у цифрових відеопослідовностях з використанням компонентів Y, що зберігають у собі всю інформацію про зображення, колірного простору YUV в досліджуваних відеокадрах; для аналізу та оцінки використовувалися різні метрики оцінки якості, такі як коефіцієнт кореляції Пірсона, MSE, NMSE, SNR, AAD, Image Fidelity;

- алгоритм реалізований у вигляді програмного продукту;

- проведено низку експериментів над різними відеофайлами без стиснення, порівняння точності виявлення фальсифікованих кадрів та порівняння часу, який потрібний для аналізу кожною метрикою. З результатів проведених експериментів зроблено висновки, а також визначено перспективу модифікації програми для використання її надалі для аналізу відео з декотрим ступенем стиснення.

ПЕРЕЛІК ПОСИЛАНЬ

1. Проблеми фальсифікації фото- та відеоматеріалів на сучасному етапі розвитку цифровізації URL: http://www.techportal.ru/293097
2. Mauricio S., Monterrubio M., Maestre J. Vidal: Forensics video H264 falsification detection based on Energy Consumption. P.1.
3. Singh R.D., Aggarwal N: Video content authentication techniques: a comprehensive survey. P. 211–240
4. Li Q., Wang R., Xu D. An Inter-Frame Forgery Detection Algorithm for Surveillance Video. P.15.
5. De A., Chadha H., Gupta S.: Detection of forgery in digital video. In: *Proceedings of 10th World Multi Conference on Systems, Cybernetics and Informatics*. 2006. P. 229–233.
6. Detection of malevolent changes in digital video for forensic applications. *Proceedings of SPIE Conference on Security, Steganography and Watermarking of Multimedia Contents*. 2007. Vol. 6505, No. 1.
7. Kobayashi M., Okabe T., Sato Y. Detecting forgery from static-scene video based on inconsistency in noise level functions. *IEEE Trans. Inf. Forensics Secur*. 2010. 5(4), P.883–892.
8. Kancherla K., Mukkamal S. Novel blind video forgery detection using Markov models on motion residue. *Intell. Inf. Database Syst*. 2012. 7198, P.308–315.
9. Chao, J., Jiang, X., Sun, T. A novel video inter-frame forgery model detection scheme based on optical flow consistency. *Digit. Forensics Watermarking*. 2013. 7809. P. 267–281.
10. Wang Q., Li Z., Zhang Z., Ma Q. Video inter-frame forgery identification based on consistency of correlation coefficients of gray values. *J. Comput. Commun*. 2014. 2(4). P. 51–57.

11. Lin G.S., Chang J.F., Chuang F.H. Detecting frame duplication based on spatial and temporal analyses. In: *Proceedings of 6th IEEE International Conference on Computer Science and Education (ICCSE'11)*. P. 1396–1399.
12. Bestagini P., Battaglia S., Milani S., Tagliasacchi M., Tubaro S. Detection of temporal interpolation in videosequences. *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'13)*. 2013
13. Xia M., Yang G., Li L., Li R., Sun X. Detecting video frame rate up-conversion based on frame-level analysis of average texture variation. *Multimed. Tools Appl.* 2016. 72(1), P.1–23.
14. Forensics Investigations of Multimedia Data: A Review of the State-of-the-Art
15. Medaris K., Mislan R. Expert: Digital evidence just as important as dna in solving crimes. URL:http://news.uns.purdue.edu/x/2008a/080425T-MislanPhones.html
16. Lanh T.V., Chong K.S., Emmanuel S., Kankanhalli M.S. A survey on digital camera image forensic methods. *ICME*. 2007, P.16–19.
17. Houten W., Geradts Z. Source video camera identification for multiply compressed videos originating from youtube. *Digital Investigation*. 2009.Vol. 6, No. 1-2, P. 48–60.
18. Lee C., Lee J., Pyo Y., Broken H.L. Integrity Detection of Video Files in Video Event Data Recorders. 2—15p.
19. Houten W., Geradts Z. Source video camera identification for multiply compressed videos originating from youtube. *Digital Investigation*. 2009. V.6. No. 1-2. P. 48–60.
20. M.Tecalp: Digital video processing Prentice Hall; 1st edition 560c., 1995
21. Гонсалєс Р.С., Вудс Р.Є. Цифровая обработка изображений. М.: Техносфера, 2012. Р.22
22. Chan J. Learn C# In One Day and Learn It Well. 2015. 161p.
23. Albahary J., Albahary B. C# 8.0 Pocket Reference. O'Reilly Medi. 2019. 246p.

24. Whitaker R.B. C# Player’s Guide, Starbound Software. 2021, P.38
25. Ferrone H. Learning C# by Developing Games with Unity, Packt Publishing. 2019, P.55
26. Accord.Net Framework URL: http://accord-framework.net/docs/html/R_Project_Accord_NET.htm

Додаток А Лістинг програмного продукту

```
namespace videoframes
{
    public partial class Form1 : Form
    {
        Dictionary<string, double> dict_corr = new
Dictionary<string, double>();
        double value;
        Dictionary<string, double> dict_mse = new
Dictionary<string, double>();
        //Dictionary<int, int> check = new Dictionary<int,
int>();
        Dictionary<string, double> dict_normminc = new
Dictionary<string, double>();
        //Dictionary<string, double> dict_ssim = new
Dictionary<string, double>();
        Dictionary<string, double> dict_AAD = new
Dictionary<string, double>();
        Dictionary<string, double> dict_nmse = new
Dictionary<string, double>();
        Dictionary<string, double> dict_snr = new
Dictionary<string, double>();
        Dictionary<string, double> dict_imf = new
Dictionary<string, double>();
        int count_frames = 0;
        public Form1()
        {
            InitializeComponent();
        }
        private void open_file_Click(object sender, EventArgs
e)
        {
            OpenFileDialog Ofd = new OpenFileDialog();
            if (Ofd.ShowDialog() == DialogResult.OK)
            {
                path_tofile.Text = Ofd.FileName;
            }
            var videoFile = path_tofile.Text;
            var capture = new VideoCapture(videoFile);
            var image = new Mat();
            while (capture.IsOpened())
            {
                capture.Read(image);
                if (image.Empty())
                {
                    break;
                }
                count_frames++;
            }
            framescount.Text += count_frames.ToString();
```

```
            }
            private void to_do_Click(object sender, EventArgs e)
            {
                List<int> check = new List<int>();
                List<int> check_2 = new List<int>();
                List<int> check_3 = new List<int>();
                List<int> check_4 = new List<int>();
                List<int> check_5 = new List<int>();
                List<int> check_6 = new List<int>();
                List<int> check_7 = new List<int>();
                double threshold_corr = 1;
                double threshold_mse = 0;
                double threshold_normminc = 0;
                double threshold_AAD = 0;
                double threshold_nmse = 0;
                double threshold_snr = double.PositiveInfinity;
                double threshold_imf = 1;
                VideoFileReader reader = new VideoFileReader();
                reader.Open(path_tofile.Text);
                VideoFileReader falser = new VideoFileReader();
                falser.Open(path_tofile.Text);
                //Коэффициент корреляции Пирсона
                if (metrics_corr.Checked)
                {
                    Stopwatch watch = new Stopwatch();
                    watch.Start();
                    for (int i = 0; i < int.Parse(numberofframes.Text); i++)
                    {
                        Bitmap videoFrame = reader.ReadVideoFrame(i);

                        ImageReader im_original = new ImageReader(videoFrame);
                        im_original.CreateArrayYUV();
                        ColorModels.RGBToYUV(im_original.R, im_original.G, im_original.B, im_original.Y, im_original.U, im_original.V, im_original.ImWidth, im_original.ImHeight);
                        for (int j = 0; j < int.Parse(numberofframes.Text); j++)
                        {
                            if (i != j)
                            {
                                Bitmap falsevideoFrame = falser.ReadVideoFrame(j);
                                ImageReader im_false = new ImageReader(falsevideoFrame);
                                im_false.CreateArrayYUV();
                                ColorModels.RGBToYUV(im_false.R, im_false.G, im_false.B, im_false.Y, im_false.U, im_false.V, im_false.ImWidth, im_false.ImHeight);
```

```
                                double corr = Statistics.Corralation(im_original.Y, im_false.Y, im_original.ImWidth, im_original.ImHeight);

                                field_tocheck.Text += "№" + i.ToString() + " — № " + j.ToString() + "   " + corr + "\n";

                                dict_corr.Add(i.ToString() + "f" + j.ToString() + "f", corr);
                                if (corr == threshold_corr & i != j)
                                {
                                    check.Add(i);
                                }

                                if (falsevideoFrame != null)
                                {
                                    falsevideoFrame.Dispose();
                                }

                                if (im_false != null)
                                {
                                    im_false.Dispose();
                                }

                            }
                        }
                        if (videoFrame != null)
                        {
                            videoFrame.Dispose();
                        }
                        if (im_original != null)
                        {
                            im_original.Dispose();
                        }
                    }
                    reader.Close();
                    falser.Close();
                    if (dict_corr.TryGetValue(frame_i.Text + frame_j.Text, out value))
                    {
                        zn_cor.Text = value.ToString();
                    }
                    List<int> noDupes = check.Distinct().ToList();
                    foreach (var checking in noDupes)
                    {
                        replace_fr.Text += " №" + checking.ToString();
                    }
                    watch.Stop();
                    TimeSpan ts = watch.Elapsed;
```

```
                    string                elapsedTime                =
String.Format("{0:00}:{1:00}:{2:00}:{3:00}",  ts.Hours,  ts.Minutes,
ts.Seconds, ts.Milliseconds / 10);
                    time.Text += elapsedTime;
                }
                //MSE
                if (metrics_mse.Checked)
                {
                    Stopwatch watch = new Stopwatch();
                    watch.Start();
                    for      (int      i      =      0;      i      <
int.Parse(numberoofframes.Text); i++)
                    {
                        Bitmap              videoFrame              =
reader.ReadVideoFrame(i);
                        ImageReader      im_original      =      new
ImageReader(videoFrame);
                        im_original.CreateArrayYUV();
                        ColorModels.RGBToYUV(im_original.R,
im_original.G,    im_original.B,    im_original.Y,    im_original.U,
im_original.V, im_original.ImWidth, im_original.ImHeight);
                        for     (int     j     =     0;     j     <
int.Parse(numberoofframes.Text); j++)
                        {
                            if (i != j)
                            {

                                Bitmap       falsevideoFrame       =
falser.ReadVideoFrame(j);

                                ImageReader    im_false    =    new
ImageReader(falsevideoFrame);
                                im_false.CreateArrayYUV();
                                ColorModels.RGBToYUV(im_false.R,
im_false.G,   im_false.B,   im_false.Y,   im_false.U,   im_false.V,
im_false.ImWidth, im_false.ImHeight);
                                double             corr            =
Statistics.MSE_metrics(im_original.Y,                    im_false.Y,
im_original.ImWidth, im_original.ImHeight);
                                field_tocheck.Text                 =
field_tocheck.Text + "№" + i.ToString() + " — № " + j.ToString() + "
" + corr + "\n";
                                dict_mse.Add(i.ToString()  +  "f"  +
j.ToString() + "f", corr);
                                if (corr == threshold_mse & i != j)
                                {
                                    check_2.Add(i);
                                }
                                if (im_false != null)
                                {
                                    im_false.Dispose();
                                }
```

```
                                if (falsevideoFrame != null)
                                {
                                    falsevideoFrame.Dispose();
                                }
                            }
                        }
                        if (im_original != null)
                        {
                            im_original.Dispose();
                        }
                        if (im_original == null)
                        {
                            videoFrame.Dispose();
                        }
                    }
                    reader.Close();
                    falser.Close();
                    List<int>               noDupes_2               =
check_2.Distinct().ToList();
                    foreach (var checking in noDupes_2)
                    {
                        replace_fr.Text      +=      "      №"      +
checking.ToString();
                    }
                    watch.Stop();
                    TimeSpan ts = watch.Elapsed;
                    string                elapsedTime               =
String.Format("{0:00}:{1:00}:{2:00}:{3:00}",   ts.Hours,   ts.Minutes,
ts.Seconds, ts.Milliseconds / 10);
                    time.Text += elapsedTime;
                    //MessageBox.Show("Done!    "    +    elapsedTime,
"Message", MessageBoxButtons.OK, MessageBoxIcon.Information);
                }
                //AAD
                if (metrics_aad.Checked)
                {
                    Stopwatch watch = new Stopwatch();
                    watch.Start();
                    for      (int      i      =      0;      i      <
int.Parse(numberoffframes.Text); i++)
                    {
                        Bitmap              videoFrame              =
reader.ReadVideoFrame(i);
                        ImageReader      im_original      =      new
ImageReader(videoFrame);
                        im_original.CreateArrayYUV();
                        ColorModels.RGBToYUV(im_original.R,
im_original.G,     im_original.B,     im_original.Y,     im_original.U,
im_original.V, im_original.ImWidth, im_original.ImHeight);
                        for     (int     j     =     0;     j     <
int.Parse(numberoffframes.Text); j++)
                        {
                            if (i != j)
```

```
                        {
                            Bitmap       falsevideoFrame       =
falser.ReadVideoFrame(j);
                            ImageReader    im_false    =    new
ImageReader(falsevideoFrame);
                            im_false.CreateArrayYUV();
                            ColorModels.RGBToYUV(im_false.R,
im_false.G,    im_false.B,    im_false.Y,    im_false.U,    im_false.V,
im_false.ImWidth, im_false.ImHeight);
                            double            corr            =
Statistics.Average_Absolute_Difference(im_original.Y,     im_false.Y,
im_original.ImWidth, im_original.ImHeight);
                            field_tocheck.Text                =
field_tocheck.Text + "№" + i.ToString() + " — № " + j.ToString() + "
" + corr + "\n";
                            dict_AAD.Add(i.ToString()  +  "f"  +
j.ToString() + "f", corr);
                            if (corr == threshold_AAD & i != j)
                            {
                                check_4.Add(i);
                            }
                            if (im_false != null)
                            {
                                im_false.Dispose();
                            }
                            if (falsevideoFrame != null)
                            {
                                falsevideoFrame.Dispose();
                            }
                        }
                    }
                    if (im_original != null)
                    {
                        im_original.Dispose();
                    }
                    if (im_original == null)
                    {
                        videoFrame.Dispose();
                    }
                }
                reader.Close();
                falser.Close();
                List<int>              noDupes_4              =
check_4.Distinct().ToList();
                foreach (var checking in noDupes_4)
                {
                    replace_fr.Text     +=     "     №"     +
checking.ToString();
                }
                watch.Stop();
                TimeSpan ts = watch.Elapsed;
```

```
                    string                elapsedTime                =
String.Format("{0:00}:{1:00}:{2:00}:{3:00}",   ts.Hours,   ts.Minutes,
ts.Seconds, ts.Milliseconds / 10);
                    time.Text += elapsedTime;
                }
                //NMSE
                if (metrics_nmse.Checked)
                {
                    Stopwatch watch = new Stopwatch();
                    watch.Start();
                    for      (int      i      =      0;      i      <
int.Parse(numberofframes.Text); i++)
                    {
                        Bitmap              videoFrame              =
reader.ReadVideoFrame(i);

                        ImageReader      im_original      =      new
ImageReader(videoFrame);
                        im_original.CreateArrayYUV();
                        ColorModels.RGBToYUV(im_original.R,
im_original.G,     im_original.B,     im_original.Y,     im_original.U,
im_original.V, im_original.ImWidth, im_original.ImHeight);

                        for     (int     j     =     0;     j     <
int.Parse(numberofframes.Text); j++)
                        {
                            if (i != j)
                            {
                                Bitmap       falsevideoFrame       =
falser.ReadVideoFrame(j);

                                ImageReader    im_false    =    new
ImageReader(falsevideoFrame);
                                im_false.CreateArrayYUV();
                                ColorModels.RGBToYUV(im_false.R,
im_false.G,    im_false.B,    im_false.Y,    im_false.U,    im_false.V,
im_false.ImWidth, im_false.ImHeight);
                                double              corr             =
Statistics.NMSE(im_original.Y,    im_false.Y,    im_original.ImWidth,
im_original.ImHeight);
                                field_tocheck.Text                   =
field_tocheck.Text + "№" + i.ToString() + " — № " + j.ToString() + "
" + corr + "\n";
                                dict_nmse.Add(i.ToString()  +  "f"  +
j.ToString() + "f", corr);

                                if  (corr  ==  threshold_nmse  &  i  !=
j)
                                {
                                    check_5.Add(i);
                                }
                                if (im_false != null)
                                {
```

```
                                im_false.Dispose();
                            }
                            if (falsevideoFrame != null)
                            {
                                falsevideoFrame.Dispose();
                            }
                        }
                    }
                    if (im_original != null)
                    {
                        im_original.Dispose();
                    }
                    if (im_original == null)
                    {
                        videoFrame.Dispose();
                    }
                }
                reader.Close();
                falser.Close();
                List<int>               noDupes_5               =
check_5.Distinct().ToList();
                foreach (var checking in noDupes_5)
                {
                    replace_fr.Text      +=      "      №"      +
checking.ToString();
                }
                watch.Stop();
                TimeSpan ts = watch.Elapsed;
                string               elapsedTime               =
String.Format("{0:00}:{1:00}:{2:00}:{3:00}",  ts.Hours,  ts.Minutes,
ts.Seconds, ts.Milliseconds / 10);
                time.Text += elapsedTime;
            }
            //SNR
            if (metrics_snr.Checked)
            {
                Stopwatch watch = new Stopwatch();
                watch.Start();
                for      (int      i      =      0;      i      <
int.Parse(numberoffframes.Text); i++)
                {
                    Bitmap              videoFrame              =
reader.ReadVideoFrame(i);
                    ImageReader      im_original      =      new
ImageReader(videoFrame);
                    im_original.CreateArrayYUV();
                    ColorModels.RGBToYUV(im_original.R,
im_original.G,     im_original.B,     im_original.Y,     im_original.U,
im_original.V, im_original.ImWidth, im_original.ImHeight);
                    for     (int     j     =     0;     j     <
int.Parse(numberoffframes.Text); j++)
                    {
                        if (i != j)
```

```
                                {
                                    Bitmap       falsevideoFrame       =
falser.ReadVideoFrame(j);
                                    ImageReader    im_false    =    new
ImageReader(falsevideoFrame);
                                    im_false.CreateArrayYUV();
                                    ColorModels.RGBToYUV(im_false.R,
im_false.G,    im_false.B,    im_false.Y,    im_false.U,    im_false.V,
im_false.ImWidth, im_false.ImHeight);
                                    double             corr             =
Statistics.SNR(im_original.Y,     im_false.Y,     im_original.ImWidth,
im_original.ImHeight);
                                    field_tocheck.Text                   =
field_tocheck.Text + "№" + i.ToString() + " — № " + j.ToString() + "
" + corr + "\n";
                                    dict_snr.Add(i.ToString()  +  "f"  +
j.ToString() + "f", corr);

                                    if (corr == threshold_snr & i != j)
                                    {
                                        check_6.Add(i);
                                    }
                                    if (im_false != null)
                                    {
                                        im_false.Dispose();
                                    }

                                    if (falsevideoFrame != null)
                                    {
                                        falsevideoFrame.Dispose();
                                    }
                                }
                            }
                            if (im_original != null)
                            {
                                im_original.Dispose();
                            }
                            if (im_original == null)
                            {
                                videoFrame.Dispose();
                            }
                        }
                        reader.Close();
                        falser.Close();
                        List<int>               noDupes_6               =
check_6.Distinct().ToList();
                        foreach (var checking in noDupes_6)
                        {
                            replace_fr.Text      +=      "      №"      +
checking.ToString();
                        }
                        watch.Stop();
```

```
                    TimeSpan ts = watch.Elapsed;
                    string elapsedTime =
String.Format("{0:00}:{1:00}:{2:00}:{3:00}", ts.Hours, ts.Minutes,
ts.Seconds, ts.Milliseconds / 10);
                    time.Text += elapsedTime;
                }
                //Image Fidelity
                if (metrics_ImF.Checked)
                {
                    Stopwatch watch = new Stopwatch();
                    watch.Start();
                    for (int i = 0; i <
int.Parse(numberoframes.Text); i++)
                    {
                        Bitmap videoFrame =
reader.ReadVideoFrame(i);

                        ImageReader im_original = new
ImageReader(videoFrame);
                        im_original.CreateArrayYUV();
                        ColorModels.RGBToYUV(im_original.R,
im_original.G, im_original.B, im_original.Y, im_original.U,
im_original.V, im_original.ImWidth, im_original.ImHeight);

                        for (int j = 0; j <
int.Parse(numberoframes.Text); j++)
                        {
                            if (i != j)
                            {
                                Bitmap falsevideoFrame =
falser.ReadVideoFrame(j);

                                ImageReader im_false = new
ImageReader(falsevideoFrame);
                                im_false.CreateArrayYUV();
                                ColorModels.RGBToYUV(im_false.R,
im_false.G, im_false.B, im_false.Y, im_false.U, im_false.V,
im_false.ImWidth, im_false.ImHeight);
                                double corr =
Statistics.ImF(im_original.Y, im_false.Y, im_original.ImWidth,
im_original.ImHeight);
                                field_tocheck.Text =
field_tocheck.Text + "№" + i.ToString() + " — № " + j.ToString() + "
" + corr + "\n";
                                dict_imf.Add(i.ToString() + "f" +
j.ToString() + "f", corr);

                                if (corr == threshold_imf & i != j)
                                {
                                    check_7.Add(i);
                                }
                                if (im_false != null)
                                {
```

```
                                im_false.Dispose();
                            }
                            if (falsevideoFrame != null)
                            {
                                falsevideoFrame.Dispose();
                            }
                        }
                    }
                    if (im_original != null)
                    {
                        im_original.Dispose();
                    }
                    if (im_original == null)
                    {
                        videoFrame.Dispose();
                    }
                }
                reader.Close();
                falser.Close();
                List<int>               noDupes_7               =
check_7.Distinct().ToList();
                foreach (var checking in noDupes_7)
                {
                    replace_fr.Text      +=      "      №"      +
checking.ToString();
                }
                watch.Stop();
                TimeSpan ts = watch.Elapsed;
                string               elapsedTime               =
String.Format("{0:00}:{1:00}:{2:00}:{3:00}",  ts.Hours,  ts.Minutes,
ts.Seconds, ts.Milliseconds / 10);
                time.Text += elapsedTime;
            }
        }
        private  void  path_tofile_TextChanged(object  sender,
EventArgs e)
        {

        }
        private  void  look_corr_Click(object  sender,  EventArgs
e)
        {
            if (metrics_corr.Checked)
            {
                field_tocheck.Text = "Корреляция Пирсона : ";
                foreach (var corrr in dict_corr)
                {
                    field_tocheck.Text  +=  "\n"  +  ($"кадры:
{corrr.Key}  зн.кор.: {corrr.Value}");
                }
            }
            if (metrics_mse.Checked)
            {
```

```
                    field_tocheck.Text = "MSE: ";
                    foreach (var corrr in dict_mse)
                    {
                        field_tocheck.Text  +=  "\n"  +  ($"кадры:
{corrr.Key}  зн.кор.: {corrr.Value}");
                    }
                }
                if (metrics_aad.Checked)
                {
                    field_tocheck.Text    =    "Average    Absolute
Difference: ";
                    foreach (var corrr in dict_AAD)
                    {
                        field_tocheck.Text  +=  "\n"  +  ($"кадры:
{corrr.Key}  зн.кор.: {corrr.Value}");
                    }
                }
                if (metrics_nmse.Checked)
                {
                    field_tocheck.Text = "NMSE: ";
                    foreach (var corrr in dict_nmse)
                    {
                        field_tocheck.Text  +=  "\n"  +  ($"кадры:
{corrr.Key}  зн.кор.: {corrr.Value}");
                    }
                }
                if (metrics_snr.Checked)
                {
                    field_tocheck.Text = "SNR: ";
                    foreach (var corrr in dict_snr)
                    {
                        field_tocheck.Text  +=  "\n"  +  ($"кадры:
{corrr.Key}  зн.кор.: {corrr.Value}");
                    }
                }
                if (metrics_ImF.Checked)
                {
                    field_tocheck.Text = "Image Fidelity: ";
                    foreach (var corrr in dict_imf)
                    {
                        field_tocheck.Text  +=  "\n"  +  ($"кадры:
{corrr.Key}  зн.кор.: {corrr.Value}");
                    }
                }

            }
            private    void    vScrollBar_i_Scroll(object    sender,
ScrollEventArgs e)
            {
                if (metrics_corr.Checked)
                {
                    frame_i.Text  =  vScrollBar_i.Value.ToString()  +
"f";
```

```
                    if     (dict_corr.TryGetValue(frame_i.Text     +
frame_j.Text, out value))
                    {
                        zn_cor.Text = value.ToString();
                    }
                }
                if (metrics_mse.Checked)
                {
                    frame_i.Text  =  vScrollBar_i.Value.ToString()  +
"f";
                    if     (dict_mse.TryGetValue(frame_i.Text     +
frame_j.Text, out value))
                    {
                        zn_cor.Text = value.ToString();
                    }
                }
                if (metrics_aad.Checked)
                {
                    frame_i.Text  =  vScrollBar_i.Value.ToString()  +
"f";
                    if     (dict_AAD.TryGetValue(frame_i.Text     +
frame_j.Text, out value))
                    {
                        zn_cor.Text = value.ToString();
                    }
                }
                if (metrics_nmse.Checked)
                {
                    frame_i.Text  =  vScrollBar_i.Value.ToString()  +
"f";
                    if     (dict_nmse.TryGetValue(frame_i.Text     +
frame_j.Text, out value))
                    {
                        zn_cor.Text = value.ToString();
                    }
                }
                if (metrics_snr.Checked)
                {
                    frame_i.Text  =  vScrollBar_i.Value.ToString()  +
"f";
                    if     (dict_snr.TryGetValue(frame_i.Text     +
frame_j.Text, out value))
                    {
                        zn_cor.Text = value.ToString();
                    }
                }
                if (metrics_ImF.Checked)
                {
                    frame_i.Text  =  vScrollBar_i.Value.ToString()  +
"f";
                    if     (dict_imf.TryGetValue(frame_i.Text     +
frame_j.Text, out value))
                    {
```

```
                        zn_cor.Text = value.ToString();
                    }
                }
            }
            private    void    vScrollBar_j_Scroll(object    sender,
ScrollEventArgs e)
            {
                if (metrics_corr.Checked)
                {
                    frame_j.Text  =  vScrollBar_j.Value.ToString()  +
"f";
                    if (frame_i.Text == frame_j.Text)
                    {
                        zn_cor.Text = "1";
                    }
                    else  if  (dict_corr.TryGetValue(frame_i.Text  +
frame_j.Text, out value))
                    {
                        zn_cor.Text = value.ToString();
                    }
                }
                if (metrics_mse.Checked)
                {
                    frame_j.Text  =  vScrollBar_j.Value.ToString()  +
"f";
                    if (frame_i.Text == frame_j.Text)
                    {
                        zn_cor.Text = "0";
                    }
                    else   if   (dict_mse.TryGetValue(frame_i.Text   +
frame_j.Text, out value))
                    {
                        zn_cor.Text = value.ToString();
                    }
                }
                if (metrics_aad.Checked)
                {
                    frame_j.Text  =  vScrollBar_j.Value.ToString()  +
"f";
                    if (frame_i.Text == frame_j.Text)
                    {
                        zn_cor.Text = "0";
                    }
                    else   if   (dict_AAD.TryGetValue(frame_i.Text   +
frame_j.Text, out value))
                    {
                        zn_cor.Text = value.ToString();
                    }
                }
                if (metrics_nmse.Checked)
                {
                    frame_j.Text  =  vScrollBar_j.Value.ToString()  +
"f";
```

```
                    if (frame_i.Text == frame_j.Text)
                    {
                        zn_cor.Text = "0";
                    }
                    else  if  (dict_nmse.TryGetValue(frame_i.Text  +
frame_j.Text, out value))
                    {
                        zn_cor.Text = value.ToString();
                    }
                }
                if (metrics_snr.Checked)
                {
                    frame_j.Text  =  vScrollBar_j.Value.ToString()  +
"f";
                    if (frame_i.Text == frame_j.Text)
                    {
                        zn_cor.Text                                    =
double.PositiveInfinity.ToString();
                    }
                    else  if  (dict_snr.TryGetValue(frame_i.Text  +
frame_j.Text, out value))
                    {
                        zn_cor.Text = value.ToString();
                    }
                }
                if (metrics_ImF.Checked)
                {
                    frame_j.Text  =  vScrollBar_j.Value.ToString()  +
"f";
                    if (frame_i.Text == frame_j.Text)
                    {
                        zn_cor.Text = "1";
                    }
                    else  if  (dict_imf.TryGetValue(frame_i.Text  +
frame_j.Text, out value))
                    {
                        zn_cor.Text = value.ToString();
                    }
                }
            }
            private void two_page_Click(object sender, EventArgs e)
            {
                Form2 newForm = new Form2();
                newForm.Show();
            }

            private void framescount_Click(object sender, EventArgs
e)
            {

            }
        }
    }
```