НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ
КАФЕДРА СИСТЕМНОГО ПРОГРАМУВАННЯ І
СПЕЦІАЛІЗОВАНИХ КОМП'ЮТЕРНИХ СИСТЕМ

«На правах рукопису»

УДК _**519.688**_____

«До захисту допущено»

Завідувач кафедри СПСКС

_____
В.П.Тарасенко_
(підпис) (ініціали, прізвище)

"___"_____2018р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 123 Комп'ютерна інженерія

Комп'ютерні системи та компоненти

на тему: ПРОГРАМНІ ЗАСОБИ РОЗПІЗНАВАННЯ РУХОМОГО ОБ'ЄКТА ДЛЯ СИСТЕМ ВІДЕОСПОСТЕРЕЖЕННЯ

Виконав: студент II курсу, групи _КВ-71мп_____
(шифр групи)

Лисенко Віталій Андрійович _____ _____
(прізвище, ім'я, по батькові) (підпис)

Науковий керівник: доцент кафедри СПіСКС, к.т.н.,

доцент Петрашенко А.В. _____ _____

(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Рецензент: _____ _____

(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.

Студент _____

(підпис)

Київ – 2018 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра системного програмування і спеціалізованих комп'ютерних систем

Рівень вищої освіти – другий (магістерський)
Спеціальність 123 Комп'ютерна інженерія
Спеціалізовані комп'ютерні системи

ЗАТВЕРДЖУЮ
Завідувач кафедри СПСКС
_____ Тарасенко В.П.
(підпис) (ініціали, прізвище)
"___" _____ 2018 р.

ЗАВДАННЯ
на магістерську дисертацію студенту

Лисенку Віталію Андрійовичу
(прізвище, ім'я, по батькові)

1. Тема дисертації «Програмні засоби розпізнавання рухомого об'єкта для систем відеоспостереження»»

науковий керівник дисертації доц.каф.СПСКС, к.т.н., доц. Петрашенко А.В.

затверджені наказом по університету від 30 жовтня 2018 р. №4030-с

2. Термін подання студентом дисертації: 7 грудня 2018 р.

3. Об'єкт дослідження: розпізнавання руху об'єкта в системах відеоспостереження.

4. Предмет дослідження: розробка програмних засобів розпізнавання руху об'єкта для інтеграції у системи відеоспостереження, які призначені для зменшення розміру пам'яті, що використовується для збереження відеоматеріалу.

5. Перелік завдань, які потрібно розробити:

- розглянути та проаналізувати існуючі рішення систем відеоспостереження;
- запропонувати новий підхід фіксації даних;
- розробити програмні засоби для реалізації запропонованого підходу;
- протестувати роботу запропонованого підходу, зробити висновки.

6. Перелік ілюстративного матеріалу: презентація (кількість аркушів: 12)

7. Перелік публікацій:

- наукова конференція магістрантів та аспірантів «Прикладна математика та комп'ютинг» ПМК-2018 (Київ, 14-16 листопада 2018р.);
- V Міжнародна науково-технічна Internet-конференція «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами» (22 листопада 2018 р. , Національний університет харчових технологій)

8. Дата видачі завдання 5 вересня 2017 .

Календарний план

№	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1.	Розроблення та узгодження завдання магістерської дисертації	57.10.2017	
2.	Аналіз існуючих підходів фіксування даних	14.01.2018	
3.	Тестування існуючих підходів фіксування даних	02.03.2018	
4.	Підготовка матеріалів 1-го розділу магістерської дисертації	24.04.2018	
5.	Розробка власного підходу	27.06.2018	
6.	Підготовка матеріалів 2-го розділу магістерської дисертації	27.07.2018	
7.	Реалізація запропонованого підходу	16.09.2018	
8.	Підготовка матеріалів 3-го розділу магістерської дисертації	29.10.2018	
9.	Підготовка матеріалів 4-го розділу дипломного проекту	17.10.2018	
10.	Оформлення документації дипломного проекту	25.11.2018	
11.	Підготовка графічної частини дипломного проекту	05.12.2018	
	Попередній розгляд магістерської дисертації на кафедрі	26.11.2018	

Студент _____ ___Лисенко В.А._____

(підпис) (прізвище, ініціали)

Науковий керівник дисертації _____ Петрашенко А.В.
(підпис) (прізвище, ініціали)

РЕФЕРАТ

Актуальність теми.

Сучасні системи відеоспостереження відіграють важливу роль у підвищенні рівня безпеки та дотримання прав приватної власності об'єкта під охороною. Щодня компанії, що працюють у сфері систем охорони, розробляють нові технології, які здатні ефективно усувати несприятливі ситуації та підвищувати рівень безпеки.

Незважаючи на те, що традиційні системи відеоспостереження є досить поширеними в сучасних охоронних системах та демонструють високий рівень захисту, вони мають певні недоліки. Основним недоліком таких систем є високі затрати пам'яті на збереження зафіксованого відеоматеріалу, оскільки подібні системи постійно фіксують ситуацію на закріпленій території та зберігають велику кількість відеоматеріалу, навіть коли в місці спостереження нічого не відбувається. Це також приводить до збільшення часу, що витрачається на перегляд та аналіз збережених матеріалів.

Тому, щоб підвищити ефективність систем відеоспостереження, потрібно позбутися наведеного недоліку, що і стало предметом даного дослідження.

Об’єктом дослідження є розпізнавання руху об’єкта в системах відеоспостереження.

Предметом дослідження є програмні засоби для підвищення ефективності систем відеоспостереження.

Мета роботи: зменшення розміру пам’яті, затраченої на збереження відеоматеріалів, а також розробка програмних засобів розпізнавання руху об’єкта для інтеграції у системи відеоспостереження.

Наукова новизна:

1. Запропоновано спосіб аналізу відеоматеріалу, який базується на порівнянні кадрів і в залежності від результату порівняння приймається рішення про збереження даного матеріалу. Це дозволяє зменшити розмір відеоматеріалів, що фіксується за певний проміжок часу.

2. Виконано порівняльний аналіз звичайної системи відеоспостереження, роботу якої було відтворено на комп'ютері, та роботу системи, яка використовує метод розпізнавання рухомих об'єктів. Також визначено ситуації, в яких дана система буде корисною, та її переваги і недоліки.

Практична цінність отриманих в роботі результатів полягає в тому, що запропонована система надає можливість значно зменшити розмір пам'яті, відведений для збереження відеоматеріалів. Крім того, запропонована система може бути інтегрована в сучасні системи відеоспостереження майже без апаратних змін.

Апробація роботи. Робота системи відеоспостереження з інтегрованою системою розпізнавання руху об'єктів була представлена та обговорювалась на науковій конференції магістрантів та аспірантів "Прикладна математика та комп'ютинг" ПМК-2018 (Київ, 14 – 16 листопада 2018) і на IV Міжнародній науково-технічній Internet-конференції «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами», яка проводилась 22 листопада 2018 в Національному університеті харчових технологій.

Структура та обсяг роботи. Магістерська дисертація складається з вступу, чотирьох розділів та висновків.

У вступі подано загальну характеристику роботи, зроблено оцінку сучасного стану проблеми, обґрунтовано актуальність напрямку досліджень, сформульовано мету і задачі досліджень, показано наукову новизну отриманих результатів.

У першому розділі розглянуто існуючі системи відеоспостереження, їхні особливості, недоліки та переваги, розглянуто різні реалізації.

У другому розділі запропоновано систему відеоспостереження яка буде вирішувати виявлену проблему, базуючись на аналізуванні кадрів вхідного відеоматеріалу.

У третьому розділі наведені особливості реалізації розроблених програмних засобів для реалізації запропонованого підходу.

У четвертому розділі представлено результати тестування розроблених програмних засобів.

У висновках представлені результати проведеної роботи.

Робота представлена на 84 аркушах, містить посилання на список використаних літературних джерел.

Ключові слова: система відеоспостереження, рухомий об’єкт, відеоматеріал.

РЕФЕРАТ

Актуальность темы.

Современные системы видеонаблюдения играют важную роль в повышении уровня безопасности и соблюдения прав частной собственности объекта под охраной. Ежедневно компании, работающие в сфере систем охраны, разрабатывают новые технологии, которые способны эффективно устранять неблагоприятные ситуации и повышать уровень безопасности.

Несмотря на то, что традиционные системы видеонаблюдения являются довольно распространенными в современных охранных системах и демонстрируют высокий уровень защиты, они имеют определенные недостатки. Основным недостатком таких систем являются высокие затраты памяти на хранение зафиксированного видеоматериала, поскольку подобные системы постоянно фиксируют ситуацию на закрепленной территории и сохраняют большое количество видеоматериала, даже если в месте наблюдения ничего не происходит. Это также приводит к увеличению времени, затрачиваемого на просмотр и анализ сохранившихся материалов.

Поэтому, чтобы повысить эффективность систем видеонаблюдения, нужно избавиться приведенного недостатка, что и стало предметом данного исследования.

Объектом исследования является распознавание движения объекта в системах видеонаблюдения.

Предметом исследования есть программные средства для повышения эффективности систем видеонаблюдения.

Цель работы: уменьшение размера памяти, затраченного на сохранение видеоматериалов, а также разработка программных средств распознавания движения объекта для интеграции в системы видеонаблюдения.

Научная новизна:

1. Предложен способ анализа видеоматериала, который базируется на сравнении кадров и в зависимости от результата сравнения принимается решение о сохранении данного материала. Это позволяет уменьшить размер видеоматериалов, который фиксируется за определенный промежуток времени.

2. Выполнен сравнительный анализ обычной системы видеонаблюдения, работу которой было воспроизведено на компьютере, и работу системы, которая использует метод распознавания движущихся объектов. Также определены ситуации, в которых данная система будет полезной, и ее преимущества и недостатки.

Практическая ценность полученных в работе результатов заключается в том, что предложенная система позволяет значительно уменьшить размер памяти, отведенный для сохранения видеоматериалов. Кроме того, предложенная система может быть интегрирована в современные системы видеонаблюдения почти без аппаратных изменений.

Апробация работы. Работа системы видеонаблюдения с интегрированной системой распознавания движения объектов была представлена и обсуждалась на научной конференции магистрантов и аспирантов "Прикладная математика и компьютинг" ПМК-2018 (Киев, 14 - 16 августа 2018) и на IV Международной научно-технической Internet- конференции «Современные методы, информационное, программное и техническое обеспечение систем управления организационно-техническими и технологическими комплексами», которая проводилась 22 ноября 2018 в Национальном университете пищевых технологий.

Структура и объем работы. Магистерская диссертация состоит из введения, четырех глав и выводов.

Во *введении* представлена общая характеристика работы, произведена оценка современного состояния проблемы, обоснована актуальность направления исследований, сформулированы цели и задачи исследований, показано научную новизну полученных результатов.

В *первом разделе* рассмотрены существующие системы видеонаблюдения, их особенности, недостатки и преимущества, рассмотрены различные реализации.

Во *втором разделе* предложена система видеонаблюдения которая будет решать обнаруженную проблему, основываясь на анализе кадров входящего видеоматериала.

В *третьем разделе* приведены особенности реализации разработанных программных средств для реализации предложенного подхода.

В *четвертом разделе* представлены результаты тестирования разработанных программных средств.

В *выводах* представлены результаты проведенной работы.

Работа представлена на 84 листах, содержит ссылки на список использованных литературных источников.

Ключевые слова: система видеонаблюдения, движущийся объект, видеоматериал.

ABSTRACT

Actuality of theme.

Modern CCTV systems play an important role in increasing the level of security and observance of the private property rights of the protected object. Every day, companies working in the field of security systems are developing new technologies that can effectively eliminate adverse situations and increase security.

Despite the fact that traditional CCTV systems are quite common in modern security systems and display a high level of protection, they have certain disadvantages. The main disadvantage of such systems is the high cost of memory to save the recorded video, since such systems permanently capture the situation on a fixed territory and store a large amount of video material, even when nothing happens on the spot of observation. This also leads to an increase in the time spent on viewing and analyzing stored materials.

Therefore, in order to improve the effectiveness of video surveillance systems, you need to get rid of this shortcoming, which became the subject of this study.

The object of the study is the recognition of object movement in video surveillance systems.

The subject of the study is software tools for improving the effectiveness of video surveillance systems.

The purpose of the work is to reduce the amount of memory spent on the storage of video materials by analysis of personnel, as well as the development of software for recognizing the object movement for integration into video surveillance systems.

Scientific novelty:

1. The method of analysis of video material, which is based on comparison of frames, and depending on the result of comparison, is made decision on the preservation of this material. This allows you to reduce the size of video footage, which is fixed for a certain period of time.

2. A comparative analysis of the usual CCTV system performed on the computer was performed, and the system work using the method of recognition of moving objects. It also identifies the situations in which this system will be useful, and its advantages and disadvantages.

The practical value of the results obtained in the work is that the proposed system provides the opportunity to significantly reduce the size of memory allocated to save video materials. In addition, the proposed system can be integrated into modern video surveillance systems with almost no hardware changes.

Test work. The work of the CCTV system with the integrated object recognition recognition system was presented and discussed at the Scientific Conference of Masters and Postgraduates "Applied Mathematics and Computer", PMK-2018 (Kyiv, November 14-16, 2018) and at the IV International Scientific and Technical Internet- conference "Modern methods, information, software and technical support of control systems for organizational, technical and technological complexes", held on November 22, 2018 at the National University of Food Technologies.

Structure and scope of work. The master's thesis consists of an introduction, four chapters and conclusions.

The introduction gives a general description of the work, assesses the current state of the problem, substantiates the relevance of the research direction, formulates the purpose and objectives of the research, shows the scientific novelty of the results.

The *first section* deals with existing video surveillance systems, their features, disadvantages and advantages, different implementations are considered.

The *second section* proposes a video surveillance system that will solve the problem, based on the analysis of input video frames.

In the *third section*, features of implementation of the developed software tools for implementation of the proposed approach are presented.

The *fourth section* presents the test results of the developed software.

The *conclusions* are the results of the work.

The work is presented on 84 sheets, contains a link to the list of used literary sources.

Keywords: video surveillance system, moving object, video material.

ЗМІСТ

СПИСОК ТЕРМІНІВ, СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

Байт-код – машинно-незалежний код низького рівня, що генерується транслятором і виконуваний інтерпретатором.

Бібліотека — збірка об'єктів чи підпрограм для вирішення близьких за тематикою задач.

Відеопотік — це часова послідовність кадрів певного формату, закодована у бітовий потік.

Відеореєстратор — пристрій, призначений для запису, зберігання та відтворення відеоінформації.

Відеофайл – файл, що зберігає у собі відеоматеріали.

Віджет – клас допоміжних міні-програм – графічних модулів, які розміщуються в робочому просторі відповідної батьківської програми і служать для прикраси робочого простору.

Інтерпретатор – програма чи технічні засоби, необхідні для виконання інших програм, вид транслятора, який здійснює пооператорну (покомандну, построкову) обробку, перетворення у машинні коди та виконання програми або запиту.

Інтерфейс — сукупність засобів, методів і правил взаємодії (управління, контролю і т. д.) між елементами системи.

Кадр – один з багатьох фотографічних зображень у кінофільмі.

Клас (ООП) — це спеціальна конструкція, яка використовується для групування пов'язаних змінних та функцій.

Кодек — пристрій або програма, здатна виконувати перетворення потоку даних або сигналу.

Метод (ООП) — підпрограма (процедура, функція), що використовується виключно разом із класом (методи класу) або з об'єктом (методи екземпляра).

Модуль — функціонально завершений фрагмент програми, оформлений у вигляді окремого файлу з сирцевим кодом або його іменованої частини, призначений для використання в інших програмах.

Системи відеоспостереження, системи стеження — це система що складається з відеокамер та пристрою з обробки відеоінформації, куди зводяться сигнали від усіх відеокамер в системі. Пристроєм обробки відеоінформації зазвичай є відеореєстратор.

HDTV (англ. High-definition television) — система трансляції цифрового телебачення з роздільною здатністю вищою ніж аналогові системи телебачення.

ВСТУП

Їдучи в довгоочікувану відпустку, кожен із нас намагається забути про всі побутові проблеми, щоб спокійно насолодитися відпочинком і набратися сил. Але, на жаль, повністю позбутися турбот не вдається. Багатьом не дає спокою думка про те, що раптом хтось незаконно потрапить за їх відсутності в квартиру або будинок. Основними засобами підвищення рівня безпеки на даний момент залишаються системи відеоспостереження – це програмно-апаратні комплекси, що встановлюються в будинках, на терміналах самообслуговування, поштоматах, банкоматах, смарт-сейфах та в інших важливих місцях. Сучасні системи відеоспостереження відіграють важливу роль у підвищенні рівня безпеки та дотримання прав приватної власності об'єкта під охороною. Щодня компанії, що працюють у сфері систем охорони, розробляють нові технології, які здатні ефективно усувати несприятливі ситуації та підвищувати рівень безпеки.

Комплексна інтегрована система безпеки може складатися з:

- Системи відеоспостереження.
- Системи контролю доступу, яка дозволяє вести облік робочого часу співробітників, отримувати інформацію про зональне місцезнаходження персоналу.
- Системи автоматичного розпізнавання номерів на в'їзді та місцях паркування, яка надає статистику відвідувачів, здійснює допуск на парковку за перепустками.
- Систем охоронної, пожежної сигналізації, пожежогасіння. Виконання охоронних функцій найефективніше за допомогою детекторів руху відеокамер та охоронних датчиків. Оператор негайно інформується системою про настання небажаної події в конкретній зоні.

Інтегровані системи безпеки можуть бути побудовані на територіально віддалених об'єктах, але мати загальний центр управління.

Системи відеоспостереження забезпечують фіксацію відеопотоку та серій фотознімків для розслідування спірних ситуацій, які виникають під час роботи пристроїв (махінації з викраденими картками, шахрайства під час видачі готівки), а також під час розслідувань актів вандалізму та пограбування.

Незважаючи на те, що традиційні системи відеоспостереження є досить поширеними в сучасних охоронних системах та демонструють високий рівень захисту, вони мають певні недоліки. Основним недоліком таких систем є високі затрати пам'яті на збереження зафіксованого відеоматеріалу, оскільки подібні системи постійно фіксують ситуацію на закріпленій території та зберігають велику кількість кадрів, навіть коли в місці спостереження нічого не відбувається. Це також приводить до збільшення часу, що витрачається на перегляд та аналіз збережених матеріалів.

Тому, щоб підвищити ефективність систем відеоспостереження, потрібно позбутися наведеної проблеми, що і стало предметом даного дослідження.

1. АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ТА ОБҐРУНТУВАННЯ ТЕМИ МАГІСТЕРСЬКОЇ ДИСЕРТАЦІЇ

Систем захисту приміщень від несанкціонованого вторгнення розроблено досить багато. Серед них є досить дорогі і цілком «демократичні», вони можуть захищати як невеликий кабінет чи квартиру, так і величезні площі приватних особняків з прилеглими до них територіями.

Самими простими системами є системи охоронної сигналізації, що включають в себе датчики руху, тепла, що реагують на звук розбитого скла. Також досить поширеними є камери відеоспостереження. Датчики і камери, як правило, підключаються до приймально-контрольного приладу, який посилає сигнали тривоги на пульт охоронної фірми або безпосередньо на телефон господарів будинку. Також він може включати сирени тривоги, блокувати замки на вхідних дверях.

Варіанти більш дорогих охоронних систем можуть вберегти приміщення не тільки від грабіжників, але і від неприємних наслідків пожеж, витоку газу або протікання водопровідних труб. Для цього в шлейфи системи окрім власне охоронних датчиків підключаються також датчики диму, газоаналізатори та інші датчики. Камери в таких системах теж досить дорогі, оснащені інфрачервоним підсвічуванням та широким кутом огляду. При цьому отримана інформація зберігається на накопичувачі на протязі певного часу.

1.1. Існуючі варіанти систем сигналізації

Система сигналізації - це основа вашої системи безпеки. Професійні пристрої сигналізації дозволяють моментально відреагувати на вторгнення у ваш будинок, при цьому спрацює світло-звукова сирена, яка сповістить сусідів про вторгнення. Також система зробить сповіщення за допомогою дзвінка або СМС на ваш телефон або в службу позавідомчої охорони. Не варто економити на серці цієї системи - керуючому приладі, так як якісна система не дасть збій в самий невідповідний момент. До того ж до

професійного обладнання для сигналізації можна підключити такі пристрої [15]:

- Датчик руху - виявляє рух в приміщенні і відправляє сигнал тривоги на керуючий прилад (при певній настройці не реагує на домашніх тварин)
- Датчик розбиття скла - віброакустичний датчик, який виявляє розбиття вікна або скляних дверей і відправляє сигнал тривоги на керуючий прилад.
- Датчик відкриття дверей - реєструє злом і проникнення в приміщення через двері.
- Датчик затоплення - реєструє витік води і затоплення приміщення
- Датчик газу - реєструє витік газу і відправляє сигнал тривоги на керуючий пристрій
- Датчик диму - відправляє сигнал тривоги на керуючий пристрій при задимленні приміщення внаслідок пожежі.
- Інфрачервоні бар'єри - встановлюються по периметру прилеглої до будинку території, дозволяють виявити зловмисника, коли він тільки перебирається через паркан.
- Система відеоспостереження - припускає установку камер на території вашого приватного будинку. На вулиці встановлюються вуличні антивандальні камери з інфрачервоним підсвічуванням, які дозволяють вам бачити те, що відбувається навколо будинку навіть вночі. Запис відео здійснюється на відеореєстратор (у випадку встановлення аналогової системи) або на відеосервер (у випадку встановлення цифрової системи з IP-камерами). Система відеоспостереження дозволяє не тільки відлякати передбачуваних зловмисників, але і їх ідентифікувати, що б допомогти правоохоронним органам оперативно їх знайти. Також сучасні системи відеоспостереження дозволяють переглядати відео з камери

віддалено через інтернет з будь-якої точки світу на вашому комп'ютері, телефоні чи планшеті.

Варіант організації системи захисту для звичайного будинку зображений на рисунку 1:

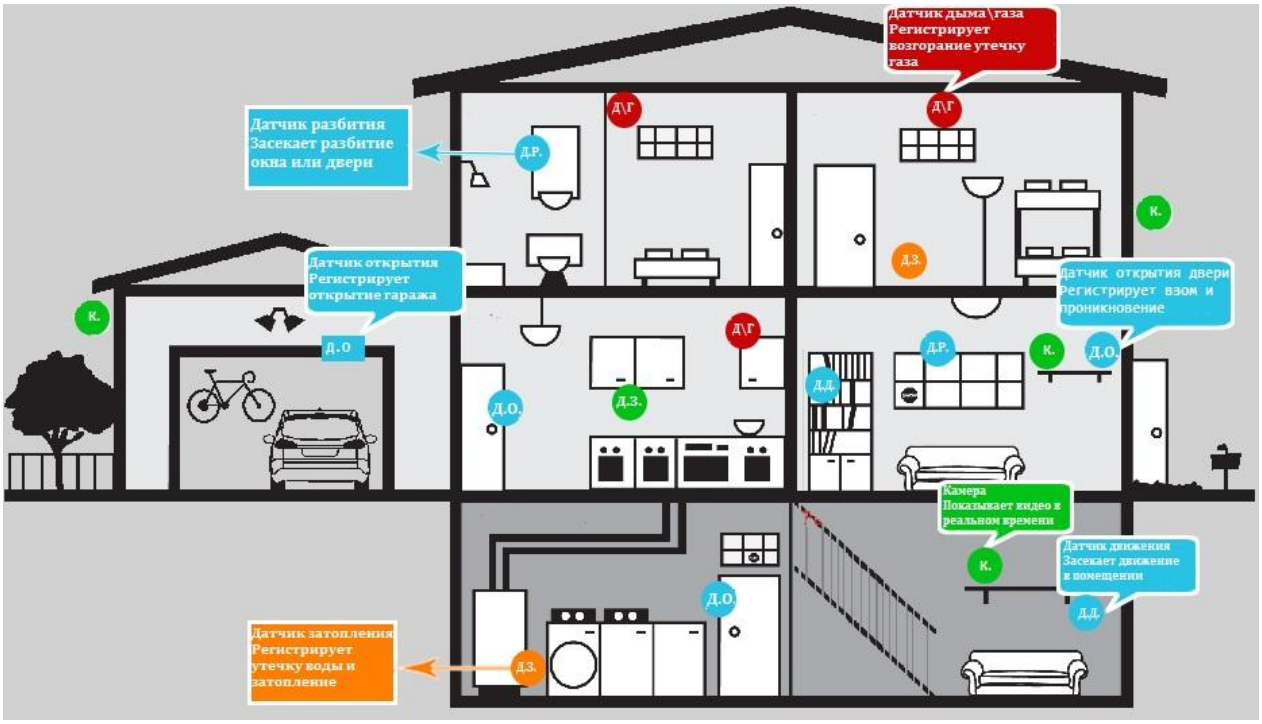

Рисунок 1 – Варіант організації системи захисту

У даній магістерській дисертації будуть детально розглядатися саме системи відеоспостереження. Саме ці системи в сучасному світі є найбільш популярними та ефективними, оскільки за допомогою систем відеонагляду і вдається встановити факт правопорушення та притягнути зловмисників до відповідальності. З іншого боку подібні системи проявляють до себе велику увагу розробників цих систем, оскільки кожен виробник прагне розробити досконалішу систему, яка водночас буде ефективною і економною як для виробника, так і для користувача.

1.2. Існуючі варіанти систем відеоспостереження

Відеоспостере́ження (англ. Video surveillance) — система передавання інформації з відеокамер, телевізійних камер на обмежену кількість моніторів та/або записувальних пристроїв.

Відмінність систем відеоспостереження від телевізійного мовлення полягає у тому, що сигнал не передається у відкритому режимі. Системи відеоспостереження часто використовуються для спостереження у місцях, які потребують постійного нагляду, таких як банки, банкомати, казино, вокзали, аеропорти, військові об'єкти та подібні місця.

На промислових об'єктах камери спостереження можуть використовуватись для централізованого стеження за виробничим процесом. Системи відеоспостереження можуть знімати безперервно. Досконаліші системи стеження, з використанням відеореєстраторів, дозволяють створювати записи, які зберігатимуться роками, з різною якістю та з додатковими можливостями (такими як виявлення рухів та оповіщення через електронну пошту).

Система відеонагляду — це система що складається з відеокамер та пристрою з обробки відеоінформації, куди зводяться сигнали від усіх відеокамер в системі. Пристроєм обробки відеоінформації зазвичай є відеореєстратор.[7]

На даний час існують наступні типи систем відеонагляду:

- аналогові (максимальна якість картинки 900 твл. (телевізійних ліній))
- HD системи (1080p)
- IP системи максимальна якість 20Mp

1.2.1. Аналогові системи відеонагляду

Аналогова система відеоспостереження – комплекс технічних засобів охоронного відеоспостереження, спрямованих на забезпечення безпеки на

об’єкті під охороною та здійснення належного нагляду з відеозаписом. Принцип дії аналогової відеокамери полягає у формуванні відеосигналу зі світлового потоку, який прямує через лінзи об’єктиву та потрапляє на матрицю охоронного відеопристрою. Аналогові системи використовуються, як правило, на невеликих підприємствах.[8]

Аналогова система відеоспостереження широко поширена завдяки ряду переваг:

- простота користування;
- висока надійність;
- стабільне та безперебійне функціонування обладнання;
- конкурентоспроможна ціна.

Серед недоліків аналогових систем можна назвати обмеженість їх функцій та потреба у постійному обслуговуванні (заміні та архівації відеокасет, очищенні та заміні відеоголовок). Тому все частіше аналогові системи відеонагляду поступаються цифровим, які мають спрощену схему обробки та зберігання відеозаписів.

Аналогові системи відеоспостереження працюють у телевізійному стандарті PAL (англ. Phase Alternating Line — порядкова зміна фази) — система аналогового кольорового телебачення, розроблена інженером німецької компанії «Telefunken» Вальтером Брухом і прийнята в якості стандарту телевізійного мовлення в 1966 році у Німеччині, Великобританії та ряді інших країн Західної Європи.

У даний час система PAL є найпоширенішою у світі. Система PAL у більшості випадків використовується у поєднанні з європейським стандартом розкладання 576i. Стандарт 576i — це стандарт розкладання, який прийнятий для аналогового і цифрового телебачення стандартної чіткості SDTV у Росії, Європі, Австралії, частини країн Азії, Африки та Південної Америки. При повному числі рядків рівному 625, у побудові зображення беруть участь тільки 576. Тому у цифровому телебаченні цей стандарт позначається 576i.

Буква «і» означає черезрядкове розгорнення, яке передає 25 цілих кадрів у 50 полях за секунду.[8]

Під аналоговим відеоспостереженням маються увазі старі камери, які працювали з роздільною здатністю CIF (360x240 співвідношення сторін 4:3), що за кількістю рядків навіть менше, ніж було закладено у стандарт PAL. Камери з роздільною здатністю CIF сьогодні практично не продаються.

Роздільна здатність деяких стандартних зображень показана на рисунку 2:

Рисунок 2 – Варіанти стандартних розмірів зображень

Повноцінна реалізація у 576 рядків була реалізована у стандарті D1 (576x720 співвідношення сторін 4:3), але поширення широкоформатних моніторів і поява нових матриць стимулювало появу нових форматів у аналоговому відеоспостереженні. Більшість сучасних аналогових камер працюють у стандарті 960H, роздільна здатність 576x960 і співвідношення сторін 16x10. Камери у цьому стандарті мають широку площу огляду у порівнянні з D1. Зображення не спотворюється на широкоформатних моніторах.

Попри низькі показники характеристик аналогових систем відеоспостереження, вони як і раніше повсюдно застосовуються у невеликих системах при спостереженні у приватному секторі, невеликих магазинах, кіосках і скрізь, де пріоритетом є низька вартість обладнання.

Прилади аналогового типу — це камери, які передають дані (інформацію) за допомогою аналогового сигналу. Пристрій використовується в комплексі з цілою системою відеоспостереження: відеореєстратором, монітором, мультиплексором.[9] Схема побудови такої системи відеонагляду представлена на рисунку 3:

Рисунок 3 – Схема побудови аналогової системи відеоспостереження

Порядок роботи:

- Потік світла, що проходить через лінзи, потрапляє на матрицю.
- Генерується відеосигнал.
- Через кабель сигнал надходить на відеореєстратор.
- Дані відображаються на моніторі.

Перевагою аналогових камер відеоспостереження є:

- Взаємна сумісність пристроїв незважаючи на виробництво від різних компаній.
- Процес монтажу досить легкий.
- Простота налаштувань, через передбачене меню в апараті.

- Пристрій не пропускає жодної секунди відео в процесі запису. Фіксується абсолютно все.
- У комплексі з апаратом можна встановити мікрофон.
- Низька собівартість.
- Великий вибір аналогової відео-оптики.

Недоліки аналогової відео-оптики:

- Рівень захисту від стороннього втручання — низький, тобто відсутній принцип шифрування.
- При впливі з іншими кабелями при монтажі — спостерігаються перешкоди.
- Відео не управляється/проглядається через Інтернет.
- Якість дозволу — низька (при деталізованій зйомці предмети не розглядаються — вони розмиті).
- При використанні з мікрофоном, необхідно проводити окремий кабель для передачі аудіофайлів.
- Неможливо відтворити дані, отримані камерою, на ПК. Це здійснюється тільки із застосуванням додаткових приладів.
- Не має режимів цифрового збільшення, відсутня управління рухом через один і той же підключений кабель, виключається робота в комплексі з детектором руху.
- При монтажі цього виду камер, слід передбачити резервне джерело живлення, врахувати відстань від інших кабелів, встановити відеореєстратор або допоміжний пристрій для перегляду на ПК.

1.2.2. HD системи відеонагляду

HD відеоспостереження - це відеоспостереження високої роздільної здатності у форматі Full HD. І на сьогоднішній день відеоспостереження Full HD є максимально можливим стандартом передачі відео високої роздільної

здатності. HD відеоспостереження дозволяє отримувати зображення з камер спостереження в розмірі 1920x1080.

Системи відеоспостереження HD використовують абсолютно новий стандарт передачі відеосигналів HD SDI (High-Definition Serial Digital Interface), при цьому використовуючи звичний для всіх користувачів коаксіальний кабель. Якщо розглядати сам стандарт HD-SDI, то з'явився він спочатку у сфері телебачення високої чіткості HDTV. HD-SDI був стандартизований Товариством інженерів кіно і телебачення (SMPTE) для забезпечення передачу відео потоку даних по коаксіальному кабелю. Успішне використання стандарту HD-SDI в телебаченні, спричинило створення можливості застосування цього стандарту і для охоронного відеоспостереження за об'єктами.[13] Компонентна схема зображена на рисунку 4:

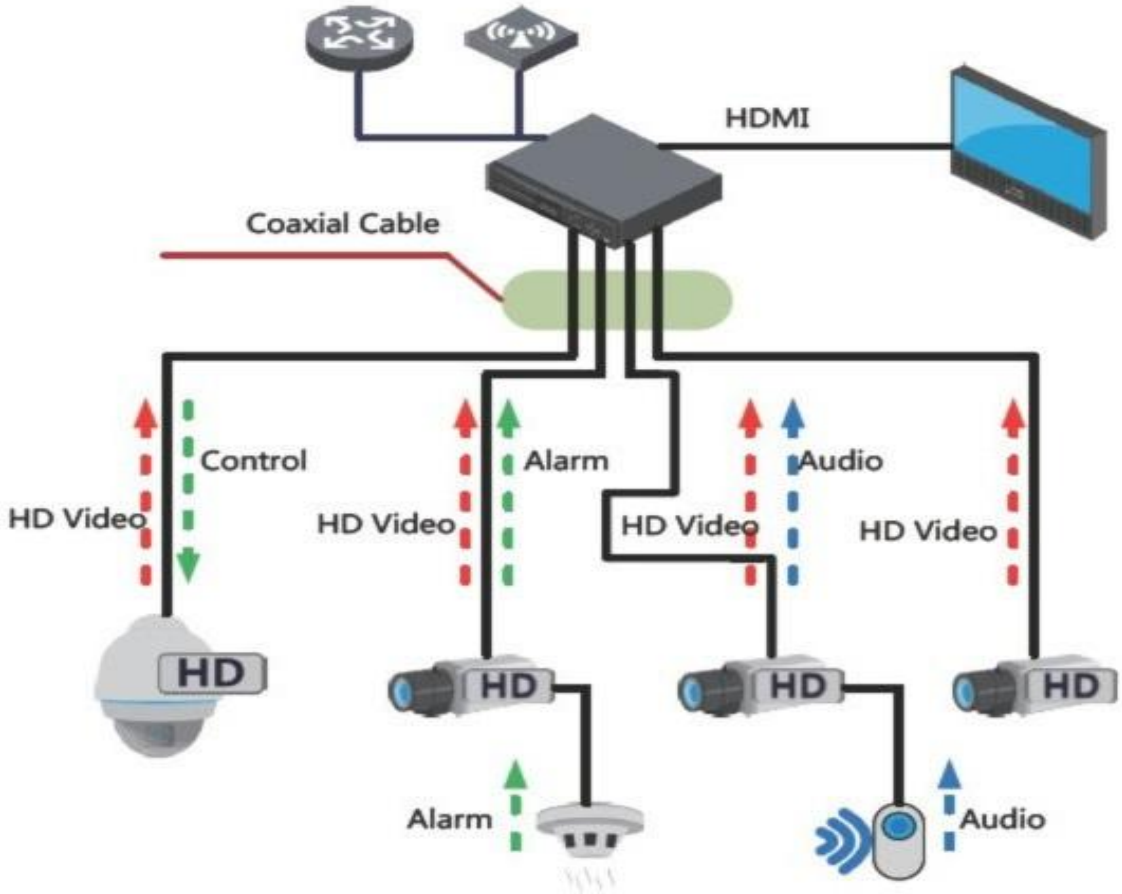

Рисунок 4 – Схема побудови HD системи відеспостереження

У HD відеоспостереженні обробка відеозображення здійснюється за допомогою сучасного кодека H.264, який є одним з варіантів кодека MPEG4. І якщо подивитися на це більш глобально, то як користувачі системи ви можете отримати безперечну перевагу в тому, що з використанням сучасних камер можна охопити на багато більше спостережуваного простору, ніж при використанні аналогових камер відеоспостереження.

Це найбільш очевидно в створенні охоронного відеоспостереження на невеликих об'єктах. Так якщо вам потрібно організувати спостереження в офісі, будинку, невеликому магазині або кафе, то достатньо буде 2-3 камер відеоспостереження, щоб охопити весь об'єкт. Що вже говорити про об'єкти з великою територією, де HD відеоспостереження може в рази зменшити кількість встановлених камер і забезпечити користувачів охоронної системи деталізованим чітким зображенням.[12]

Якщо розглядати охоронне відеоспостереження стандарту HD-SDI, то тут можна чітко виділити два різних формати:

- HD Ready 720p (1280x720 пікселів, відношення сторін зображення 16:9), забезпечується камерами з 1,3-мегапіксельною роздільною здатністю.
- Full HD 1080p (1920x1080 пікселів, відношення сторін зображення 16:9), забезпечується камерами з 2-мегапіксельною роздільною здатністю.

Визначимо головні переваги HD відеоспостереження:

1. Вартість камер HD істотно нижче аналогічних IP відеокамер, що істотно відображається на підсумковій ціні всієї системи відеоспостереження.

2. Витрати на встановлення системи відеоспостереження HD менше, ніж на монтаж IP відеоспостереження, оскільки немає необхідності залучати висококваліфікованих IT-фахівців для налаштування мережі та мережевого обладнання.

3. При передачі зображення не піддається ніякому впливу і немає шумів.

4. Немає ніякої тимчасової затримки при перегляді відео з камер.

5. Якість запису дуже висока, так як відеозображення надходить на монітор або пристрій відеореєстрації в чистому стислому вигляді. Таке зображення можна масштабувати без втрати якості.

6. Дуже проста організація переобладнання аналогової системи відеоспостереження і перехід до HD систем відеоспостереження. Для цього не потрібно ніяких глобальних змін системи, всі комунікації залишаються ті ж самі.

7. На відміну від аналогових відеокамер, HD відеокамери охоплюють в 3-4 рази більше спостережуваного простору, що дозволяє істотно економити на кількості камер на території.

8. Застосування стандарту HD-SDI в відеоспостереженні являє собою дуже просте рішення з легкими інструментами установки. Адже впровадити HD систему відеонагляду можна на будь-який об'єкт, де прокладений коаксіальний кабель.

9. Живлення самих камер відеоспостереження можна з легкістю організувати по коаксіальному кабелю.

10. При функціонуванні HD відеоспостереження з використанням коаксіального кабелю, на передачу сигналів від камер не впливає якість інтернет з'єднання і навантаження мережі, як у випадку з IP відеоспостереженням.

11. Якщо система відеоспостереження складається з роботизованих поворотних камер, то саме HD відеоспостереження забезпечить можливість передачі сигналів управління камерам.

12. Якщо система відеоспостереження вимагає можливості запису аудіозвуку і передачу його, то це легко можна організувати по коаксіальному кабелю.

13. Камери HD-SDI формують 2-мегапіксельне зображення Full HD, що можна розглядати як обмеження, але якщо подивитися на це поглядом професіоналів, то можна точно стверджувати, що такої роздільної здатності зображення більш ніж достатньо, щоб отримати всі деталі того, що відбувається на об'єкті.

Дійсно переваг у HD відеоспостереження не мало і саме завдяки їм цей тип відеоспостереження і займає свою частку на ринку безпеки, нехай зовсім не велику, але число користувачів, які зробили свій вибір на установці саме HD систем відеоспостереження постійно зростає.[13]

На даний момент аналітики ринку відеоспостереження не прогнозують бурхливого зростання популярності і затребуваності HD відеоспостереження. Адже такий прорив як IP відеоспостереження воно не зможе повторити. У свою чергу HD системи відеонагляду – це хороший варіант для оновлення існуючих аналогових систем відеоспостереження.

Таким чином стає очевидно, що HD відеоспостереження - це прогресивний крок вперед на ринку сучасного відеоспостереження. HD відеоспостереження вдихнуло нове життя в застаріле аналогове відеоспостереження, дозволивши його удосконалити і оновити. Так якщо користувачі систем відеоспостереження хочуть перейти з низькоякісного аналогового зображення на цифрове, то ідеальним варіантом буде організація HD відеоспостереження, що дозволить використовувати ту ж саму інфраструктуру системи і не змінювати існуючу систему кабелів.

Але варто відзначити, що HD відеоспостереження має тільки два стандарти передачі даних: HD 720P і HD 1080P, що відповідає HD роздільній здатності і Full HD роздільній здатності. І на даний момент немає ніяких перспектив подальшого збільшення роздільної здатності зображення в рамках стандарту HD-SDI.[14]

1.2.3. IP системи відеонагляду

У сучасному світі Internet Protocol (IP) є міжмережевим протоколом, який зробив можливим об'єднання всіх окремих підмереж у всесвітню мережу Інтернет. І звичайно обов'язковою частиною існування Internet Protocol (IP) є адресація мереж - тобто IP-адресу кожного терміналу мережі, а в нашому випадку кожної мережевої відеокамери в системі спостереження.[10]

Так якщо формат IP вже давно став стандартом передачі даних в комунікаційних мережах, то звичайно це не обійшло стороною і сферу відеоспостереження. Так саме завдяки Internet Protocol відбувається об'єднання камер в єдину мережу і стає можливою доставка відеоданих між будь-якими вузлами мережі.

Таким чином, стає очевидним, що мережеві IP-відеокамери, завдяки своїй індивідуальній IP-адресі, дозволяють записувати і переглядати відеодані в режимі реального часу (он-лайн) з будь-якого місця на планеті, де є доступ до мережі Інтернет. [11]

І звичайно зі стрімким масовим розвитком Інтернету по всьому світу IP відеоспостереження стає з кожним днем все більш затребуваним і поширеним. Так на сьогоднішній день мережеве IP відеоспостереження займає ключове місце на ринку систем безпеки.

Більше того ще однією особливістю IP відеоспостереження є передача у всесвітню мережу вже оцифрованого відеопотоку даних. Саме оцифрований відеопотік з високою якістю зображень і відрізняє IP відеоспостереження від HD систем відеоспостереження або аналогового відеоспостереження.

Фактично користувачі системи мережевого відеоспостереження стають володарями цифрової системи охоронного відеоспостереження, яка використовує для аналізу та обробки даних новітні інформаційні технології.

IP відеоспостереження передбачає насамперед використання IP камер і може працювати без відеореєстратора. IP камера знімає відео і передає дані

по локальній мережі. Камери мають мікрофони, сигнал тривоги, нічне підсвічування, аудіовхід та аудіовихід, веб-сервер. IP камери налаштовуються через браузер. Багато IP камер мають також можливість підключення за допомогою WiFi, що дає можливість їх встановлення у важкодоступних місцях. Зберігати інформацію можна на жорстких дисках або за допомогою хмарних технологій.

IP відеоспостереження може бути двох видів:

- Провідне відеоспостереження
- Безпровідне відеоспостереження

Класичне IP відеоспостереження складається з пристроїв, що функціонують на базі стандартної мережевої архітектури - локальної мережі Ethernet. Локальна мережа Ethernet - це свого роду стандарт організації локальних обчислювальних систем, який використовується для об'єднання пристроїв, що знаходяться поблизу один від одного (один будинок, група офісів, група будинків). Структурна схема IP системи відеоспостереження представлена на зображенні 5:

Рисунок 5 – Схема побудови IP системи відеонагляду

Безпровідне IP відеоспостереження користується великою популярністю за рахунок масового використання технологій передачі даних

Wi-Fi, Bluetooth і Wi-MAX. Все, що користувачам потрібно для використання безпровідного мережевого відеоспостереження - це наявність на об'єкті покриття безпровідного інтернету Wi-Fi. IP відеокамери можна підключати безпосередньо через модем або адаптер. Також IP відеоспостереження можна реалізувати використовуючи мережі мобільного зв'язку.

Сучасний функціонал всесвітньої мережі інтернет дозволяє використовувати відкриті стандарти мережі і таким чином мати доступ до локальної мережі в будь-якій точці світу при цьому перебувати також в будь-якому місці, де є можливість підключення до мережі.

Перевагами IP відеоспостереження є:

- Перегляд on-line з будь-якого пристрою
- Висока якість зйомки
- Віддалений доступ
- Можливість бездротового підключення (WiFi)
- Можливість подачі живлення через Інтернет-кабель (технологія PoE)

Серед недоліків — ціна. IP відеоспостереження перевищує ціну на аналогове та HD відеоспостереження. Також IP камери не варто використовувати у тих випадках, де потрібне зображення у реальному часі. IP камери хоч і забезпечують належну якість картинки, але все ж не можуть забезпечити безперервність зображення.

Охоронне IP відеоспостереження поєднує в собі всі переваги всесвітньої комп'ютерної мережі інтернет на базі IP протоколу і мережевих камер з чітким цифровим зображенням. Якщо раніше охоронне відеоспостереження обмежувалося лише одним стаціонарним комп'ютером на пульті охорони і відповідно перегляд відео на ньому, то сучасне об'єднання охоронного відеоспостереження з локальною мережею - це справжній прорив в цій області. Така система надає можливість доступу до відеоданих будь-де і отримувати відео з будь-якої камери.

1.3. Обґрунтування теми магістерської дисертації

Усі розглянуті вище системи відеоспостереження працюють за таким спрощеним алгоритмом:

1. Отримується інформація з камер відеонагляду і передається каналами зв’язку на пристрій, який буде оброблювати даний відеопотік (відеореєстратори, сервери).
2. Пристрій обробки відеопотоку організовує запис відеоматеріалу на встановлені пристрої збереження інформації та передають потік на пристрої виведення інформації.
3. Інформація, що надходить з відеореєстратора, відображається на моніторах у реальному часі. Це дозволяє миттєво реагувати на надзвичайні ситуації.

Головним недоліком такого підходу є організація постійного запису відеоматеріалу навіть у випадках, коли на території, що знаходиться під контролем, не відбувається ніяких змін. Це спостерігається у малолюдних важливих місцях, за якими потрібний постійний нагляд. Такими місцями можуть бути сховища банків, перепускні пункти на деяку територію, що знаходиться під охороною та інші подібні місця. Системи відеонагляду, що встановлені в таких місцях, будуть постійно записувати дані, навіть при відсутності змін на місцевості під контролем. Це призводить до виникнення наступних проблем:

1. Процес зберігання відеоматеріалу потребуватиме великого об’єму пам’яті, яка витрачатиметься на збереження непотрібних даних.
2. Аналіз збережених матеріалів займатиме більше часу, оскільки такі дані можуть містити багато повторюваної інформації, які і будуть перешкоджати швидкому перегляду відео.

Тому метою даної роботи стало створення нового підходу, який зможе аналізувати відеопотік та, залежно від інформативності відео, фільтрувати дані безпосередньо перед записом на носій інформації. Таким чином

кількість непотрібних даних значно зменшиться, що позитивно вплине на розмір збереженого матеріалу та швидкість аналізування матеріалу в майбутньому.

2. ОПИС ЗАПРОПОНОВАНОГО ПІДХОДУ

Для вирішення поставленої задачі розроблювана система повинна мати наступні можливості:

- Система має аналізувати потік, що надходить з відеокамери.
- Якщо в потоці відео виявлено рух будь-яких об’єктів, система організовує запис відеофайлу. Інші матеріали ігноруються.
- Система відеоспостереження також мусить контролювати час, оскільки коли відбуваються несанкціоновані дії, то для встановлення всіх обставин важливо знати точну дату і час, коли зафіксована подія відбулась.
- Також всі вище описані дії мають відбуватись максимально швидко, щоб система вчасно реагувала на зміни у відеопотоці. Якщо ця умова не буде дотримана, тоді така система відеонагляду вважатиметься не ефективною, оскільки можлива втрата важливої інформації, що недопустимо для подібних систем.

Таким чином вище описані можливості і складають основні вимоги до розроблюваної системи. Задачу розпізнавання руху можна вирішити двома способами:

1. Додати до системи відеоспостереження датчик руху, згідно показань якого і розпочинати запис матеріалів.
2. Розробити програмні засоби, які і будуть контролювати запис відеофайлів.

Перший спосіб не є вдалим через наступні причини:

- Додаткове апаратне забезпечення потребує певних фінансових витрат.
- При встановленні датчика руху, коректність роботи системи буде залежати від додаткового компоненту. Більш того, якщо датчик руху вийде зі строю, є шанси, що система відеоспостереження

цього навіть не помітить. Тоді система буде очікувати команди «запис» від датчика, який буде не в змозі віддати її, тим самим ігноруючи важливі події, які система відеоспостереження мусить зафіксувати.

- Датчик руху має можливість не реагувати на невеликі об’єкти, наприклад тварин, птахів. Для систем освітлення ця можливість буде надзвичайно корисною. Але в системах відеонагляду це може навпаки негативно вплинути, оскільки навіть невеликі об’єкти можуть бути потенційно небезпечними.

Саме тому для вирішення поставлених проблем було прийняте рішення розроблювати саме програмні засоби для систем відеоспостереження. Це має декілька переваг, порівняно з першим способом:

- Непотрібно зайвих витрат на додаткове апаратне устаткування.
- Система відеоспостереження в такому випадку не потребує модифікації. Оскільки це програмні засоби, то вони можуть бути встановлені безпосередньо на керуючі пристрої, тим самим модифікуючи роботу вже існуючої системи. Таким чином в результаті ми отримуємо ті ж самі апаратні комплекси, а поставлені проблеми вирішуватимуться програмно.
- Виробники систем захисту матимуть змогу пропонувати клієнтам більш досконалий продукт, не змінюючи апаратну складову такої системи. Отже клієнти матимуть змогу отримувати більш функціональну систему відеоспостереження за відносно невелику переплату.

Розроблювана система базуватиметься на алгоритмах порівняння зображень. Такий спосіб дозволяє якнайшвидше визначити рух об’єктів на підконтрольній території, що, згідно вимог до даної системи, є першочергово важливим. Скорочений алгоритм роботи системи відеонагляду (детально розглядається в п. 2.2) виглядатиме наступним чином:

1. Зчитуються дані з відеокамер.
2. Система покадрово аналізує дані, якщо виникли зміни – потрібно організувати запис матеріалу. Саме цей факт і означає, що на місцевості під контролем відбулися деякі зміни.
3. Якщо зміни в кадрах зникли, це означає, що змін далі не відбувається і записувати відеоматеріал далі не має сенсу.

Отже для подальшої роботи над самим алгоритмом роботи системи, нам потрібно визначити, який з алгоритмів порівняння зображень нам необхідно обрати.

2.1. Існуючі алгоритми порівняння зображень

Варіантів алгоритмів порівняння існує дуже багато. Саме тому з цієї великої кількості потрібно обрати декілька рішень, які найбільше підходять до вирішення поставлених проблем. Перш за все, головна вимога до алгоритмів це швидкодія, оскільки алгоритм працюватиме з потоком кадрів і швидкість роботи має найбільший вплив на розроблювану систему. Також слід врахувати, що камера майбутньої системи буде знімати кадри з однаковою розмірністю, саме тому алгоритми, які порівнюють зображення з різною розмірністю можна виключити.

Існують варіанти алгоритмів, які можуть розпізнавати об’єкти на зображенні. Безумовно така функція була б корисною в системах відеонагляду. Але такі алгоритми потребують більше часу на розпізнавання образів та потребують швидкісного підключення до мережі Інтернет, що не завжди доступно в місцях встановлення систем відеонагляду. Через ці причини подібні алгоритми також будуть не доречними в розробці системи відеонагляду з розпізнаванням руху об’єктів.

Згідно цих вимог, найбільш доречними будуть саме ці алгоритми:

- Алгоритм MSE
- Алгоритм SSIM

- Алгоритм NRMSE

2.1.1. Алгоритм MSE

В статистиці середня квадратична помилка (MSE) або середнє квадратичне відхилення (MSD) оцінювача (процедура оцінки необмеженої кількості) вимірює середнє значення квадратів помилок, тобто середню квадратичну різницю між оціночними властивостями і тим, що оцінюється. MSE - це функція ризику, яка відповідає очікуваній величині втрати квадрату помилки. Той факт, що результат MSE практично завжди суворо позитивний (а не нульовий), обумовлений тим, що оцінювач не враховує інформацію, яка могла б дати більш точні оцінки. Саме тому значення, близькі до нуля, краще.

MSE - це друга момента (про походження) помилки і, таким чином, включає як дисперсію оцінювача (як широко поширюються оцінки від однієї вибірки даних до іншого), так і його зміщення (як далеко від середньої оціночної вартості від істини) Для об'єктивної оцінки, MSE є дисперсія оцінювача. Як і дисперсія, MSE має ті ж одиниці виміру, що й квадрат оцінюваної кількості. За аналогією зі стандартним відхиленням, прийняття квадратного коріння з MSE дає середньоквадратичну похибку або середньоквадратичне відхилення (RMSE або RMSD), що має ті ж одиниці, що і оцінювана величина; для об'єктивної оцінки, RMSE - квадратний корінь дисперсії, відома як стандартна помилка.

MSE оцінює якість оцінювача (тобто, математичну функцію, яка відбиває зразок даних на оцінку параметра популяції, з якої дані відбираються) або предиката (тобто функція, яка відображає довільні входи до зразка значення деякої випадкової величини). Визначення MSE відрізняється залежно від того, як описується вона – оцінкою чи предикатом.

Якщо Y є вектором з n векторних передбачень згенерованих з вибірки n точок даних всіх змінних, і Y' являє собою вектор спостережуваних значень

змінної, що прогнозується, то середня вибірка MSE предикату обчислюється формулою (1):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - Y_i')^2 \quad (1)$$

Тобто, MSE - це середнє значення квадратів помилок. Це легко обчислювана кількість для конкретного зразка (і, отже, залежить від вибірки).

Для кадрів X та Y розміром M×N пікселів значення функції MSE обчислюється формулою (2):

$$MSE = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}[X(m,n) - Y(m,n)]^2 \quad (2)$$

де X(i, j) — значення компоненти яскравості пікселя (i, j) зображення X.

Середня квадратична помилка досить ненадійна тому, що не відповідає системі візуального сприйняття людини (human visual system, HVS). Слід зазначити, що значення середньої квадратичної помилки може незначно змінюватися при істотному погіршенні якості вхідного зображення. Перевагами даного алгоритму є відносна простота реалізації і швидкість роботи [1].

2.1.2. Алгоритм SSIM

Індекс структурної подібності (SSIM) - це метод автоматичного прогнозування сприйманої якості зображень та відео. Це стало надзвичайно популярним за останні кілька років. У науковій галузі оригінальний документ SSIM, опублікований у 2004 році, отримав понад 35 000 цитат Google Scholar. Це можливо, більше, ніж будь-який документ у літературі відеотехніки. Він також отримав премію Best Paper Award for the IEEE Signal Processing, одну з найпрестижніших паперових нагород у обробці сигналів. У промисловості SSIM-алгоритм отримав нагороду Primetime Engineering

Emmy Award, одну з найпрестижніших нагород технологій у галузі телебачення.

Однією з основних причин є те, що існує дуже сильний попит на показники якості зображення, можливо, значно більший, ніж більшість людей усвідомлює. Хоча аргументація вище звучить, вона не пояснює, що особливо важливо для SSIM, тому що будь-який хороший об'єктивний захід якості виконуватиме цю роботу. Щоб зрозуміти це краще, ми повинні повернутися до перших років після нового тисячоліття, коли ідея SSIM була ініціалізована. У той час в області вже було здійснено значну роботу, але загальною точністю було те, що прогнозування візуального сприйняття якості зображення є надзвичайно складною проблемою. Щоб досягти поставленої мети, потрібно мати всебічне розуміння обчислювальних механізмів на візуальному шляху, до якого була присвячена велика література з психофізичного та фізіологічного бачення, але все одно можна зрозуміти дуже мало (навіть зараз). У інженерному світі обчислювальні моделі, що використовувались для оцінки якості відео, були настільки складними, що люди мали робити чіпи ASIC для виконання обчислень, а обладнання може коштувати 10 або 100 тис. Доларів, однак можна було б оцінити тільки невеликі зразки сегменти відео, а не в режимі реального часу. Більш того, навіть для цих складних візуальних моделей, багато людей все ще ставлять під сумнів, чи вони взагалі надають цінні візуальні прогнози якості. Хорошим прикладом може служити перший незалежний тест, проведений групою експертів з оцінки якості відео (VQEG) у 2000 році, де всі розширені моделі виконували аналогічно MSE / PSNR. У той час здавалося, що єдиний шлях до вдосконалення полягає в тому, щоб зробити моделі ще більш складними, з тим щоб більш точно зафіксувати більше візуальних функцій або зменшити проблему до конкретних програм, так що ряд об'єктивних моделей може бути розроблено, кожне націлювання на певний тип спотворень.

Зважаючи на наведене вище, стає набагато легше зрозуміти, чому SSIM здивував людей, коли вони були вперше опубліковані. По-перше, SSIM не будується для прямого впровадження будь-якої психологічної або фізіологічної моделі зору. Замість цього, це робить просте припущення про загальну функціональність візуальної системи, тобто для вилучення структурної інформації з візуальної сцені. Потім він намагається окремо фіксувати структурні та неструктурні спотворення, перед тим як розчісувати їх. До SSIM було зроблено дуже мало зусиль, щоб вирішити загальний принцип проектування моделей якості зображень, і більшість людей не вірили, що прогнозування якості зображення завжди можливо, незважаючи на те, як нейрони працюють. По-друге, формула SSIM цілком відрізняється від будь-якого методу оцінки якості зображень або будь-якої моделі біологічного зору на той час, і обчислення є простим і швидким, набагато швидшим, ніж найсучасніші підходи. По-третє, алгоритм SSIM (разом з його попередньою версією, універсальним індексом якості зображень) був представлений вражаючими демонстраціями, де зображення, які зазнають дуже різних типів спотворень, але мають однакове значення MSE / PSNE, мають різко відмінну візуальну якість та такі варіації якості добре прогнозується SSIM. По-четверте, незважаючи на свою простоту, SSIM дає набагато кращі прогнози якості зображення, ніж інші складні методи при тестуванні на базі суб'єктних баз даних зображень, доступних тоді. Все це робить SSIM особливим. Що більш важливо, з SSIM, ми раптом виявили, що нам набагато ближче до розгортання високо ефективних і високоефективних автоматизованих систем оцінки якості зображення в реальному світі.

Успіх SSIM зіграв важливу роль у стимулюванні великої кількості дослідницьких робіт з оцінки якості зображення протягом останніх десяти років. Як і SSIM, багато хто з нових запропонованих підходів не суворо дотримуються моделей біологічного бачення. Велика кількість дослідників з різноманітним фоном залучаються до поля, і все більше і більше учених присвячують свої роботи проблемам якості зображення. Таким чином,

різноманітність методологій проектування моделей оцінки якості зображення була значною мірою збагачена.

Індекс структурної подібності є одним з методів вимірювання схожості між двома зображеннями. SSIM-індекс це метод повного зіставлення, іншими словами, він проводить вимірювання якості на основі вихідного зображення. Відмінною особливістю методу є те, що метод враховує «сприйняття помилки» завдяки обліку структурної зміни інформації. Ідея полягає в тому, що пікселі мають сильний взаємозв'язок, особливо коли вони близькі просторово. Дані залежності несуть важливу інформацію про структуру об'єктів і про сцену в цілому[3]. Індекс структурної подібності обчислюється за такою формулою (2):

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \qquad (2)$$

в яку включені такі параметри, як розташування вікна N x N у кожному зображенні (x, y), середнє значення інтенсивності пікселів у напрямку x і y, дисперсія інтенсивностей у напрямку x і y, а також коваріація.

Варто зазначити, що алгоритм SSIM використовується для порівняння двох вікон (тобто невеликих частин зображення), а не всього зображення, як у MSE. Це призводить до більш надійного підходу, який може відображати зміни в структурі зображення[1].

2.1.3. Алгоритм NRMSE

Даний алгоритм є досить популярною модифікацією MSE. Алгоритм NRMSE (Normalized root-mean-square error) - це квадратичне правило підрахунку, яке також вимірює середню величину помилки та нормалізоване певним чином для покращення аналізу даних. З математичної точки зору це квадратний корінь з середніх квадратів помилок між зображеннями (3):

$$NRMSE = 100\frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(S_i - O_i)^2}}{nval} \quad (3)$$

де nval — тип нормалізації, що використовується.

Нормалізація NRMSE сприятиме кращому розумінню між наборами даних або моделями з різними шкалами. Хоча в літературі немає узгоджених засобів нормалізації, загальний вибір - це середнє або діапазон (визначається як максимальне значення мінус мінімальне значення) виміряних даних.

Слід також зазначити, що NRMSE є квадратним коренем з середнього числа квадратів помилок. Вплив кожної помилки на NRMSE пропорційний розміру квадрата помилки. Таким чином, великі помилки роблять непропорційно великий вплив на NRMSE, що підвищує точність оцінки зображень [2].

2.1.4. Порівняльна характеристика

Тестування алгоритмів проводилось на комп’ютері з такими характеристиками:

- процесор: Intel(R) Core(TM) i3-3110M;
- графічний процесор: Intel(R) HD Graphics 4000
- частота процесора: 2.4 ГГц;
- оперативна пам’ять: 8 ГБ;
- операційна система: Ubuntu 64-bit.

Алгоритми було реалізовано на мові програмування Python (детальніше у розділі 3.1). Тест проводився за двома параметрами:

- Швидкість роботи
- Отримуваний результат

Також досліджувались різні набори зображень, включаючи їх зміст (однакові чи різні) та їх розміри:

- 1920x1080 пікселів

- 640x480 пікселів
- 320x240 пікселів

У першому експерименті проводилось тестування на швидкість роботи алгоритмів. Для тесту було обрано різні набори зображень однакового змісту. Результати обчислювались як середній час виконання, що був затрачений на одну оцінку зображень. Результати продемонстровані на скріншотах 6 - 8:

```
/usr/bin/python3.6 /home/vetalik/PycharmProjects/test/testImages.py
Equvivalent pictures 1920x1080:
MSE:             time: 0.055070352554321286
SSIM:             time: 1.7659446239471435
NRMSE:             time: 0.06710920333862305
```

Рисунок 6 – Час порівняння ідентичних зображень розміром 1920x1080 пікселів

```
/usr/bin/python3.6 /home/vetalik/PycharmProjects/test/testImages.py
Equvivalent pictures 640x480:
MSE:            time: 0.0042152881622314455
SSIM:            time: 0.1563161849975586
NRMSE:            time: 0.0058157920837402345
```

Рисунок 7 – Час порівняння ідентичних зображень розміром 640x480 пікселів

```
/usr/bin/python3.6 /home/vetalik/PycharmProjects/test/testImages.py
Equvivalent pictures 320x240:
MSE:           time: 0.0009141921997070313
SSIM:           time: 0.03518552780151367
NRMSE:           time: 0.0013552665710449218
```

Рисунок 8 – Час порівняння ідентичних зображень розміром 320x240 пікселів

Далі проводився подібний експеримент, але для тестування було обрано зображення, що відрізняються за змістом. Результати роботи відображено на скріншотах 9 - 11:

```
/usr/bin/python3.6 /home/vetalik/PycharmProjects/test/testImages.py
Different pictures 1920x1080:
MSE:           time: 0.053900289535522464
SSIM:           time: 1.745074462890625
NRMSE:           time: 0.06721925735473633
```

Рисунок 9 – Час порівняння різних зображень розміром 1920x1080 пікселів

```
/usr/bin/python3.6 /home/vetalik/PycharmProjects/test/testImages.py
Different pictures 640x480:
MSE:            time: 0.003949165344238281
SSIM:            time: 0.15549592971801757
NRMSE:            time: 0.00580143928527832
```

Рисунок 10 – Час порівняння різних зображень розміром 640x480 пікселів

```
/usr/bin/python3.6 /home/vetalik/PycharmProjects/test/testImages.py
Different pictures 320x240:
MSE:            time: 0.0009769916534423829
SSIM:            time: 0.03351812362670899
NRMSE:            time: 0.0013461589813232421
```

Рисунок 11 – Час порівняння різних зображень розміром 320x240 пікселів

Отже після цих експериментів, можна зробити кілька висновків:

1) На роботу алгоритмів зовсім не впливає зміст зображень. Різниця роботи алгоритмів з однаковим і різним змістом є нехтовно малою.
2) На час роботи алгоритмів впливає розмір самих зображень. Це обумовлено тим, що кожен з алгоритмів тим чи іншим чином перевіряє на відповідність кожен піксель двох зображень. І якщо кількість пікселів більша, то звичайно і час обробки зростає. Графік залежності часу від розмірності зображень для кожного алгоритму знаходиться на рисунку 12:

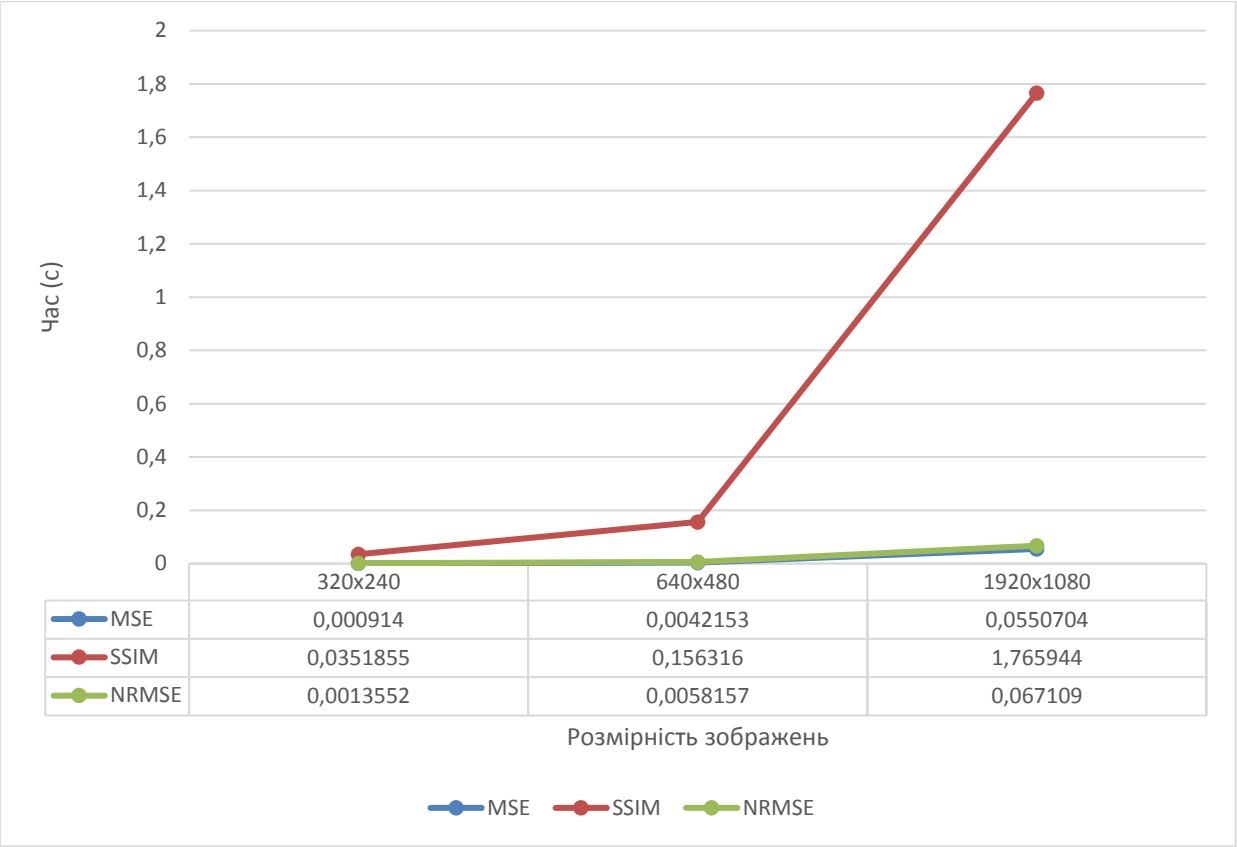

Рисунок 12 – Графік залежності часу роботи алгоритмів від розміру зображень

У наступному експерименті було протестовано правильність роботи алгоритмів у ситуаціях, подібних до першого експерименту. У першій частині використовувались ідентичні за змістом зображення. Результати експерименту продемонстровані на скріншотах 13 - 15:

```
/usr/bin/python3.6 /home/vetalik/PycharmProjects/test/testImages.py
Equvivalent pictures 1920x1080:
MSE:        result: 0.0
SSIM:        result: 1.0
NRMSE:        result: 0.0
```

Рисунок 13 – Результат порівняння ідентичних зображень розміром 1920x1080 пікселів

```
/usr/bin/python3.6 /home/vetalik/PycharmProjects/test/testImages.py
Equvivalent pictures 320x240:
MSE:        result: 0.0
SSIM:        result: 1.0
NRMSE:        result: 0.0
```

Рисунок 14 – Результат порівняння ідентичних зображень розміром 640x480 пікселів

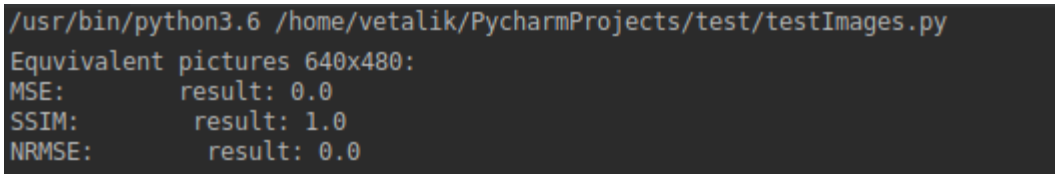

Рисунок 15 – Результат порівняння ідентичних зображень розміром 320x240 пікселів

У наступній частині експерименту проводилось тестування різних між собою зображень. Результати, що були отримані після експерименту, відображені на скріншотах 16 - 18:

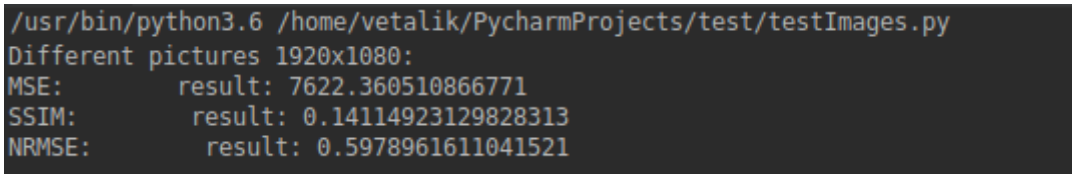

Рисунок 16 – Результат порівняння різних зображень розміром 1920x1080 пікселів

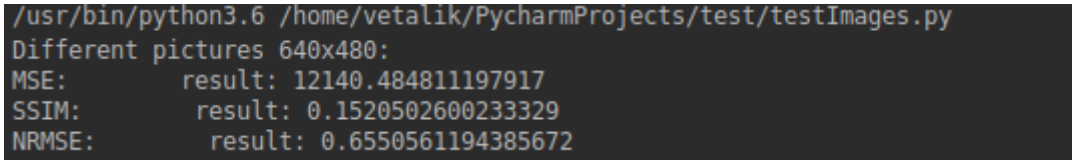

Рисунок 17 – Результат порівняння різних зображень розміром 640x480 пікселів

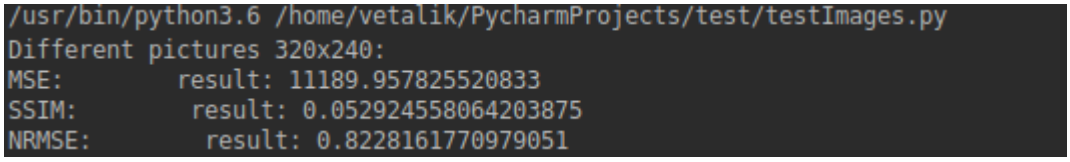

Рисунок 18 – Результат порівняння різних зображень розміром 320x240 пікселів

Отже, як бачимо, всі алгоритми, що тестувалися, правильно провели порівняння зображень. Також бачимо, що тільки алгоритм MSE має діапазон

результатів більше одиниці, на відміну від інших алгоритмів. Це дещо ускладнює роботу з ним, але не є підставою для відмови від цього алгоритму.

Наступний експеримент проводився на наборах зображень однакової розмірності, у яких будо дещо відредаговано колірну гамму. Зміст при цьому залишився незмінним. Значення, які було отримано в результаті проведення експерименту зображені на скріншоті 19:

```
/usr/bin/python3.6 /home/vetalik/PycharmProjects/test/testImages.py
Semi-equal pictures 640x480
MSE:        result: 2765.033837890625
SSIM:        result: 0.8168390771568612
NRMSE:        result: 0.31261565196056645
```

Рисунок 19 – Результат порівняння зображень зі зміненою колірною гаммою розміром 640x480 пікселів

Як бачимо, навіть з такою задачею обрані алгоритми справляються правильно.

Отже на основі цих експериментів можна зробити висновок:

- Найшвидшим алгоритмом є алгоритм MSE. Дещо довше з поставленими задачами працює алгоритм NRMSE, а найповільнішим виявився алгоритм SSIM. Це обумовлено тим, що останній алгоритм оцінює саме структурну різницю зображень, що й призводить до значного збільшення часу роботи алгоритму.
- Усі обрані алгоритми демонструють правильну роботу на обраних наборах зображень, отже всі вони можуть бути використані у подальшій розробці системи відеонагляду.
- Час роботи алгоритмів не залежить від змісту самих зображень. Це дуже добре впливатиме на стабільність роботи системи відеоспостереження, оскільки під час роботи такої системи потрібно аналізувати велику кількість як однакових, так і різних зображень.
- Час роботи залежить від розмірності вхідних зображень. Цей важливий момент не варто ігнорувати при майбутній розробці.

Розроблювані програмні засоби мають відповідати певним часовим вимогам, саме тому потрібно гарантувати коректність роботи такої системи для певної розмірності відео, яке отримуватиметься з відеокамери.

На основі результатів дослідження для подальшої розробки слід обрати алгоритм MSE, оскільки саме він має найкращу швидкість роботи, порівняно з іншими досліджуваними варіантами. Саме цей алгоритм і буде використовуватись при розробці програмних засобів для системи відеонагляду.

2.2. Опис програмних засобів розпізнавання руху об’єкта для систем відеоспостереження.

Розроблюваний програмний засіб повинен мати перш за все графічний інтерфейс, щоб користувач мав змогу керувати системою. Основною частиною інтерфейсу має бути вікно, в якому відображатиметься інформація, яка отримується з камери. Також повинні бути кнопки керування, такі як «Розпочати стеження» і «Припинити стеження». Після натискання кожної з них програма має відповідно реагувати на обрану користувачем дію.

Після запуску програми, автоматично повинні виконатися наступні дії:

1. Виконати пошук відеокамери та організувати між нею зв'язок
2. Створити шаблон вікна головної програми
3. На вікні розмістити полотно (canvas), в якому і будуть відображатися дані, що отримуються з відеокамери
4. На вікні також необхідно розмістити дві кнопки (button) «Розпочати стеження» і «Припинити стеження», за допомогою яких користувач і матиме змогу керувати програмою

Таким чином після запуску додатку користувач побачить вікно і поле, в якому вже відображаються дані, що надходять від камери. Оскільки відеопотік відображається безпосередньо при підключенні до камери, то користувач має змогу коригувати положення відеокамери. Також відео, що

відображається у вікні, не зупиняється після початку стеження. Тому особа, яка буде слідкувати за територією матиме змогу особисто слідкувати за змінами на контрольованій місцевості та швидко реагувати у разі надзвичайної ситуації.

Далі розглянемо дії програми при натисканні кнопки «Розпочати стеження». Після натискання даної кнопки запускається алгоритм, який виконує наступні дії:

1. Програма зберігає початковий стан місцевості. Тобто кадр, який записала відеокамера одразу ж після запуску алгоритму зберігається. Цей кадр і буде в подальшому братись за основу.
2. Для кожних наступних кадрів відбувається порівняння з основним кадром. Далі може бути два варіанти:
   2.1. Якщо змін в подальших кадрах не відбувається, то це означає, що всі об’єкти, що потрапили у кадр є статичними, а це свідчить, що на місцевості змін не відбувається і записувати ідентичну інформацію не має сенсу. Саме тому подібні знімки ігноруються програмою і програма переходить до аналізу наступних даних, що передаються нею, тобто повертаємось до пункту 2.
   2.2. Якщо зміни на території, що фіксується камерою, відбулися, то виконується ряд наступних дій:
      2.2.1. Необхідно замінити основний кадр поточним. Ця дія забезпечує в подальшому ігнорування ідентичних кадрів, при умові, що на території з’явиться новий об’єкт. Таким чином момент появи цього об’єкта обов’язково фіксується, а в подальшому, якщо він залишиться нерухомим, ідентична інформація буде ігноруватись.
      2.2.2. Потрібно організувати запис кадру до відеофайлу. Якщо такий файл ще не створено, то відбувається зчитування поточної дати та часу і створюється файл з ім’ям, що відповідає поточній даті та поточному часу. Таким чином, коли рух об’єктів було зафіксовано, дата та час початку спостережуваних змін відображатимуться в

імені відеофайлу. Після створення файлу (або ж даний файл вже було створено раніше) відбувається запис до файлу. Далі знову переходимо до зчитування наступних кадрів, а саме до пункту 2.

2.3. Якщо рух припиняється, то певну кількість кадрів після зміни розташування об’єктів також потрібно зафіксувати. Якщо ж рух припиняється остаточно, то програма закриває відеофайл і зберігає його. Наступні зміни на спостережуваній території будуть фіксуватися в наступний файл. Знову повертаємось до пункту 2.

Таким чином, як бачимо з опису дій, алгоритм є циклічним. Отже він буде працювати постійно після запуску стеження за місцевістю. Дата та час відображатиметься у назві збереженого відеофайлу, що значно прискорюватиме пошук потрібної інформації, порівняно зі звичайними системами захисту.

Для порівняння кадрів буде використовуватись алгоритм MSE, розглянутий вище. Швидкість його роботи має важливе значення, оскільки система працює з потоком інформації і має швидко її оброблювати, інакше створені програмні засоби будуть не дієздатні. При надто довгому аналізуванні зображень додаток буде пропускати кадри, що отримуватимуться з камери. Тому велика кількість інформації буде пропущена, що недопустимо для систем захисту.

Запис певної кількості кадрів потрібний для зменшення кількості файлів, що будуть створені. Можлива ситуація, коли на місцевості буде рухомий об’єкт, який на долю секунди стане нерухомим. Якби даної функції не було б у системі, програма б створювала велику кількість дуже коротких відео. Таким чином одна подія може бути поділена на десяток частин, які зберігатимуться окремо. Це ускладнювало б перегляд події та не вирішувало б повністю поставлену проблему.

Також для кращої інформативності у графічний інтерфейс слід додати індикатор, який буде інформувати користувача про процес запису кадрів у файл. Це буде корисно як при тестуванні, так і при роботі даних програмних

засобів, оскільки графічний інтерфейс чітко інформуватиме осіб, які будуть працювати з програмою, про організацію процесу запису у файл.

Також слід розглянути дії, що мають виконуватись при завершенні стеження. Зупинка стеження має повертати стан програми у початковий. Для цього слід:

1. Припинити порівняння наступних зображень, що зчитуються;
2. Якщо в цей момент відбувався запис матеріалу у файл, то:
   2.1. Зафіксувати кадри, що мають бути додані до відеофайлу після завершення руху. Але в даному випадку рух програмою вже не фіксуватиметься
   2.2. Правильно закрити відеофайл
   2.3. Зберегти відеофайл
   2.4. Очистити дані, що були потрібні для реалізації додатку

Таким чином після виконання вище описаних дій, програма повернеться до того стану, який безпосередньо був після запуску додатку.

Враховуючи алгоритмічну особливість даної системи можна зробити висновок, що дана система буде проявляти максимальну ефективність саме в малолюдних місцях, оскільки запис у відеофайл буде відбуватись лише тоді, коли система фіксуватиме рух у кадрі. Отже в найгіршому випадку, коли рух у відеопотоці буде постійним, розроблювана система буде поводити себе так, як звичайні системи відеоспостереження. Тобто у такій ситуації даний додаток не матиме явних переваг над іншими системами. Але при використанні розробки у місцях, де рух на території відбувається нечасто, дана розробка проявить свою ефективність, порівняно з іншими системами. Розроблювана система стеження в таких випадках буде значно економити вільне місце на носії інформації та зменшувати кількість часу, що витрачатиметься на перегляд відеоматеріалу. При цьому стеження буде проводитись безперервно. Тому дану систему відеоспостереження є сенс встановлювати у важливих мало відвідуваних місцях, таких як сховища

банків, архіви компаній, перепускні пункти на приватну територію та інші подібні місця.

3. ОПИС РОЗРОБЛЕНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ, ЇХ АЛГОРИТМІЧНІ ОСОБЛИВОСТІ

У даному розділі розглядаються особливості реалізації запропонованих програмних засобів для систем відеоспостереження. Програмні засоби реалізовані у вигляді класів, що взаємодіють між собою. Розробка проводилася мовою програмування Python, з використанням бібліотек OpenCV та Scikit.Image, які більш детально розглядатимуться в наступних підпунктах.

3.1. Підстави обрання мови програмування Python

Python проста у використанні, та водночас повноцінна мова програмування, що надає багато засобів для структурування і підтримки великих програм. Також вона краще за C обробляє помилки, і, будучи мовою дуже високого рівня, має вбудовані типи даних високого рівня, такі як гнучкі масиви і словники, ефективна реалізація яких на C потребує значних витрат часу.

Завдяки більш загальним типам даних, Python застосовують до більш широкого кола задач, ніж Awk і навіть Perl, у той ж час багато речей на мові Python робляться настільки ж просто.

Python дозволяє розбивати програми на модулі, що потім можуть бути використані в інших програмах. Python поставляється з великою бібліотекою стандартних модулів, які можна використовувати як основу для нових програм або як приклади при вивченні мови. Стандартні модулі надають засоби для роботи з файлами, системними викликами, мережними з'єднаннями і навіть інтерфейсами до різних графічних бібліотек.

Python - інтерпретована мова, що дозволяє заощадити значну кількість часу, що зазвичай витрачається на компіляцію. Інтерпретатор можна використовувати інтерактивно, що дозволяє експериментувати з можливостями мови, писати шаблони програм або тестувати функції при розробці “знизу-вверх”. Він також зручний як настільний калькулятор. Python дозволяє писати дуже компактні й зручні для читання програми.

Програми, написані мовою Python, звичайно значно коротші еквівалента на C або C++ з декількох причин:

- типи даних високого рівня дозволять Вам виразити складні операції однією інструкцією;
- групування інструкцій виконується за допомогою відступів замість фігурних дужок;
- немає необхідності в оголошенні змінних;

Відмінностей Python від інших мов доволі багато, перерахуємо основні з них:

- Керування пам'яттю – цілком автоматичне. Не потрібно хвилюватися щодо розподілу або звільнення пам'яті. Немає загрози “небезпечного посилання”.
- Типи зв'язані з об'єктами, а не зі змінними. Це означає, що змінній може бути призначене значення будь-якого типу, і що (наприклад) масив може містити об'єкти різних типів. Традиційні мови не надають такої можливості.
- Операції звичайно виконуються в більш високому рівні абстракції. Це частково результат того, як написана мова, і частково результат розширеної стандартної бібліотеки кодів, що поставляється разом з Python.

Ці та інші особливості Python роблять розгортання додатків надзвичайно швидким. Один недолік Python, у порівнянні з найбільш традиційними мовами, полягає в тому, що це - не цілком компільована мова; замість цього, вона частково транслює програму до внутрішньої форми байт-коду, і цей байт-код виконується інтерпретатором Python. Однак, у перспективі – сучасні комп'ютери мають так багато невикористовуваного обчислювального потенціалу, що для 90% додатків швидкодія зв'язана з вибором мови. Java теж компілюється в байт-код, але в даний час працює повільніше ніж Python у більшості випадків. Крім того, дуже просто

об'єднати Python з модулями, написаними на C або C++, які можна використовувати, щоб збільшити швидкість роботи програм в критичних ділянках.

До переваг мови програмування Python також можна віднести велику кількість бібліотек, які можна з легкістю використати у своїх розробках. Ці бібліотеки значно розширюють можливості Python та сприяють прискоренню розробки програмних засобів, використовуючи мову програмування Python.

Також додатки, розроблені на мові Python не залежать від операційної системи. Таким чином розроблений продукт буде коректно працювати на будь-якій операційній системі, за умови, що на ній буде встановлений інтерпретатор з тими бібліотеками, які необхідні додатку для роботи. Дана можливість позитивно вплине на будь-яку розробку, оскільки без зміни вихідного коду додаток може бути запущений на багатьох пристроях із різними операційними системами, що призведе до збільшення кількості потенційних користувачів.

Враховуючи всі переваги, було прийняте рішення виконувати розробку програмних засобів розпізнавання руху об’єкта для систем відеоспостереження саме на мові Python.

3.2. Бібліотека OpenCV

OpenCV (Open Source Computer Vision Library) - це бібліотека програмного забезпечення з відкритим кодом для комп'ютера та машинного навчання. OpenCV була побудована, щоб забезпечити загальну інфраструктуру для комп'ютерних програм зору та прискорити використання сприйняття машин у комерційних продуктах. OpenCV, що є продуктом ліцензії BSD, дозволяє розробникам легко використовувати та змінювати код.[16]

Бібліотека має понад 2500 оптимізованих алгоритмів, що включає в себе повний комплект як класичного, так і сучасного комп'ютерного бачення та алгоритмів машинного навчання. Ці алгоритми можуть бути використані

для виявлення та розпізнавання обличчя, визначення об'єктів, класифікації людських дій у відео, витягування 3D-моделей об'єктів, зшивання зображень разом для отримання високої роздільної здатності зображення цілісної сцени, знайти схожі зображення з бази даних зображень, видалити червоні очі з зображень, створених за допомогою спалаху, стежити за рухами очей, визнати декорації та встановити маркери для накладання його на додану реальність тощо. OpenCV налічує більше 47 тисяч користувачів спільноти та понад 14 мільйонів завантажень. Бібліотека широко використовується в компаніях, дослідницьких групах та урядових структурах.

Поряд із добре відомими компаніями, такими як Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota, які використовують цю бібліотеку, є багато нових компаній, таких як Applied Minds, VideoSurf та Zeitera, які широко використовують OpenCV.

Вона має інтерфейси C++, Python, Java та MATLAB і підтримує різні операційні системи, такі як Windows, Linux, Android та Mac OS. OpenCV спрямовується в основному на додатки в режимі реального часу та використовує інструкції MMX та SSE, коли це можливо. В даний час активно розвиваються повнофункціональні інтерфейси CUDA та OpenCL. Існує понад 500 алгоритмів і приблизно в 10 разів більше функцій, які складають або підтримують ці алгоритми. OpenCV написано спочатку на C++ і має шаблонний інтерфейс, який працює без проблем з контейнерами STL.[16]

У рамках даного проекту з цієї бібліотеки будуть використовуватися підпрограми, які дозволяють отримати доступ до відеокамери та зчитувати дані з неї. Також з бібліотеки будуть використані алгоритми по перетворенню зображення з однієї колірної моделі в іншу. Річ у тім, що бібліотека OpenCV працює з використанням колірної моделі BGR, яку перед записом у файл необхідно змінити, інакше на виході буде отримано зображення з відповідними кольоровими дефектами.

Із даної бібліотеки у розробці системи стеження буде використовуватись декілька підпрограм, що допомагають у реалізації програмних засобів.

- Клас *cv2.VideoCapture*.

Клас призначений для опису методів, які допомагають у отриманні доступу до відеокамери і зчитування інформації з неї. При створенні екземпляру класу командою:

- *cv2.VideoCapture(video_source);*

буде створений екземпляр класу, що має доступ до відеокамери. *Video_source* це номер відеокамери у системі, за замовчуванням має значення нуль. За допомогою методів даного екземпляру можна отримувати дані з камери. Для цього використовується метод:

- *cv2.VideoCapture(video_source).read();*

який і повертає саме зображення, отримане з камери у місце виклику. Для перевірки коректності отримання доступу до камери існує метод-предикат:

- *cv2.VideoCapture(video_source).isOpened();*

який повертає значення «істина», якщо вдалось отримати доступ до камери.

Також з цього класу використовується метод:

- *cv2.VideoCapture(video_source).release();*

який правильно завершує роботу з камерою.

- Клас *cv2.VideoWriter*.

Клас призначений для організації процесу запису кадрів у файл. При створенні екземпляру класу командою:

- *cv2.VideoWriter(name_, fourcc, 20, (640, 480))*

у яку передаються чотири параметри:

1. Назва відеофайлу.
2. Кодек, що використовуватиметься для кодування відеоматеріалів.

3. Число, що задає кількість кадрів у секунду. Саме ця кількість кадрів і записуватиметься у відеофайл кожної секунди.
4. Розмірність відеофайлу.

Після виконання даної команди буде створений екземпляр класу *VideoWriter*, який і надає зручний інтерфейс за організації запису кадрів у файл. Після запису, відеофайл можна переглянути за допомогою будь-якого відеопрогравача.

- Підпрограма *cv2.VideoWriter_fourcc.*

Ця підпрограма задає тип кодеку, який використовуватиметься для запису відеофайлу. Ко́дек (англ. codec — скорочено від coder/decoder (кодування/декодування) або compressor/decompressor) — програма, здатна виконувати перетворення потоку даних або сигналу. Кодеки можуть як кодувати потік/сигнал (часто для передачі, зберігання або шифрування), так і розкодовувати — для перегляду або зміни у форматі, що більше підходить для цих операцій. Кодеки часто використовуються при цифровій обробці відео й аудіо.

Даний метод приймає в себе чотири символьні параметри, за допомогою яких і визначається тип кодеку, що використовуватиметься.

- Підпрограма *cv2.VideoWriter().write().*

Цей метод виконує запис у файл, що був створений разом зі створенням екземпляру цього класу. Параметром передається сам кадр, тобто зображення, яке необхідно записати у відеофайл.

- Підпрограма *cv2.imread.*

За допомогою неї відбувається зчитування зображення з файлу. Параметром у дану підпрограму передається шлях до самого файлу в системі.

- Підпрограма *cv2.imwrite.*

За допомогою неї відбувається запис зображення у файл. Параметром у дану підпрограму передається ім’я файлу зображення і саме зображення, яке необхідно зберегти у файл.

Усі вище описані можливості цього модулю будуть використовуватися при розробці програмних засобів для системи стеження.

3.3. Бібліотека Scikit-Image

Scikit-Image – це бібліотека для обробки зображень із відкритим кодом для мови програмування Python. Вона включає алгоритми сегментації, геометричних перетворень, маніпулювання кольоровим простором, аналіз, фільтрацію, морфологію, виявлення функцій тощо. Він призначений для взаємодії з численними та науковими бібліотеками Python, такими як NumPy та SciPy.[17][18]

У даній бібліотеці реалізовано також алгоритми порівняння зображень, які використовуються у розробленій системі. Саме з цієї бібліотеки були взяті реалізації алгоритмів та використані у розробці.

Імпортувати бібліотеку потрібно командою:

- *import skimage.measure*

Після імпортування в головній програмі доступні три підпрограми, які виконують порівняння зображень за допомогою трьох різних алгоритмів. Підпрограму, яка порівнює зображення, використовуючи алгоритм MSE, можна виконати за командою:

- *skimage.measure.compare_mse(first_im, sec_im)*

Підпрограма має отримувати два зображення (*first_im, sec_im*) які і будуть порівнюватись між собою. Вона також повертає результат порівняння у місце виклику. Дана підпрограма може повертати значення більше нуля, або нуль. Якщо результатом є нуль, це означає, що два зображення, що були передані у підпрограму, рівні між собою. Якщо ж результат більше нуля,

зображення різні. Чим більший результат – тим більше різниця між зображеннями.

Наступна підпрограма порівнює зображення за допомогою алгоритму NRMSE. Виконати порівняння можна командою:

- *skimage.measure.compare_nrmse(first_im, sec_im)*

де *first_im, sec_im* – зображення, які необхідно порівняти між собою. Результатом даної підпрограми є результат оцінки зображень за шкалою від нуля до одиниці, де нуль означає, що зображення ідентичні. Оцінка, як і в попередньому випадку повертається у місце виклику команди.

Третя підпрограма, яка використовуватиметься під час розробки, оцінює зображення алгоритмом SSIM. Викликається командою:

- *skimage.measure.compare_ssim(first_im, sec_im, multichannel=True)*

Параметрами у дану підпрограму передаються два зображення (*first_im, sec_im*), які і будуть оцінюватись. Також потрібно встановити опцію *multichannel*, що розглядає останній вимір масиву як канали. Розрахунки подібності проводяться незалежно для кожного каналу, а потім усереднюються.

Результатом оцінки є число, що знаходиться у діапазоні від нуля, до одиниці, де одиниця означає ідентичність зображень.

3.4. Бібліотека TKinter

Tkinter - це кросплатформна бібліотека для розробки графічного інтерфейсу в Python (починаючи з Python 3.0 перейменована в tkinter). Tkinter розшифровується як Tk інтерфейс, і є інтерфейсом до Tcl/Tk. Tkinter входить в стандартний дистрибутив Python. [6]

У різноманітті програм, які пишуть програмісти, виділяють додатки з графічним призначеним для користувача інтерфейсом (GUI). При створенні таких програм стають важливими не тільки алгоритми обробки даних, а й розробка для користувача програми зручного інтерфейсу, взаємодіючи з яким, він буде визначати поведінку програми.

Сучасний користувач в основному взаємодіє з програмою за допомогою різних кнопок, меню, значків, вводячи інформацію у спеціальні поля, вибираючи певні значення в списках і т. д.

Для мови програмування Python такі віджети включені в спеціальну бібліотеку – tkinter. Якщо її імпортувати в програму, то можна користуватися її компонентами, створюючи графічний інтерфейс.[6]

Послідовність кроків при створенні графічного додатку має свої особливості. Програма повинна виконувати своє основне призначення, бути зручною для користувача, реагувати на його дії. Розглянемо які етапи приблизно потрібно пройти при програмуванні, щоб отримати програму з GUI:

1. Імпорт бібліотеки;
2. Створення головного вікна;
3. Створення віджетів;
4. Встановлення їх властивостей;
5. Визначення подій;
6. Визначення обробників подій;
7. Розташування віджетів на головному вікні;
8. Відображення головного вікна.

Для імпорту бібліотеки, як і будь-якого іншого модуля, tkinter в Python можна імпортувати двома способами:

- Командою *import tkinter*
- Командою *from tkinter import **

Для зручності варто користуватись другим варіантом, оскільки в такому випадку не потрібно кожного разу писати ім’я модулю перед викликом підпрограми. Слід звернути увагу, що у версії Python 3 ім'я модуля пишеться з малої літери (tkinter), хоча в більш ранніх версіях використовувалася велика літера (Tkinter).

Далі потрібно створити головне вікно програми. У даному вікні розташовуються всі інші елементи. Об'єкт вікна верхнього рівня створюється при зверненні до класу Tk модуля tkinter. Подібний рядок коду створює головне вікно:

- *root = Tk()*

Потім необхідно створити самі елементи. У модулі вже присутні деякі елементи, які з легкістю можна використати на створеному вікні. Серед таких елементів є:

- «Кнопка» (Button)
- «Мітка» (Label)
- «Полотно» (Canvas)
- «Рамка» (Frame)
- «Текст» (Text)
- «Смуга прокрутки» (Scrollbar)
- Та інші.

Для реалізації програмного продукту було використано наступні елементи: «Кнопка» - для задання дій, що має виконати програма, «Мітка» - для виводу зображення, «Полотно» - для створення області, призначеної для відображення інших елементів. Після створення елементів варто також призначити кожному елементу певні налаштування:

- Задати розміщення у головному вікні;
- Внести певні дані для коректної роботи програми;
- Задати зовнішній вигляд елементу.

Наступним кроком буде призначення обробника подій. Обробником подій в цьому випадку виступатиме підпрограма, яка почне виконуватися при настанні певної події, наприклад натискання на кнопку.

Після цього потрібно розмістити створені елементи на головному вікні, щоб вони відображалися при запуску програми. Для цього можна виконати команду:

- *<ім'я елементу>.pack()*

Ця команда автоматично розмістить елемент у головній програмі. Після цього він буде доступний для взаємодії з користувачем.\

Останнім кроком буде вивід цього вікна для користувача. Головне вікно не з'явиться, доки не буде виконаний такий рядок:

- *root.mainloop()*

Ця команда має виконуватись після усіх вище описаних дій.

3.5. Опис графічного інтерфейсу

Розроблений додаток містить у собі графічний інтерфейс, для зручного користування. Графічний інтерфейс виконаний у мінімалістичному стилі та містить лише дві кнопки для контролю над системою. Загальний дизайн системи зображений на рисунку 20:

Рисунок 20 – Загальний дизайн системи

Найбільшу площу займає панель виводу відеопотоку, що отримується з камери. Кожна система стеження має містити засоби для безпосереднього виводу зображення на екран. Саме за допомогою цієї панелі користувач має змогу слідкувати за подіями, що відбуваються в поточний момент часу. Також дана можливість допомагає при встановленні камер, оскільки при виконанні цього процесу користувач має змогу одразу ж спостерігати, яку область буде стежити встановлена відеокамера.

Також у верхньому лівому кутку є спеціальне поле-індикатор. Дане поле може фарбуватися в червоний колір. Це буде свідчити про те, що дана система стеження відслідкувала рух об’єктів у кадрі і автоматично організувала запис відеоматеріалу. Якщо ж поле не зафарбоване, це означає, що системою не розпізнається рух і запис кадрів не відбувається.

Далі розглядаються кнопки керування даною системою. Першою є кнопка «Розпочати стеження (start tracking)». Після натискання даної кнопки запускається алгоритм (див. нижче), який і переводить дану систему у стан пошуку. Після цього всі рухи об’єктів, що будуть розпізнаватися програмними засобами, записуватимуться у відеофайли. Кількість створених відеофайлів рівна кількості подій, що буде відстежуватися.

Стеження відбуватиметься доки користувач не натисне кнопку «Закінчити стеження (stop tracking)». Після натискання даної кнопки виконуються всі дії (див. нижче), що необхідні для завершення стеження і переходу системи у початковий стан, який був відразу ж після запуску системи.

3.6. Структура розробленої системи

Структура розробленої системи стеження складається з двох класів, описаних мовою програмування Python, та шести імпортованих модулів. діаграма класів розробленої програми, що демонструє зв’язки між модулями, зображена на рисунку 21:

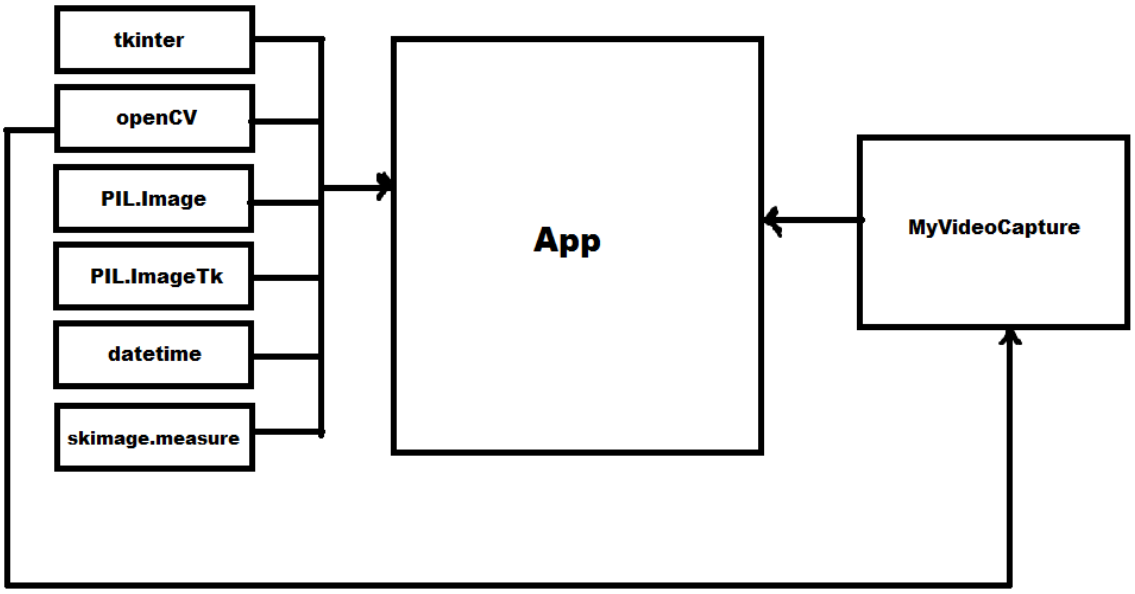

Рисунок 21 – Схема зв’язків між класами у розробленій системі

Програма імпортує в себе такі модулі:

- tkinter – модуль, що відповідає за графічний інтерфейс (п. 3.2)
- cv2 – OpenCV модуль, який відповідає за правильну роботу додатку з камерою (п. 3.3)
- PIL.Image – модуль, що дозволяє додатку виконувати різні дії над зображеннями
- PIL.ImageTk – модуль, що надає зручне API для інтеграції зображень з бібліотекою tkinter
- datetime – стандартний модуль, який виконує роботу, пов’язану з датою та часом
- skimage.measure – Scikit-Image модуль, який надає реалізовані алгоритми для обробки зображень

3.6.1. Опис реалізованого класу MyVideoCapture

Клас MyVideoCapture – клас, який полегшує роботу з отриманням інформації з відеокамери. Даний клас виконує дії для підключення до відеокамери, дає змогу зчитувати дані з камери та виконує дії, необхідні

після завершення роботи з нею. Загальна структура класу зображена на рисунку 22:

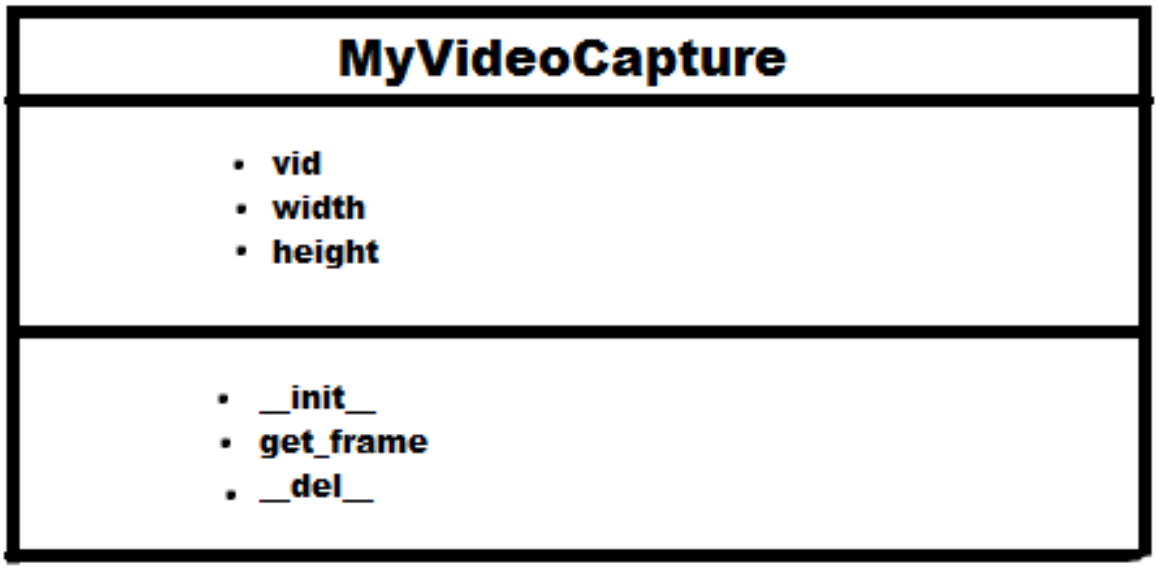

Рисунок 22 – Структура класу MyVideoCapture

Із рисунку видно, що даний клас складається тільки з трьох методів. Описаний клас буде використовуватись у головному класі *App,* що описується у пункті 3.2. Далі приводиться детальний огляд цих методів. Вихідний код класу представлено у Додатку 1.

3.6.1.1 Метод *__init__*

Даний спеціальний метод викликається одразу ж після створення екземпляру даного класу і виконує дії, які необхідні створення екземпляру. Сигнатура даного методу виглядає таким чином:

- *def __init__(self, video_source=0)*

У цей метод параметром передається сам створений екземпляр, за допомогою формального параметру *self.* Також може передаватися індекс камери, яку необхідно підключити. Відповідний формальний параметр для даного індексу – *video_source* Це створено на випадок, коли до пристрою підключено більше однієї камери. За замовчуванням цей параметр має значення нуль, що і інформує систему про необхідність підключення до найпершої камери у списку відеокамер.

У тілі самого методу виконуються дії по ініціалізації екземпляру даного класу. Перш за все метод організовує підключення до відеокамери, використовуючи модуль cv2. Одразу ж після виконання цієї дії відбувається перевірка правильності підключення, за допомогою методу *isOpened()*. Якщо ж підключення було невдалим, метод ініціює виключну ситуацію із відповідним повідомленням.

Після вдалого підключення метод також створює два атрибути-дані: *width* та *height.* Ці атрибути визначають відповідно ширину та висоту отримуваного кадру. Дані визначаються автоматично з використанням методу *get* у екземплярі класу *VideoCapture().*

Таким чином метод *__init__* створить екземпляр власного класу *MyVideoCapture*, за допомогою якого і буде відбуватись зчитування кадрів з відеокамери.

3.6.1.2. Метод *get_frame*

Цей метод зчитує поточне зображення з камери та повертає його у місце виклику. Параметром передається посилання на екземпляр, з якого і був здійснений виклик цього методу. Зчитування відбувається за допомогою команди *read()*. Ця підпрограма повертає два значення: результат зчитування і безпосередньо сам кадр. Якщо ж зчитування пройшло успішно, то метод повертає саме зображення як результат. У всіх інших випадках даний метод повертає спеціальне значення *None.*

3.6.1.3. Метод *__del__*

Спеціальний метод *__del__()* виконує дії, пов’язані із звільненням ресурсів під час видалення екземпляру класу *MyVideoCapture.* Метод викликається автоматично. У методі виконуються дії, які потрібні для правильного завершення роботи з відеокамерою. Вихідний код методу представлений у Додатку 1.

3.6.2. Опис реалізованого класу App

Клас *App* є головним класом програми. Він об’єднує всі складові додатку і безпосередньо виконує керування головної програми. Загальна структура класу представлена на рисунку 23:

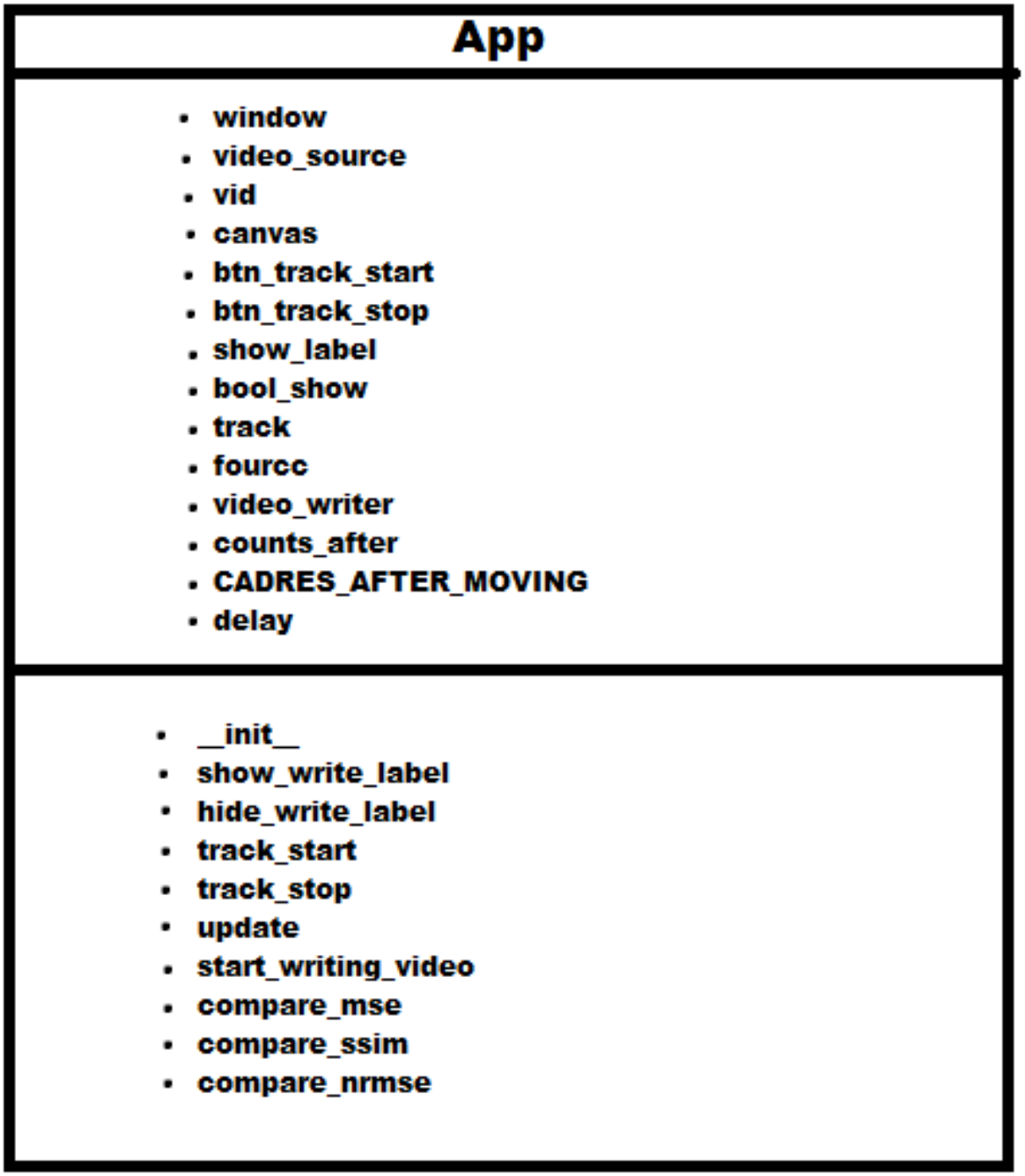

Рисунок 23 – Структура класу App

У даному класі реалізовано десять методів, за допомогою яких виконується створення головного вікна програми. Також у цьому класі містяться методи, які виконують дії алгоритму, описані в пункті 2.2. Далі

приводиться детальний огляд цих методів. Вихідний код класу представлено у Додатку 1.

3.6.2.1 Метод *__init__*.

Спеціальний метод *__init__()* є одним з головних підпрограм даного класу. Після запуску програми має створюватись екземпляр класу *App*, за що і відповідає даний метод. У ньому розміщені всі дії для надання програмі початкового стану. Такі дії виконуються у методі:

1. Ініціалізація апаратних складових системи
2. Відкриття відео ресурсу, тобто відеокамери
3. Створення елементів інтерфейсу програми
4. Встановлення керуючих атрибутів екземпляру класу у початковий стан
5. Виклик методу *update()*
6. Запуск головного циклу програми

При ініціалізації апаратних складових системи відбувається зчитування параметрів, які передаються у даний метод, та подальше збереження цих даних. Параметрами передаються саме вікно, на якому і будуть відображатись елементи інтерфейсу програми, назва програми та індекс відеокамери в системі, до якої має виконатися підключення.

Далі за допомогою створеного класу *MyVideoCapture* відбувається підключення і відкривання відео ресурсу. Сам відео ресурс вказується параметром у конструктор класу *MyVideoCapture*.

Після цього у методі виконується створення елементів інтерфейсу та розміщення їх у тому вікні, що передався параметром у метод *__init__()*. Перш за все створюється полотно, яке адаптується під розмір отримуваного зображення з камери. На цьому полотні і буде відображатися в подальшому інформація, яка отримується з відео ресурсу. Далі відбувається створення кнопок, які будуть керувати даною системою. Першою створюється кнопка «Розпочати стеження». При створенні також вказується той метод, що

запускається при натисканні цієї кнопки. Він має назву *track_start().* Такі ж самі дії виконуються і для кнопки «Зупинити стеження». Для цієї кнопки вказується інший метод, що буде запущений. Він має назву *track_stop().* Після цього також створюється додаткове полотно розмірами 27x27 пікселів. Воно буде відображати той індикатор, який і буде вказувати користувачу про те, що програма відслідкувала рух об'єктів, та розпочала запис відеоматеріалу. Також для кожного елементу в кінці потрібно викликати метод *pack().* Він автоматично розмістить всі вище створені елементи у головному вікні.

Далі встановлюються головні налаштування для відеофайлів, які будуть створені. Перш за все визначається налаштування кодеку за допомогою підпрограми *VideoWriter_fourcc(),* описаної в п.3.2. У системі буде використовуватись кодек Xvid. Xvid – відеокодек стандарту MPEG-4, є основним конкурентом кодека DivX Pro. На противагу кодеку DivX — комерційному програмному забезпеченню, розробленому компанією DivX, Inc., Xvid — це вільна програма, заснована на колись відкритому вихідному коді іншого кодека. Це також означає, що на відміну від кодека DivX, випущеного тільки для платформ Microsoft Windows і Mac OS X, Xvid можна використовувати на всіх платформах і операційних системах, для яких можна скомпілювати вихідний код кодека, наприклад, для FreeBSD.

Після встановлення налаштувань кодека також варто встановити початкові значення для змінних, які потрібні для технічної реалізації. Змінна *video_writer* буде зберігати в собі екземпляр класу *cv2.VideoWriter().* Це необхідно для організації запису кадрів у файл. При створенні нового екземпляру створюється новий файл. Таким чином можна створювати декілька файлів з різним ім'ям, що нам і потрібно для реалізації алгоритму. Також описується константа *CADRES_AFTER_MOVING*, яка за замовчуванням дорівнює 40. Це кількість кадрів, які будуть додані до файлу відео після закінчення руху в потоці відео.

Далі викликається метод *update()*, який після першого виклику буде викликатись кожні 10 мілісекунд. Період, з яким буде відбуватись виклик даного методу вказаний в змінній *delay*.

Наприкінці виконується команда *tkinter.Tk().mainloop()*. Саме після виконання цієї команди і з'являється вікно головної програми, з якою і взаємодіє користувач.

Таким чином після виконання методу *__init__()* програма створює головне вікно і задає початковий стан системи.

3.6.2.2. Методи *show_write_label* і *hide_write_label*

Ці методи стосуються однієї логічно-зв'язаної дії. Вони організовують правильну роботу з індикатором запису.

Метод *show_write_label()* відповідає за появу червоного кружечка на полотні, що було створене у методі *__init__()*. У тілі методу виконується команда *create_oval()*, яка і рисує кружечок із заданими параметрами.

Метод *hide_write_label()* відповідає за зникнення кружечка, який був нарисований раніше. У тілі цього методу відбувається видалення цього кружечка. Таким чином відповідне полотно залишиться пустим, як і було до розпізнання руху. Роботу цих методів можна спостерігати на скріншотах 24 та 25:

Рисунок 24 – Демонстрація роботи методу *hide_write_label()*

Рисунок 25 – Демонстрація роботи методу *show_write_label()*

3.6.2.3. Методи *track_start* і *track_stop*

Ці методи виконують функцію керування системою. Вони ж і запускаються при натисканні кнопки «Розпочати стеження» та «Завершити стеження» відповідно.

Робота методу *track_start()* переводить систему у стан «роботи». Після виконання дій, що описані в цьому методі програма буде виконувати алгоритм стеження, що був описаний у розділі 2.2.

Метод містить у собі наступні дії. Спочатку відбувається зчитування кадру з камери і відбувається його збереження під ім’ям «default.jpg». Це і буде початковий кадр, з яким і порівнюватимуться наступні зображення, отримані з відеокамери.

Також у цьому методі присвоюється значення 1 керуючій змінній track. Це означає, що система має перейти у стан «роботи».

Метод *track_stop()* виконує зворотню дію: він встановлює 0 у змінній *track*. Таким чином система припиняє розпізнавання руху і повертається до початкового стану.

3.6.2.4. Статичні методи *compare_mse, compare_ssim, compare_nrmse*.

Статичним методом є метод, який виконує дії незалежно від стану екземпляру, але ці дії логічно-пов’язані з роботою екземпляру. На мові Python статичний метод визначається за допомогою декоратору *@staticmethod.*

Методи *compare_mse(), compare_ssim()* та *compare_nrmse()* виконують звернення до модулю Scikit-Image та викликають відповідні підпрограми для порівняння двох зображень. Кожен з цих методів приймає 2 параметри: зображення, створене за замовчуванням, та зображення, що зчитується в поточний момент часу. Методи повертають оцінку порівняння зображень, за відповідними алгоритмами MSE, SSIM та NRMSE.

3.6.2.5. Метод *start_writing_video*

У цьому методі реалізовані дії, пов’язані з фіксацією відеоматеріалу, що отримується з камери. Параметрами передається кількість кадрів, які необхідно записати, і сам кадр, зчитаний з відеокамери. Він реалізує наступну послідовність дій:

1) Якщо рух щойно був зафіксований, то виконується створення нового відео файлу та виконується запис переданого кадру.
2) Якщо рух був зафіксований раніше, але продовжується до поточного часу, то у такому разі відбувається просто запис кадру у файл, створений раніше.
3) Якщо рух припинився, то необхідно зберегти відеофайл, та відновити початкові дані керуючих змінних.

Також після виконання дій, описаних у першому пункті необхідно здійснити виклик методу *show_write_label()* для інформування користувача про початок запису матеріалу. Відповідно після завершення запису кадрів у файл, тобто після виконання дій у третьому пункті, викликати метод *hide_write_label().* Таким чином користувач буде проінформований, у який саме момент часу відбувається збереження відео.

3.6.2.6. Метод *update*

Даний метод автоматично викликається із заданим періодом, тому саме у тілі цього методу реалізована логіка, яка в залежності від настання певних подій керує поведінку всієї системи в цілому.

Роботу цього методу можна описати такою послідовністю дій:

1) Зчитування кадру в поточний момент часу.
2) Відображення кадру на екрані головної програми. Якщо зчитування кадру пройшло успішно, то поточний кадр за допомогою підпрограми *create_image()* виводиться у створене в інтерфейсі полотно.

3) Якщо система знаходиться у стані «стеження», виконати порівняння поточного кадру з тим, що був збережений за замовчуванням при запуску системи.
4) Якщо ж аналіз кадру виявив зміни, то встановити мінімальну кількість кадрів, що має записати програма. Мінімальна кількість збережена у константі *CADRES_AFTER_MOVING*. Також необхідно перезаписати кадр, що був збережений за замовчуванням.
5) Здійснити виклик власного методу *start_writing_video()* із передачею відповідних параметрів. Далі цей метод вирішуватиме, чи потрібно зберігати кадр чи ні.
6) Створити заплановану дію. Це виконується за допомогою підпрограми *after(),* в яку першим параметром передається час, через який має виконатись підпрограма, що передана у другому параметрі.

3.7. Практична цінність розробленого продукту

Програмні засоби розпізнавання рухомого об’єкта здатні значно оптимізувати роботу систем стеження. За допомогою їх можна знизити об’єм використовуваної пам’яті, шляхом ігнорування непотрібної інформації.

Бувають ситуації, коли на території, що знаходиться під наглядом відеокамери, не відбувається ніяких змін. Таким чином камера фіксує ідентичні кадри, які потім передаються на головний керуючий пристрій. Існуючі системи відеоспостереження постійно виконують запис відеоматеріалів. Це відбувається і у вище описаній ситуації. Як результат ми отримуємо, що велика кількість таких кадрів займає певний розмір пам’яті на зберігаючому пристрої, але такі відеоматеріали не містять у собі ніякої інформативності. Саме тому дана розробка є досить корисною для систем відеоспостереження, оскільки така система аналізує вхідний потік відеокадрів і приймає рішення щодо зберігання інформації, в залежності від даних, що надходять з камери.

Така система стеження матиме значну перевагу, порівняно з існуючими. Перш за все, це об’єм пам’яті, що потрібний системі для збереження відеоматеріалу за деякий проміжок часу, враховуючи те, що стеження в такій системі не припиняється. Також це полегшує перегляд та пошук потрібної інформації при необхідності відновлення події. Оскільки розроблена система зберігає лише ту інформацію, яка безпосередньо може бути корисною, то перегляд таким чином займатиме менше часу.

4. ТЕСТУВАННЯ РОЗРОБЛЕНИХ ПРОГРАМНИХ ЗАСОБІВ

Наступним етапом розробки системи стеження є тестування. Цей етап є дуже важливий, адже він допомагає знайти помилки в розробленому програмному забезпечені та впевнитись в його коректній роботі. Тестування буде проводитись комплексно, тобто перевірці буде підлягати саме система в цілому, а не окремі її частини. Таким чином буде протестований зв'язок окремих компонентів системи, а також правильність їх роботи у системі відеоспостереження.

Тестування системи буде проводитись на комп’ютері з такими характеристиками:

- процесор: Intel(R) Core(TM) i3-3110M;
- графічний процесор: Intel(R) HD Graphics 4000
- частота процесора: 2.4 ГГц;
- оперативна пам’ять: 8 ГБ;
- операційна система: Ubuntu 64-bit.

Камера буде використовуватись вбудована в ноутбук HP Probook 4540s. Виробником даної камери є компанія HP. Максимальна підтримувана роздільна здатність – 0,3 мегапікселя при 30 кадрах у секунду.

Система підлягатиме тестуванню в різних умовах навколишнього середовища. Порівняння буде проводитись з такою ж системою, але її робота буде подібною до роботи існуючих систем. Таким чином буде виключений апаратний фактор і саме програмні засоби будуть ключовими при отриманні результатів тестування.

4.1. Огляд розробленої програми

Розроблена система дуже проста в керуванні. Вона є звичайним скриптом на мові Python. На рисунку 26 видно, що це один файл з розширенням ‘.py’.

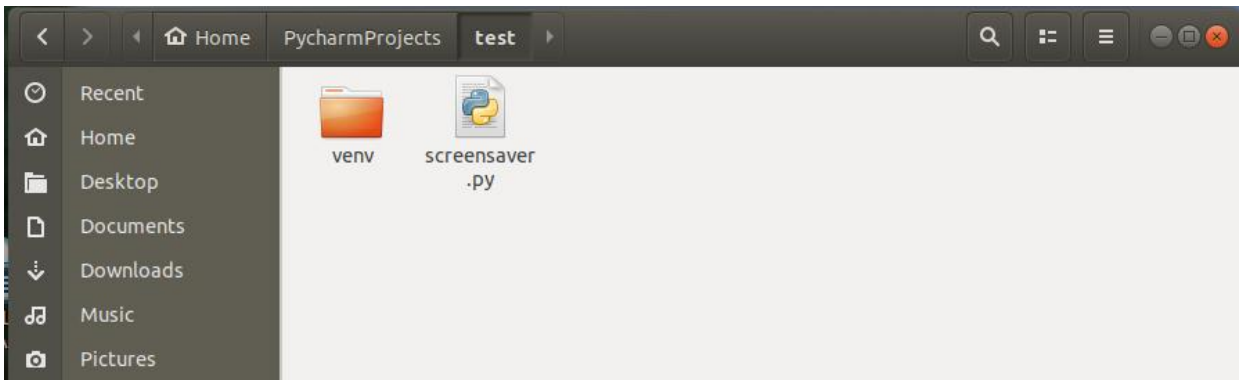

Рисунок 26 – Виконуваний файл системи

Оскільки програми на мові Python є кросплатформними, тобто такими, що можуть виконуватися незалежно від ОС, то і в даному випадку операційна система на якій буде запускатись програма є не важливою.

Для запуску скрипту необхідно мати лише встановлений інтерпретатор та бібліотеки, що описані у пункті 3.6. Виходячи з того що дана система розроблювалась з використанням мови Python третьої версії, то саме такий інтерпретатор і має бути встановлений.

Додаток можна запустити, виконавши таку команду у командному рядку:

- python3 tracksystem.py

У командному рядку це виглядає так (рис. 27):

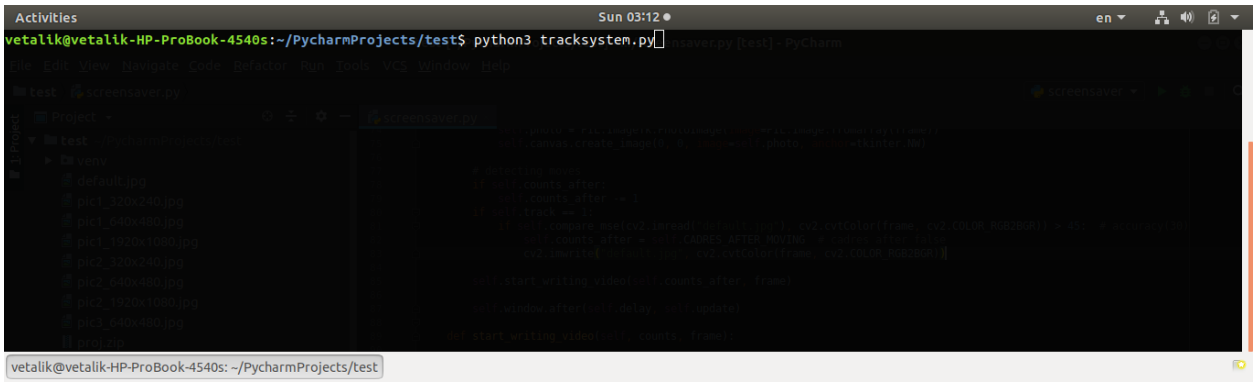

Рисунок 27 – Запуск системи у терміналі

Далі після введення такої команди користувач побачить головне вікно програми. Результат виконання команди вище зображений на рисунку 28:

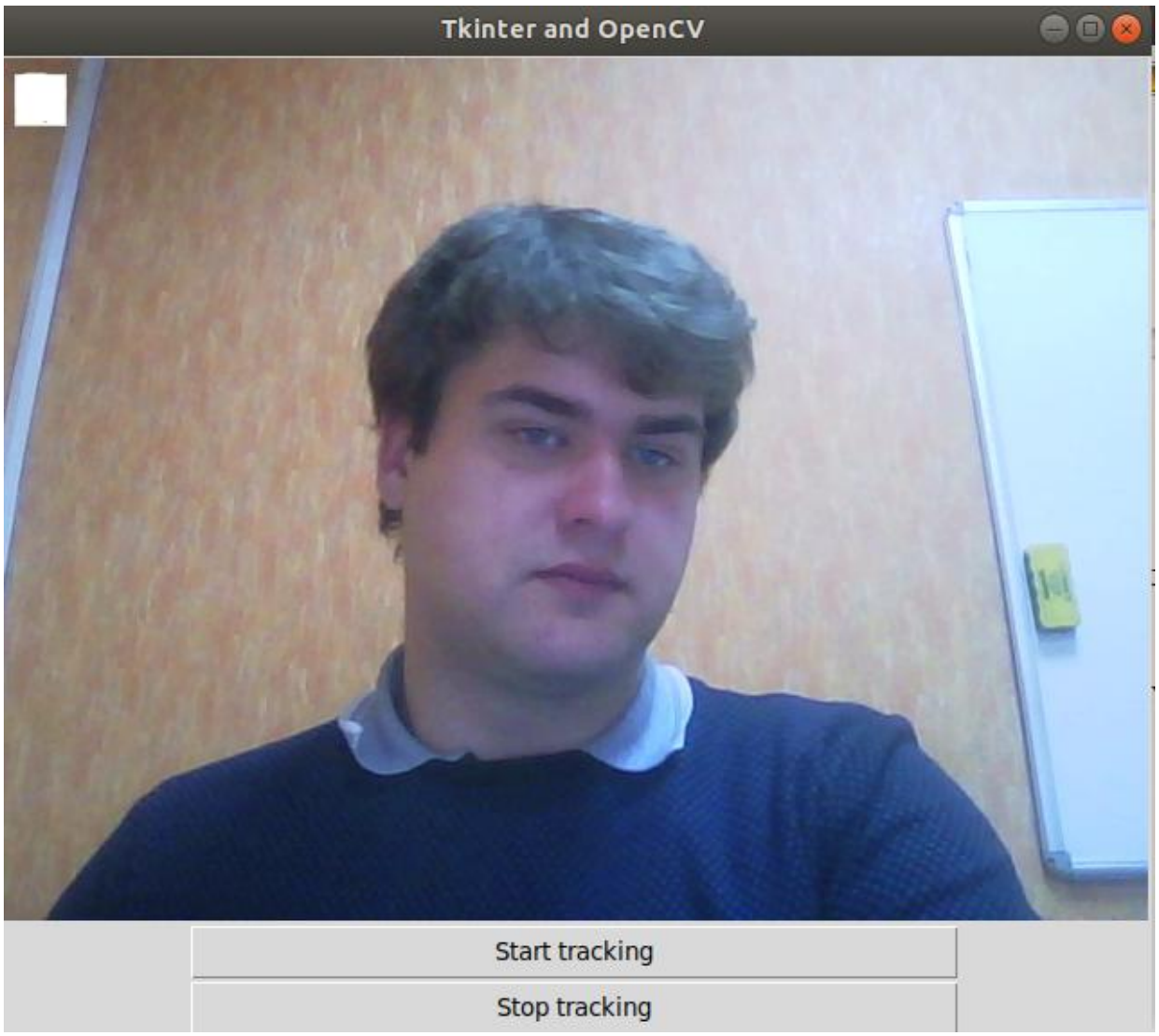

Рисунок 28 – Початковий стан системи відеонагляду

Це і є початковий стан, про який описувалось у п. 3.6. Після запуску можна спостерігати, що камера увімкнулась автоматично і зображення, яке отримується з неї, відображається у головному вікні. На цьому етапі зручно налаштовувати позицію відеокамери. Щоб камера слідкувала за деякою територією важливо правильно її встановити, в чому і допомагає розроблена система.

Далі після натискання кнопки «Start tracking» система перейде у режим стеження. Кожна подія руху будь-якого об’єкту буде зафіксована системою і збережена у відеофайл. Відеофайли зберігаються у тому ж каталозі, де й знаходиться виконуваний файл. Стан директорії у провіднику до і після роботи зображений на рисунках 29 та 30:

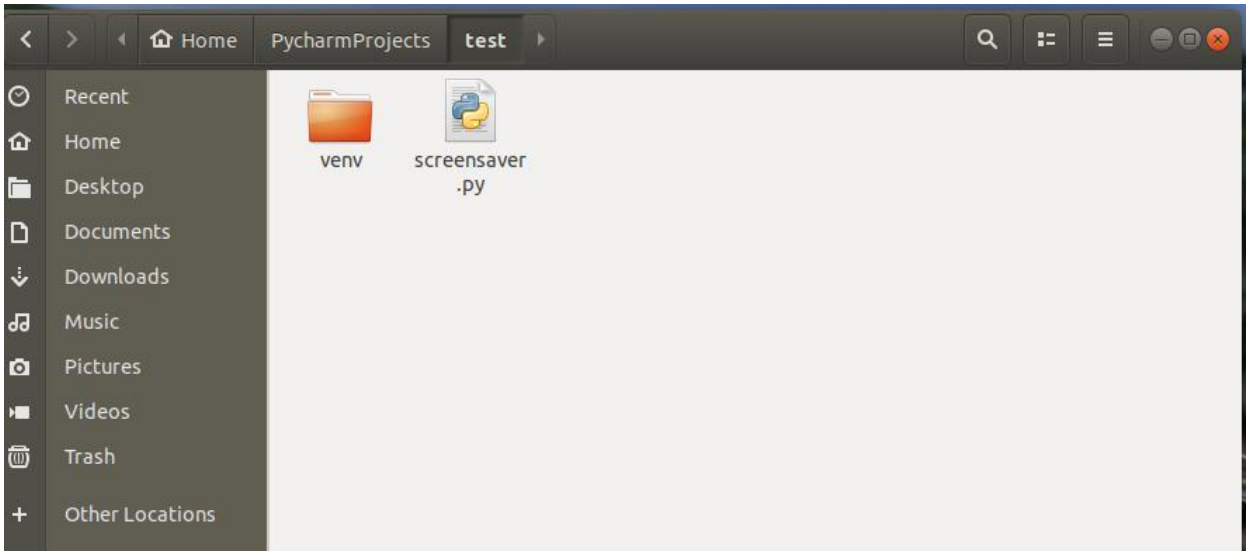

Рисунок 29 – Стан каталогу до роботи системи

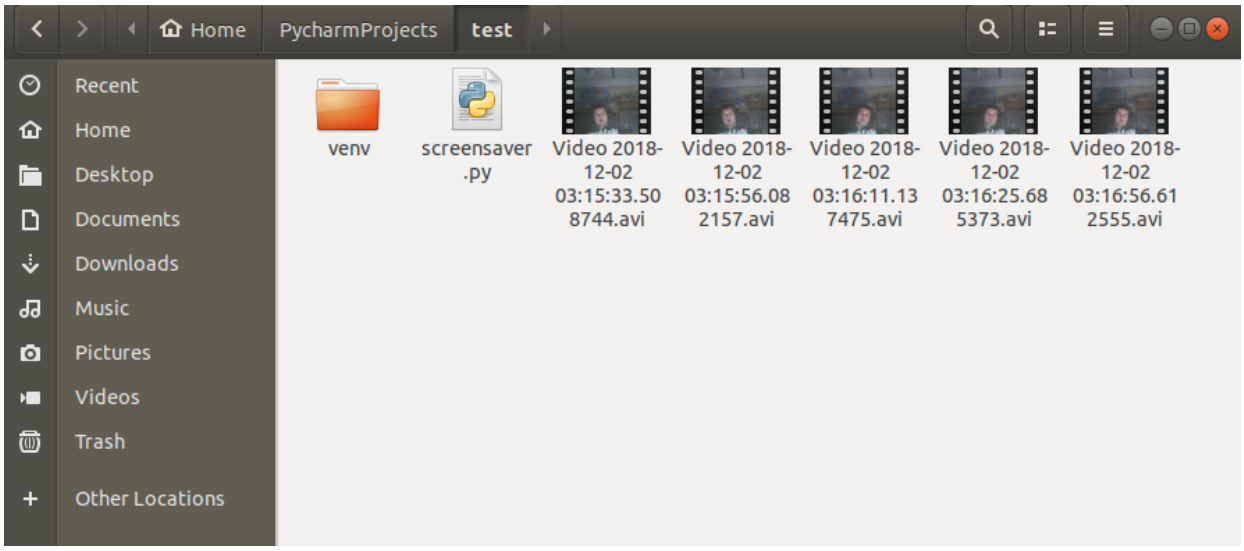

Рисунок 30 – Стан каталогу після роботи системи

Із рисунків 29 та 30 видно, що у процесі роботи додаток створює відеофайли з ім'ям «Video 'дата'.avi». Дата у назві відеофайлу відповідає даті, коли файл був створений. Тобто це точна дата початку певного епізоду, який зафіксований у відеофайлі. Таким чином дізнатись про дату настання події, під час аналізування збережених відеоматеріалів, можна за допомогою назви самого файлу. Створені відеофайли користувач має змогу переглянути будь-яким відеопрогравачем. Як приклад на рисунку 31 приведено процес перегляду відео стандартним програвачем операційної системи Ubuntu:

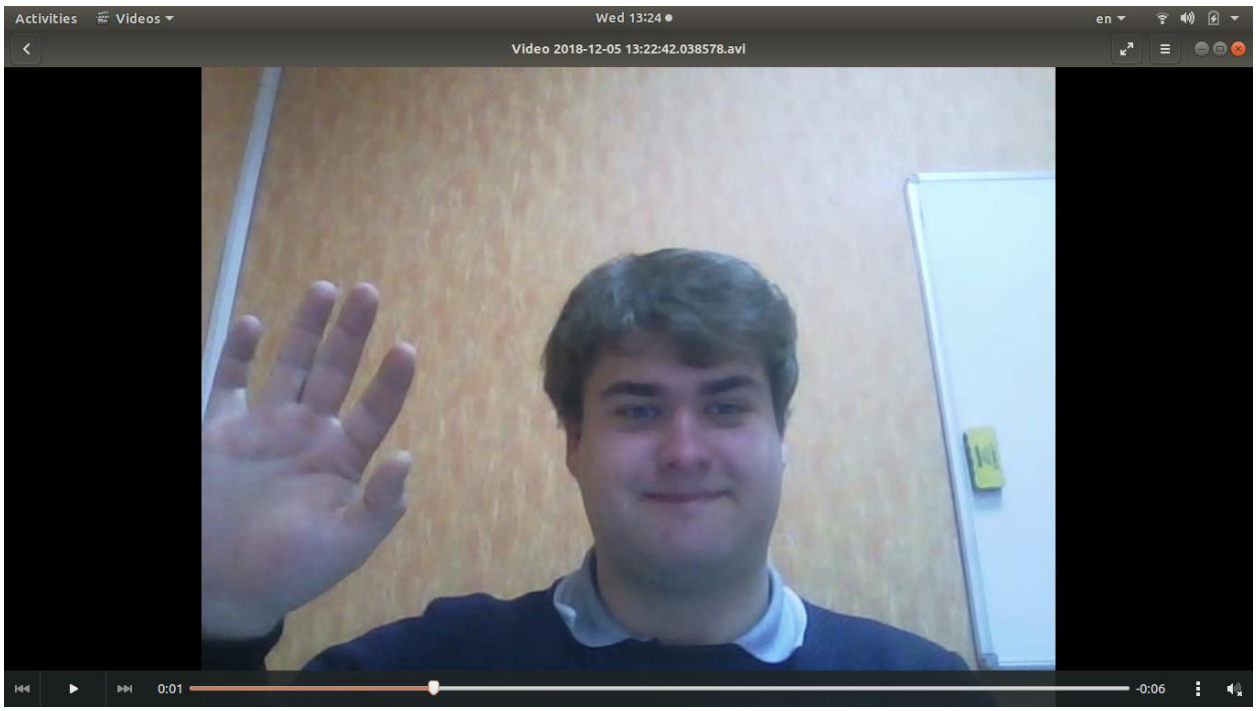

Рисунок 31 – Перегляд зафіксованих відеоматеріалів за допомогою стандартного відеопрогравача ОС Ubuntu

Якщо користувач натисне кнопку «Stop tracking» то система припинить стеження і далі запис відео з камери не відбуватиметься. Але вивід зображення з камери продовжуватиме працювати. Таким чином дії, що виконуються після натискання кнопки, переводять систему у той стан, що бачив користувач одразу ж після її запуску. Це може стати в пригоді при точному встановлення положення камери, якщо ж у процесі стеження виявлено певні недоліки.

4.2. Тестування розміру використаної пам’яті

Першим етапом тестування буде дослідження розміру використовуваної пам’яті розробленої та існуючої систем. Під існуючою системою мається на увазі саме алгоритм роботи таких систем, який описаний у п. 1.2. Дослідження відбувалось у реальних умовах.

Першою частиною цього етапу тестування є встановлення відеоспостереження у не досить людних місцях, або ж таких місцях, де будь-які події відбуваються не часто. Таким місцем було обрано звичайну кімнату

гуртожитку. Камера контролювала вхід у кімнату. Паралельно було запущено два варіанти систем стеження:

1) Розроблена система стеження.
2) Система стеження, робота якої подібна до роботи існуючих систем.

Апаратні складові систем є ідентичними. Також оскільки ці системи одночасно спостерігають за одним і тим же місцем, тобто інформація, що надходить з камери спостереження, є ідентичною протягом часу тестування, то в результаті отримуються достовірні дані, щодо розміру пам'яті, що займає кожна з систем для збереження всіх подій.

Тестування проводилось з різними проміжками часу: 30 хвилин, 3 години та 12 годин. За час тестування жодна з систем не припиняла своєї роботи.

Після тестів відбувся ретельний аналіз відеоматеріалів, та їх порівняння. Кожна з систем зберегла всі інформативні епізоди, що відбувались за час тесту. Інформативними епізодами в цьому контексті є ті події, в яких на місцевості, що спостерігалася, відбувався рух будь-яких об'єктів. Якщо певний об'єкт почав рухатись, то це може створити будь-яку небезпеку для території, саме тому такі події і необхідно фіксувати. За час стеження точна кількість таких подій, що зафіксували обидві системи, зображена у таблиці 1.

Таблиця 1 – Кількість інформативних подій, що були зафіксовані системами відеоспостереження

Система\Час	30 хвилин	3 години	12 годин
Розроблена система стеження	18	97	320
Існуюча система стеження	18	97	320

Із даної таблиці видно, що всі події були зафіксовані кожною з систем стеження. Далі наведений графік (рис. 31), що демонструє кількість використаної пам’яті для кожної з систем.

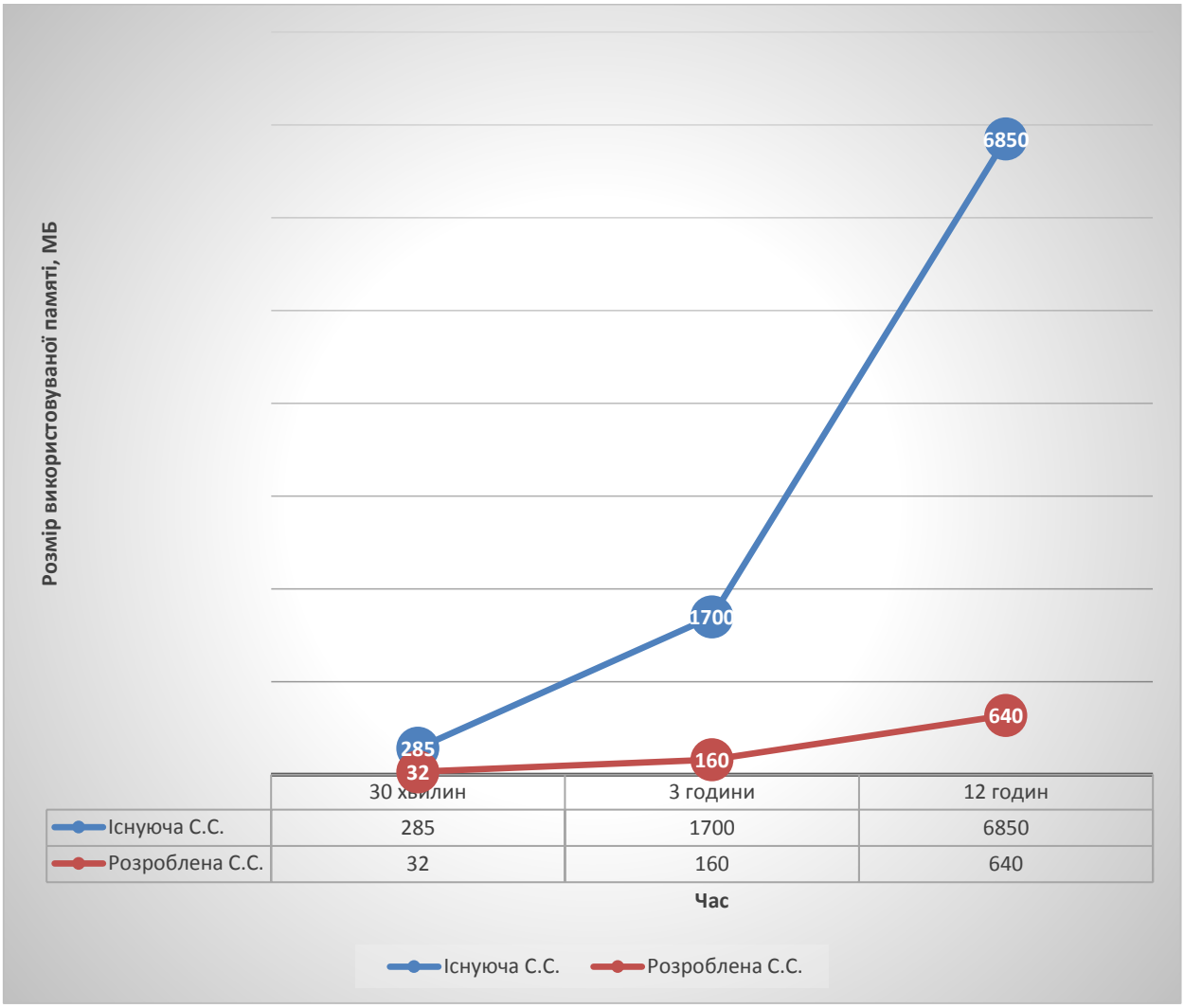

Рисунок 31 – Графік кількості затраченої пам’яті на збереження відеоматеріалів

Із даного графіку видно, що розроблена система, зберігши всю корисну інформацію, використовує майже у 10 разів менше пам’яті, порівняно з існуючими системами. Отже можна зробити висновок, що розроблена система в місцях з послабленим рухом проявляє високу ефективність щодо використання пам’яті.

Друга частина тесту передбачає встановлення такої системи у місцях з інтенсивним рухом предметів, які і будуть контролюватись системою відеоспостереження. Таким місцем можна вважати вулицю, потік машин на

дорогах, переходи в метро і подібні місця. Для тестування камеру було встановлено з вікна п’ятого поверху і направлено на вулицю. Хід тестування залишився таким же, як і в першому пункті:

- Тестування проводилось з проміжками часу у 30 хвилин, 3 години та 12 годин.
- Один і той же потік даних фіксувався двома системами: розробленою системою стеження та системою стеження, робота якої подібна до існуючої.

Скріншот роботи даної системи в таких умовах зображений на рисунку 32:

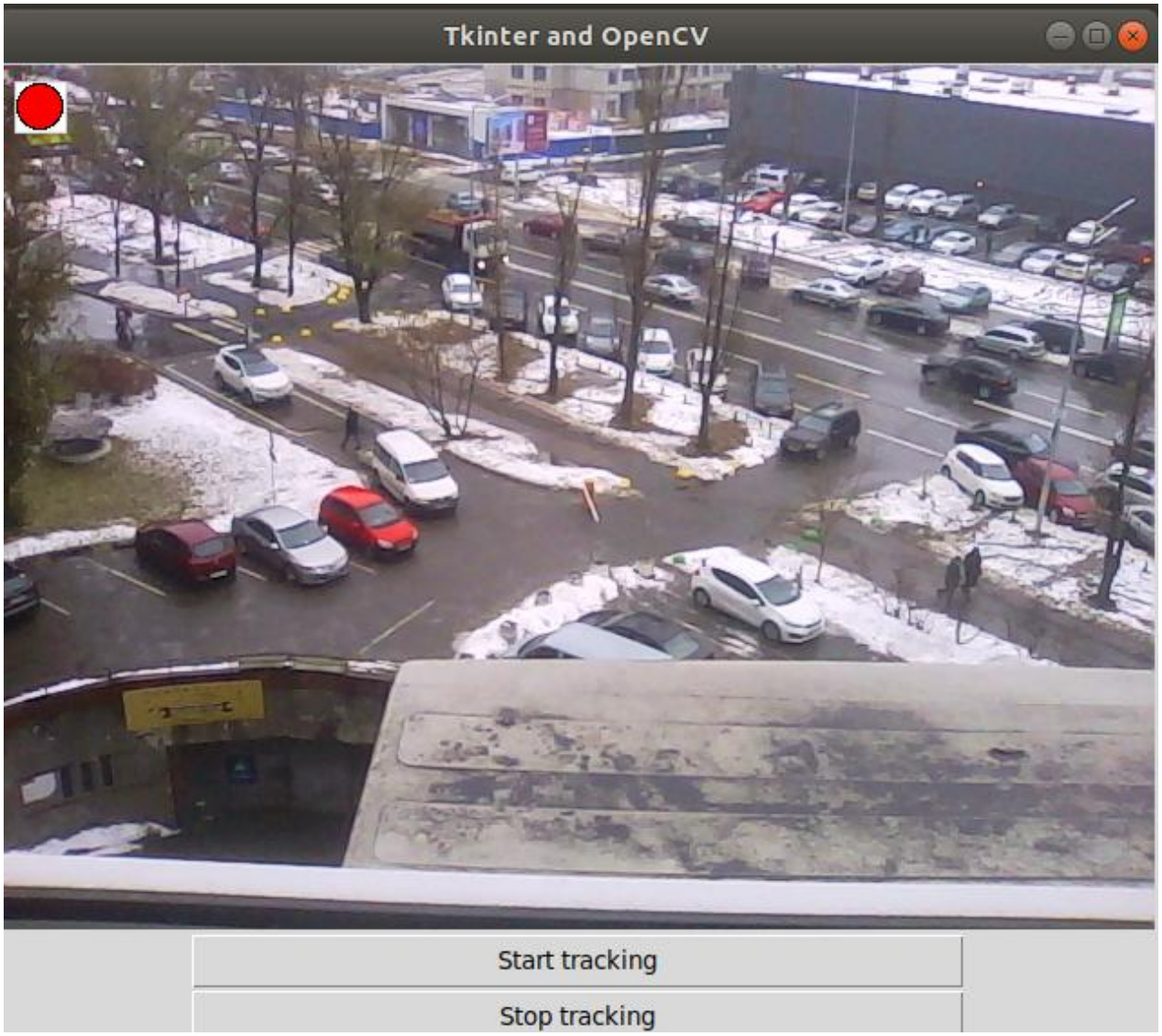

Рисунок 32 – Робота системи у місцях з інтенсивним рухом об’єктів

Результати проведення даної частини експерименту відображені на графіку (рис. 33):

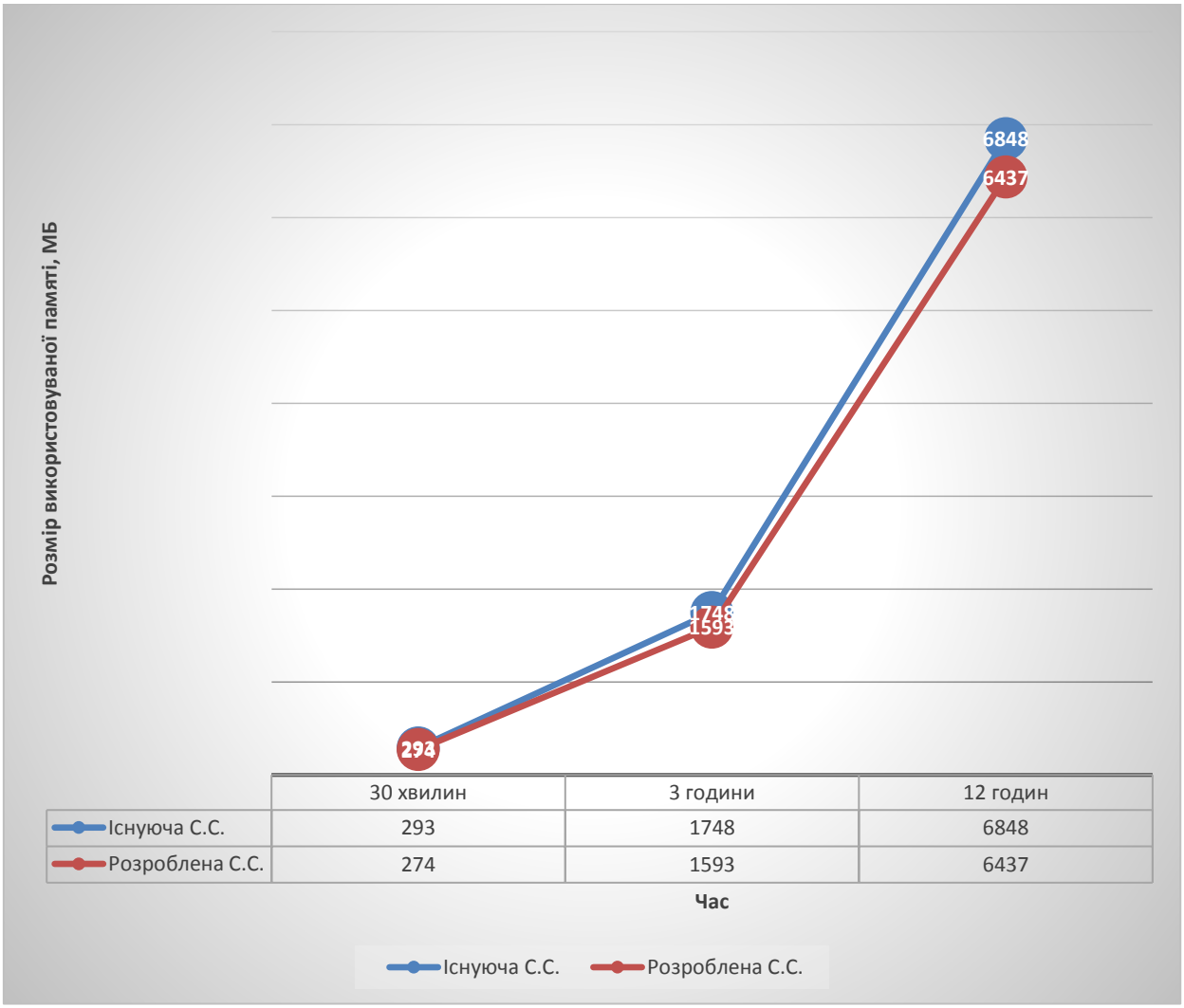

Рисунок 33 - Графік кількості затраченої пам’яті на збереження відеоматеріалів

Згідно результатів можна зробити висновок, що у місцях з посиленим рухом дана система наближується до існуючих систем у питанні використання пам’яті. Це пояснюється тим, що кожен рухомий об’єкт може спричиняти небезпеку. А оскільки таких об’єктів дуже велика кількість, то система їх усі фіксує і подій, коли рух припиняється дуже мало. Саме тому в таких випадках розроблена система майже не відрізняється від існуючих систем і зменшення використовуваної пам’яті майже непомітне.

4.3. Тестування в умовах недостатньої освітленості

Наступним етапом тестування є перевірка дієздатності системи в умовах недостатньої освітленості. Для цього тесту був використаний датчик освітленості, що вбудований у смартфон Xiaomi Redmi Note 5. Для відображення інформації з цього датчику була використана програма Light Meter від компанії My Mobile Tools Dev.

Деякі приклади освітленості представлені у таблиці 2:

Таблиця 2 – Приклади освітленості

Освітленість, люкс	Приклад
10^{-5}	Світло Сіріуса, найяскравішої зірки нічного неба
0.01	Чверть Місяця
1	Повний місяць в тропіках
50	Житлова кімната
100	Дуже темний похмурий день
320–500	Офісне освітлення
400	Схід та захід сонця в ясний день
10 000–25 000	Ясний сонячний день (в тіні)
32 000–100 000	Пряме сонячне світло
135 000	Поза атмосферою на середній відстані Землі від Сонця.

Метою даного тесту є знаходження мінімальної освітленості, при якій розроблена система демонструє свою працездатність. Мінімальна освітленість буде знаходитись експериментальним шляхом, поступово зменшуючи її до того значення, при якому система стеження буде правильно розпізнавати та фіксувати інформацію, що отримується з камери.

Результати мінімальної освітленості, а також кадр відео, що був записаний у момент тестування зображений на рисунках 34 та 35:

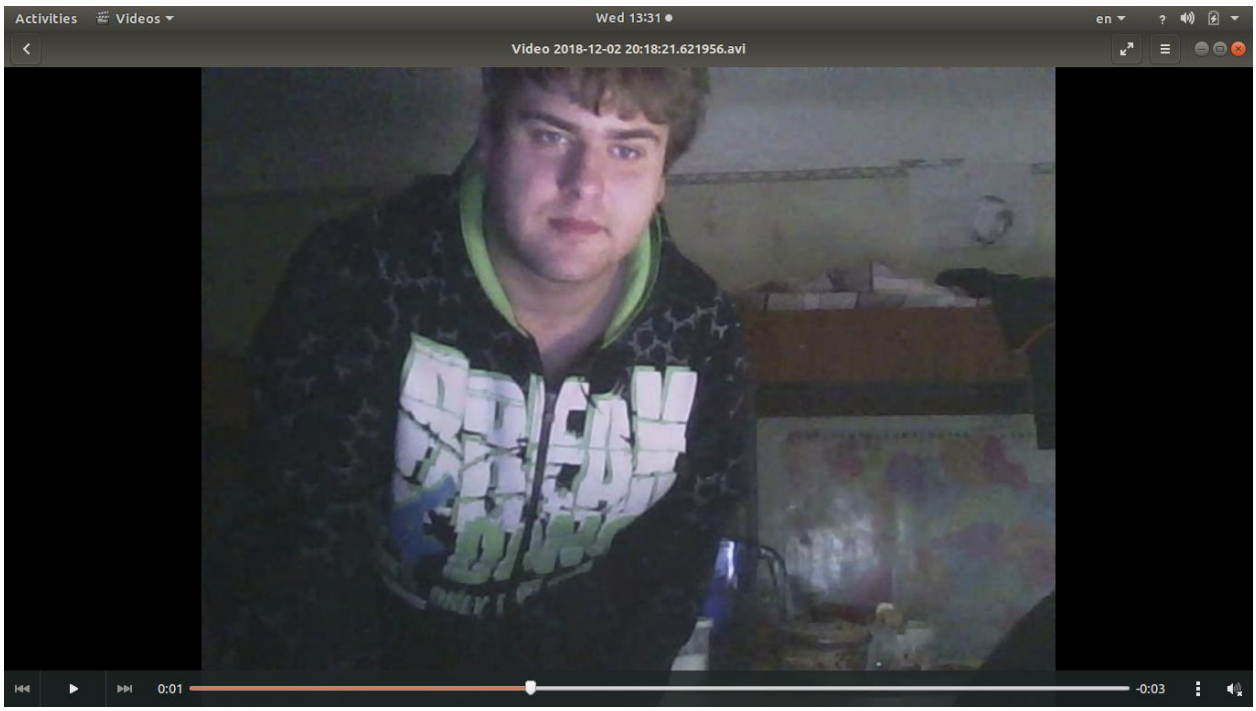

Рисунок 34 – Кадр відеоматеріалу, що був зафіксований системою стеження у недостатній видимості

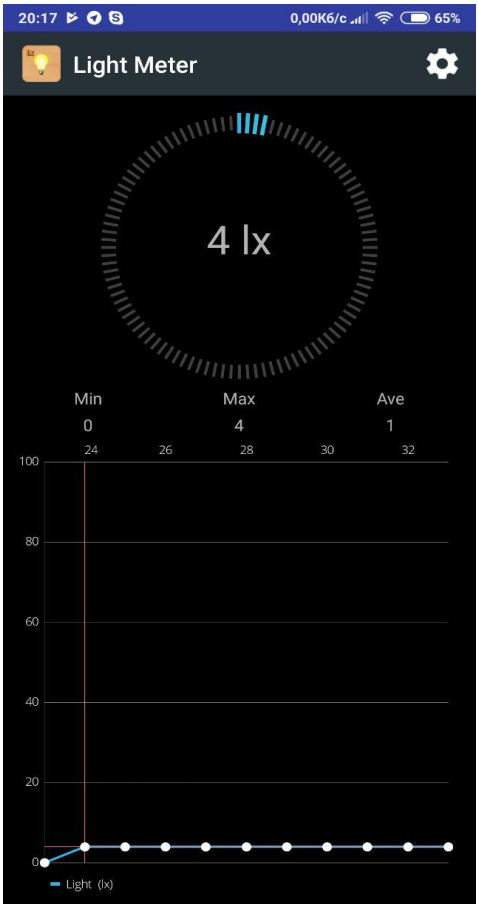

Рисунок 35 – Показання датчику освітленості під час проведення експерименту

При меншій освітленості система не завжди точно фіксує події, саме тому результатом є те значення, при якому система працює коректно.

Отже як показали результати тестування, навіть з дуже низьким освітленням додаток працює правильно. Варто також зазначити, що розробленою системою використовувалась камера, яка не призначена для роботи у низькій освітленості. Для систем стеження використовуються камери з інфрачервоним підсвічуванням. Такі камери мають змогу проводити відеозйомку навіть за повної відсутності освітлення, що є одним з ключових можливостей для систем відеонагляду, оскільки стеження має відбуватись цілодобово. Приклад зйомки кадру такою відеокамерою зображений на рисунку 36:

Рисунок 36 – Приклад зображення, що отримується камерою з підсвічуванням у абсолютній темряві

4.4. Тестування реагування системи на об’єкти, що швидко рухаються

Метою цього етапу тестування було визначення максимальної швидкості руху об’єкта, який буде зафіксований розробленою системою

відеонагляду. Дана система має один недолік: система може не реагувати на досить швидке переміщення, оскільки частоти оновлення зображення у 30 кадрів на секунду може бути недостатньо для фіксації певної події. Саме тому дослідження реакції системи на швидке переміщення об’єктів є важливим.

Тест буде проводитись наступним чином. Камеру системи стеження встановлено горизонтально. Деякий предмет із певної висоти буде падати і буде перевірятись, чи зафіксувала цей предмет камера, чи ні.

Із курсу фізики відомо, що на будь-який предмет діє сила тяжіння. Отже щоб знайти швидкість тіла, що знаходиться у вільному падінні, достатньо знати висоту, з якої було відпущено дане тіло. Формула знаходження швидкості тіла (5) виглядає таким чином:

$$v = \sqrt{2gh} \qquad (5)$$

де g – прискорення вільного падіння, h – висота, з якої тіло було випущено.

Використовуючи ці знання, проведення даного тесту суттєво спрощується. Досить визначити максимальну висоту, при якій відпущений предмет буде фіксуватись камерою. Знаючи висоту, підставити отримане значення у формулу (5) та обчислити швидкість, яка і буде максимальною для правильної фіксації системою відеонагляду. Максимальне значення висоти буде знаходитись експериментальним шляхом.

Предметом, що використовувався для тесту, був прямокутний паралелепіпед із розмірами 14х5,5х4,5 см. Саме його і було обрано для тесту, оскільки більшість об’єктів, що будуть фіксуватись системою відеоспостереження є об’ємними.

Під час тестування було визначено, що максимальною висотою, при якій фіксується даний предмет, є значення, що дорівнює 6,5 метрів на відстані 10 метрів від камери. При відпусканні предмету з більшої висоти поведінка системи є нестабільною. Саме тому максимальне значення висоти

прийнято вважати саме таким. Кадр що зафіксувала система у момент проведення тесту зображений на рисунку 37:

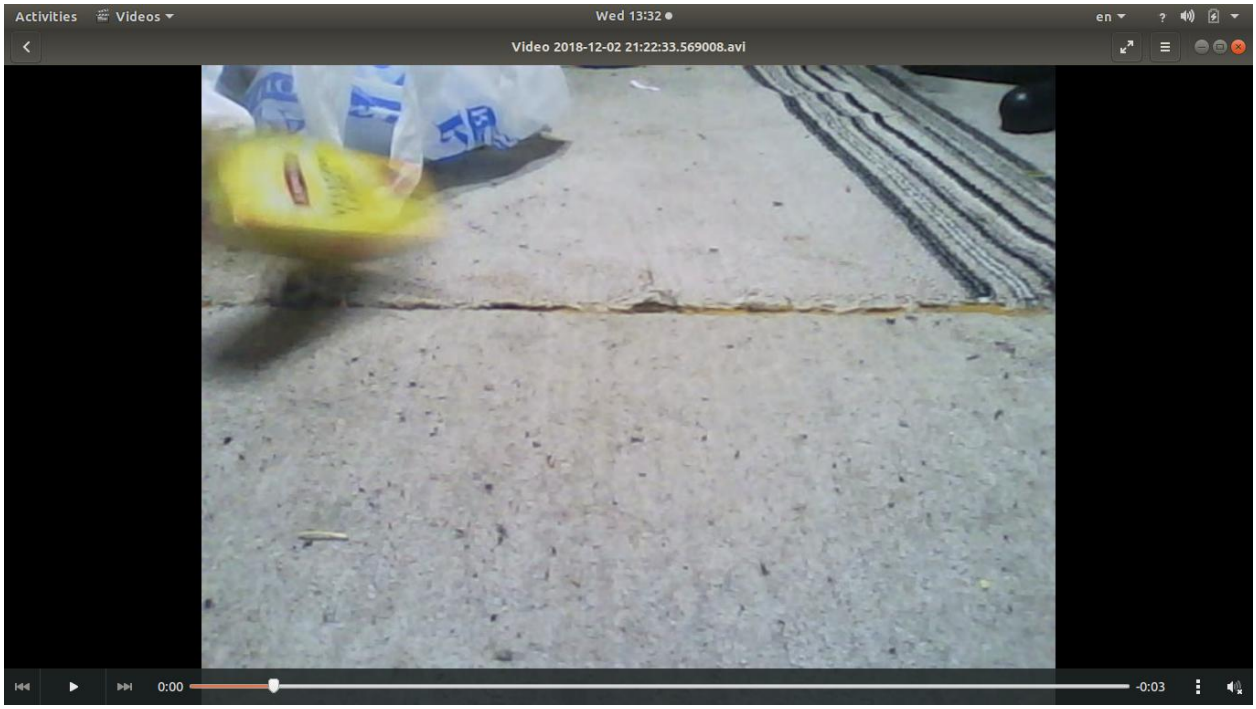

Рисунок 37 – Кадр фіксації предмету, що швидко переміщується

Підставивши значення висоти, що було отримане у ході експерименту, у формулу (5) отримаємо значення швидкості (прискорення вільного падіння – 9,8 м/с^2):

$$v = \sqrt{2 * 9.8 * 6{,}5} = 11.287 \text{ м/с} \approx 40 \text{ км/год}$$

Дана швидкість була зафіксована камерою на відстані 10 метрів від об’єктиву камери. Враховуючи середню швидкість ходьби людини, яка дорівнює 5-6 км/год, даний результат є досить значним.

Також варто зазначити, що на отриману величину швидкості суттєво впливає відеокамера. Якщо у системі застосовувати камеру з більшим показником частоти оновлення кадрів максимальна швидкість предмету, який буде зафіксований системою стеження, може суттєво збільшитись.

5. ВИСНОВОК

У даній магістерській дисертації було проаналізовано існуючі системи відеоспостереження, виявлено основні їх недоліки та розроблено програмне забезпечення, яке ці недоліки усуває.

Однією з основних причин розробки даної системи стало використання занадто великої кількості пам’яті сучасними існуючими системами стеження. Під час огляду існуючих систем було виявлено загальну проблему: всі системи виконують фіксацію постійно, навіть тоді, коли це не має сенсу. Саме цю проблему і вирішує запропонований спосіб.

Розроблена система позбавлена цього недоліку. Це досягається за рахунок того, що вона аналізує інформацію, що надходить з відеокамери та не виконує збереження у разі знаходження послідовності однакових кадрів. Саме ця функція і забезпечує досить високу економію пам’яті при розумному використанні даної системи.

Розробка проводилась з урахуванням всіх сучасних методів обробки зображень. Також розроблений продукт не є залежним від операційної системи, що значно полегшує його інтеграцію в існуючі системи відеонагляду.

До переваг даної системи можна також віднести досить зручний та мінімалістичний інтерфейс системи. Оскільки дана система буде використовуватись тільки з однією ціллю, їй не потрібний складний та заплутаний інтерфейс з безліччю налаштувань.

Ще однією вагомою перевагою є те, що розроблена система може працювати з будь-якою цифровою камерою. Для цього непотрібно вносити зміни в програму, досить лише підключити її.

Під час тестування було показано, що розроблена система стеження виконує свою роботу не гірше за інші сучасні системи відеонагляду. Але при цьому витрати пам’яті на збереження відеоматеріалів значно нижчі. Також було продемонстровано правильність роботи даної системи та її поведінку у

різних ситуаціях, де може бути застосована дана система відеоспостереження.

Практичну цінність дана розробка може мати для тих, хто вже використовує системи відеоспостереження у своїх цілях. Також це буде цікаво для виробників систем відеонагляду, оскільки продукція з такою системою не залишиться без уваги, що у свою чергу збільшить попит на подібні товари.

6. РЕКОМЕНДАЦІЇ

Базуючись на результатах проведення тестування розробленої системи відеоспостереження можна дати кілька рекомендацій щодо використання даної розробки.

Перш за все, згідно результатів експериментів, що були проведені у п. 4.2, таку систему є сенс використовувати у місцях, де рух на місцевості не є інтенсивним. Прикладом таких місць є сховища банків, архівні кімнати деяких компаній, перепускні пункти на територію, що знаходиться під контролем, тощо. Саме у подібних місцях система стеження буде максимально ефективною.

Також для забезпечення кращої роботи у складних умовах варто забезпечити систему якісною відеокамерою. Про це свідчать результати тестів, що показані у п. 4.3 та 4.4. Якщо ж система має стежити за територією і вдень, і вночі, то варто задуматись над придбанням камери з кращими характеристиками, для отримання якісних відеоматеріалів.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. *How-To: Python Compare Two Images* – [Електронний ресурс] – 2018 – Режим доступу: https://www.pyimagesearch.com/2014/09/15/python-compare-two-images/ – Дата доступу: жовтень 2018.
2. *MAE and RMSE—Which Metric is Better?* – [Електронний ресурс] – 2018 – Режим доступу: https://medium.com/human-in-a-machine-world/mae-and-rmse-which-metric-is-better-e60ac3bde13d – Дата доступу: жовтень 2018.
3. *Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli.* Image quality assessment: From error visibility to structural similarity // IEEE Transactions on Image Processing – 2004.
4. Skimage 0.14.1 docs– [Електронний ресурс] – 2018 – Режим доступу: http://scikit-image.org/docs/stable/ – Дата доступу: жовтень 2018.
5. Лутц Марк, Изучаем Python, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 1280 с., ил..
6. Модуль tkinter. Создание графического интерфейса пользователя с помощью языка программирования Python – [Електронний ресурс] – 2018 – Режим доступу: http://kabinet-vplaksina.narod.ru /olderfiles/5/Modul_tkinter.pdf– Дата доступу: жовтень 2018.
7. *Як працюють системи відеоспостереження* – [Електронний ресурс] – 2018 – Режим доступу: http://www.atlant-holding.com.ua/ua/news/64-yak-pracyuyut-sistemi-videosposterezhennya – Дата доступу: жовтень 2018.
8. *Системи аналогового відеоспостереження* – [Електронний ресурс] – 2018 – Режим доступу: http://www.bsi-group.com.ua/ua/systems-security/view/Video_analog– Дата доступу: жовтень 2018.
9. *Аналогові або цифрові камери відеоспостереження: на чому зупинитися?* – [Електронний ресурс] – 2018 – Режим доступу: http://dovidkam.com/tehnika/analogovi-abo-cifrovi-kameri-videosposterezhennya-na-chomu-zupinitisya.html– Дата доступу: жовтень 2018.
10. *IP відеоспостереження* – [Електронний ресурс] – 2018 – Режим доступу: https://xn--80adgeboqrpy5j.com.ua/ip_videosposterejennya/ – Дата доступу: жовтень 2018.
11. *Системи IP-відеоспостереження. Огляд* – [Електронний ресурс] – 2018 – Режим доступу: https://valtek.com.ua/ua/system-integration/security-control-system/video-surveillance/ip-systems-review– Дата доступу: жовтень 2018.
12. *Цифрові системи відеоспостереження* – [Електронний ресурс] – 2018 – Режим доступу: https://www.vostok.dp.ua/ukr/infa1/sistemy_vidyeonablyudeniya/digital-video/ – Дата доступу: жовтень 2018.

13. *Переваги відеоспостереження HD CCTV, HD-SDI*– [Електронний ресурс] – 2018 – Режим доступу: https://xn--80adgeboqrpy5j.com.ua/perevagu_hd/ – Дата доступу: жовтень 2018.
14. *Системи відеонагляду. Відмінність цифрової і аналогової системи відеоспостереження*– [Електронний ресурс] – 2018 – Режим доступу: http://www.ohrana-ua.com/articles/801-sistemi-vdeonaglyadu-vdmnnst-cifrovoyi-analogovoyi-sistemi-vdeosposterezhennya.html – Дата доступу: жовтень 2018.
15. *Системи безпеки для будинку* – [Електронний ресурс] – 2018 – Режим доступу: http://kristall-systems.net.ua/ua/resheniya/security_systems_for_home/ – Дата доступу: жовтень 2018.
16. *OpenCV documentation index* – [Електронний ресурс] – 2018 – Режим доступу: https://docs.opencv.org/– Дата доступу: жовтень 2018.
17. *3.3. Scikit-image: image processing*– [Електронний ресурс] – 2018 – Режим доступу: https://www.scipy-lectures.org/packages/scikit-image/index.html– Дата доступу: жовтень 2018.
18. *Getting started with Scikit-image: image processing in Python*– [Електронний ресурс] – 2018 – Режим доступу: https://www.geeksforgeeks.org/getting-started-scikit-image-image-processing-python/– Дата доступу: жовтень 2018.