WOJSKOWA AKADEMIA TECHNICZNA

im. Jarosława Dąbrowskiego

WYDZIAŁ CYBERNETYKI

ROZPRAWA DOKTORSKA

METODA STEGANOGRAFII WIDEO ODPORNA NA ZNIEKSZTAŁCENIA ANALOGOWE

mgr inż. Marcin Pery

Promotor: dr hab. inż. Tadeusz Nowicki, prof. WAT

Promotor pomocniczy: dr inż. Robert Waszkowski

Warszawa 2025

Za cierpliwość, zrozumienie i wsparcie - dziękuję mojej żonie, Pati.

Za zaufanie i pomoc w spełnieniu tego naukowego marzenia - dziękuję moim Promotorom, Panu profesorowi Tadeuszowi Nowickiemu i Panu doktorowi Robertowi Waszkowskiemu.

Za konsultację i uwagi dotyczące badań na próbie osób - dziękuję Pani doktor socjologii Uniwersytetu Warszawskiego, Izabelli Anuszewskiej.

Pracę dedykuję Tacie.

Streszczenie

Niniejsza rozprawa doktorska koncentruje się na opracowaniu i analizie metody steganografii wideo RAI (Robust Adaptive Incremental), której celem jest ukrywanie komunikatów w materiałach wideo, m.in. w formie kodów QR, w sposób zapewniający wysoką odporność, zwłaszcza na zakłócenia analogowe. Istotą proponowanego rozwiązania jest iteracyjne odzyskiwanie ukrytej informacji z kolejnych klatek wideo, co umożliwia stopniową rekonstrukcję pełnej treści ukrytej wiadomości, nawet w bardzo niesprzyjających warunkach transmisji lub odtwarzania. Zastosowany inkrementalny sposób odtwarzania danych pozwala na zdekodowanie komunikatu nawet w przypadku znaczącej degradacji jakości materiału wideo.

Metoda RAI została opracowana z myślą o dwóch kluczowych wymaganiach: odporności i niewykrywalności. Najwyższy priorytet przypisano odporności na zniekształcenia, dzięki czemu nawet intensywne zakłócenia – np. podczas wyświetlania obrazu na ekranie telewizora – nie uniemożliwiają poprawnego automatycznego odczytania ukrytej informacji. Równocześnie wprowadzone modyfikacje pozostają niewidoczne dla ludzkiego oka, co zapewnia wysoki poziom niewykrywalności.

Przeprowadzone badania eksperymentalne potwierdziły skuteczność metody RAI oraz umożliwiły określenie minimalnych parametrów kodowania i granic niewidoczności zmian dla ludzkiego wzroku. Uzyskane wyniki mogą znaleźć zastosowanie w scenariuszach steganografii wideo wymagających bezpiecznej transmisji, zwłaszcza przy bardzo trudnych warunkach technicznych transmisji sygnału lub odtwarzania obrazu.

W pracy przedstawiono również matematyczny model metody RAI, będący konkretnym przykładem implementacji ogólniejszego modelu steganografii iteracyjnej.

Zaproponowana metoda RAI może przyczynić się do rozwoju bardziej niezawodnych i odpornych technik steganografii wideo oraz wdrożenia konkretnych rozwiązań technicznych i teleinformatycznych umożliwiających realizację steganografii wideo w warunkach występowania dużych zakłóceń analogowych.

Słowa kluczowe: steganografia, steganoanaliza, teoria informacji, steganografia iteracyjna, steganografia wideo, RAI (Robust Adaptive Incremental)

Spis treści

1 Wstęp

1.1 Cel rozprawy

Niniejsza rozprawa została opracowana w celu osiągnięcia trzech zasadniczych celów. Pierwszym było dokonanie kompleksowego przeglądu aktualnego stanu wiedzy w obszarze *steganologii* ze szczególnym uwzględnieniem *steganologii wideo*. Drugim celem było formalne zdefiniowanie i opisanie problemu *zniekształceń analogowych* pomiędzy *nadawcą* i *odbiorcą komunikatu*. Trzecim i najważniejszym celem rozprawy było opracowanie autorskiej metody *steganografii wideo* cechującej się dużą odpornością na istotne i znaczące *zniekształcenia analogowe*.

Dla nowo opracowywanej metody sformułowano kilka podstawowych założeń.

Po pierwsze: miarą o najwyższym priorytecie jest *odporność*. W wyniku wystąpienia w *kanale* pomiędzy *nadawcą* a *odbiorcą* intensywnych zakłóceń o charakterze analogowym, *steganogram* wysłany przez *nadawcę* poddawany jest różnym i wielokrotnym przekształceniom oraz zakłóceniom. Pomimo to *odbiorca* musi mieć możliwość odczytania zakodowanego w *steganogramie* pierwotnego *komunikatu*.

Po drugie: miarą o niższym priorytecie, ale wciąż bardzo istotną dla nowo opracowywanej metody, jest *niewykrywalność*. W przypadku nowo opracowywanej metody najistotniejszą częścią *kanału* komunikacji pomiędzy *nadawcą* i *odbiorcą* jest część analogowa, kiedy *steganogram* i *komunikat* w nim zakodowany są przesyłane nie w postaci pliku cyfrowego, ale w postaci transmisji analogowej, np. w postaci wyświetlania obrazu na ekranie telewizora. Dlatego *niewykrywalność* rozumiana jest w tym przypadku jako brak możliwości odkrycia ludzkim wzrokiem faktu modyfikacji *steganograficznej* wyświetlanego obrazu *wideo*. Wskaźnik *niewykrywalności* powinien być na akceptowalnym poziomie.

Po trzecie: miarą o najniższym priorytecie dla nowo opracowywanej metody jest *pojemność*. Oczywiście pożądane jest, aby wskaźniki *pojemności* były jak największe, ale ze względu na bardzo duże wymagania dotyczące zwłaszcza *odporności* należy zaakceptować, że opracowywana metoda będzie optymalizowała priorytetowe miary *odporności* i *niewykrywalności* kosztem *pojemności*.

Opracowana w ramach niniejszej rozprawy oryginalna metoda steganograficzna *RAI* umożliwia zakodowanie w plikach wideo informacji w formie *QR* kodu, reprezentującego

ukrytą wiadomość tekstową. Procedura ta realizowana jest iteracyjnie, wydobywając z każdej klatki kolejną porcję ukrytej informacji i stopniowo rekonstruując pełny obraz *QR* kodu, aż do momentu uzyskania jego czytelnej postaci, pozwalającej na odczytanie zakodowanego *komunikatu*. Metoda *RAI* cechuje się bardzo dużą *odpornością* na zniekształcenia i zakłócenia obrazu wideo nawet w skrajnie niekorzystnych warunkach, takich jak np. odtwarzanie materiału wideo na ekranie telewizora z dużymi zakłóceniami obrazu, umożliwiając zdekodowanie ukrytej wiadomości w możliwie jak najkrótszym czasie. Jednocześnie *steganograficzne* modyfikacje w plikach *wideo* wprowadzane przez metodę *RAI* pozostają niezauważalne dla ludzkiego oka.

1.2 Zakres pracy

Niniejsza rozprawa została podzielona na pięć głównych części.

W części pierwszej wyszczególniono cel pracy, jej zakres oraz poczyniono niezbędne uwagi redakcyjne.

W drugiej części zostały omówione podstawowe zagadnienia i pojęcia związane ze *steganologią*, zaprezentowano stosowane w literaturze przedmiotu klasyfikacje, wykorzystywane techniki i algorytmy oraz miary. Dokonano przy tym kompleksowego przeglądu literatury naukowej związanej ze *steganologią* wskazując najważniejsze monografie i artykuły naukowe opublikowane na przestrzeni ostatnich lat. Wyszczególniono zarówno prace o charakterze przeglądowym, jak też proponujące konkretne metody, techniki i podejścia do omawianych zagadnień. Autorowi szczególnie zależało na tym, aby było to kompleksowe uporządkowanie obecnego stanu wiedzy na temat *steganologii* w ogóle, a nie tylko stanowiło wprowadzenie do zasadniczego tematu pracy. Dlatego umieszczono w niniejszej pracy liczne przykłady odnoszące się do różnych zagadnień związanych z obszarem *steganografii* i *steganoanalizy*, w tym napisane przez autora kody algorytmów, które szczegółowo ilustrują poruszane kwestie.

W części trzeciej przedstawiono genezę głównego problemu, który rozpatrywany jest w niniejszej pracy. Na tle specyficznych zagadnień związanych ze *steganologią wideo*, odnosząc się jednocześnie do najważniejszych pozycji literatury naukowej z tego zakresu, pokazano na czym polega problem *zniekształceń analogowych* w przypadku *steganogramów wideo* oraz jakie niesie to konsekwencje i wyzwania dla projektowanych technik

steganografii i *steganoanalizy*. Zidentyfikowano również lukę badawczą, zdefiniowano problem badawczy oraz postawiono tezę badawczą.

Czwarta część zawiera propozycję autorskiej metody *steganografii wideo* adresującą problem *zniekształceń analogowych* wraz z prezentacją wyników przeprowadzonych eksperymentów badających właściwości proponowanej metody. Badania przeprowadzono w dwóch obszarach. Pierwszy zakres badań dotyczył specyficznych właściwości *nośników* i *steganogramów* jakimi są pliki *wideo* w kontekście rozpatrywanego problemu badawczego, natomiast drugi obszar badań odnosił się do wrażliwości percepcji ludzkiego oka na zmiany wprowadzane przez proponowaną technikę *steganograficzną* w *nośnikach wideo*. Na koniec omówiono uzyskane wyniki eksperymentów badawczych.

W piątej części przedstawiono wnioski z przeprowadzonych prac badawczych oraz zaproponowano dalsze kierunki badań, jak też możliwych prac rozwojowych, które dotyczą zarówno zaproponowanej metody, jak też odnoszą się w sposób ogólniejszy do badań nad problemem *zniekształceń analogowych* różnych metod *steganografii*.

Niektóre z zagadnień poruszanych w rozprawie zostały już wcześniej zaprezentowane w innych pracach autora:

1. *(Pery, Zastosowanie steganologii w cyberbezpieczeństwie, 2024)*, w której dokonano m.in. przeglądu pojęć związanych z obszarem *teorii informacji*, *steganologii*, *steganografii*, *steganoanalizy* i *kryptologii*, zaprezentowano skrócony rys historyczny rozwoju *steganologii* oraz omówiono kwestię zastosowania *steganologii* w *cyberbezpieczeństwie*.
2. *(Pery i Waszkowski, Computational System for Evaluating Human Perception in Video Steganography, 2024)*, w której zaprezentowano architekturę i przykład realizacji systemu informatycznego, badającego różnice między ludzką percepcją a detekcją algorytmiczną. Przedstawiony system pozwala m.in. określić próg *niewykrywalności steganogramów wideo* dla ludzkiego oka oraz oszacować minimalny *poziom kodowania* potrzebny do automatycznego dekodowania ukrytych *informacji*.
3. *(Pery i Waszkowski, A Model and Quantitative Framework for Evaluating Iterative Steganography, 2024)* w której zdefiniowano *steganografię iteracyjną* oraz funkcję *IIF (Incremental Information Function)* wykorzystywaną do mierzenia efektów działania technik *steganograficznych* w scenariuszach *steganografii iteracyjnej*.
4. *(Pery i Waszkowski, Mathematical modeling in color difference analysis in consecutive video frames for steganography, 2024)*, w której zaproponowano matematyczny model *nośników wideo*.

5. *(Pery i Waszkowski, Analyzing natural pixel color variations in consecutive video frames to enhance spatial domain video steganography, 2024)*, w której omówiono kwestię właściwości *nośników wideo* w kontekście możliwości wykorzystania różnic w następujących po sobie *ramkach wideo* w technikach *steganografii przestrzennej*.

1.3 Uwagi redakcyjne

Używane w niniejszej pracy istotne pojęcia pisane są *kursywą*, a ich definicje są umieszczone w postaci przypisów dolnych oraz dodatkowo umieszczone w postaci spisu alfabetycznego na końcu pracy.

Definicje omawianych w pracy pojęć, wzorów i miar zilustrowano przykładami, w których jako *nośnik* oraz ukrywany *komunikat* wykorzystano zdjęcia wykonane przez autora niniejszej rozprawy. W celu uzyskania większej przejrzystości sekcje, w których przedstawiane są przykłady, wyróżniono ogranicznikami ***"Przykład"*** i ***"Koniec przykładu"***.

```
      Przedstawione w pracy algorytmy zostały zilustrowane napisanymi przez
autora kodami w języku Python w wersji 3.12.4 oraz wyróżnione formatem
identycznym jak w niniejszym akapicie. Kody napisane w języku Python
wykorzystują następujące biblioteki:

import cv2                        #version 4.10.0
import numpy as np                #version 2.0.1
from PIL import Image             #version 10.4.0
from math import log10, sqrt      #version 3.12.4
```

Na końcu pracy znajdują się spisy zdefiniowanych i używanych w rozprawie pojęć, rysunków, tabel oraz kodów algorytmów.

Niektóre ze szczegółowych wyników przeprowadzonych eksperymentów badawczych z przyczyn redakcyjnych nie zostały włączone do zasadniczej części rozprawy, a zostały umieszczone w załączniku dostępnym w internetowym repozytorium pod adresem https://github.com/MarcinPery/Rozprawa.

Wszystkie niezbędne dane umożliwiające ewentualną replikację uzyskanych rezultatów omawianych w niniejszej pracy zostały umieszczone w zasadniczej części rozprawy lub w załączniku.

2 Wprowadzenie do steganologii

W niniejszym rozdziale przedstawiono najważniejsze definicje oraz omówiono podstawowe pojęcia związane m.in. z *teorią informacji*[1] w kontekście badań nad *steganologią*[2].

Rozwinięto w nim również pojęcia *steganografii*[3] i *steganoanalizy*[4] zarówno od strony teoretycznej, jak i praktycznej, omawiając przykłady zastosowań konkretnych technik oraz miar służących ocenie jakości metod *steganograficznych* i *steganoanalitycznych*.

Wszystkie rozważania teoretyczne osadzono w kontekście przeglądu literatury naukowej, w której poruszano zagadnienia omawiane w pracy.

2.1 Pojęcie informacji

Centralnym pojęciem w dziedzinie *steganologii* jest *informacja*[5], która jest przedmiotem ukrywania w procesie *steganografii* i wykrywania w procesie *steganoanalizy*. W literaturze istnieje wiele podejść do zrozumienia i definicji pojęcia *informacji*.

W pracy H. Nyquista *"Certain Topics in Telegraph Transmission Theory" (Nyquist, 1928)* przedstawiono kluczowe postulaty dotyczące sposobów transmisji sygnałów telegraficznych, które później stały się fundamentem *teorii informacji*. Nyquist określił m.in. kryterium bezstratnej transmisji, wprowadził pojęcie szybkości przekazywania *informacji*

---

[1] **Teoria informacji** - dziedzina nauki zajmująca się kwantyfikacją, przechowywaniem, przesyłaniem i przetwarzaniem *informacji*, wykorzystująca matematyczne modele i narzędzia do analizy transmisji sygnałów, kompresji danych, przetwarzania i ukrywania *informacji (Shannon, 1948; Anderson i Johannesson, 2006)*.

[2] **Steganologia** - dziedzina nauki obejmująca dwa obszary: *steganografię* i *steganoanalizę*.

[3] **Steganografia** - proces ukrywania *komunikatów* w *steganogramach* w taki sposób, aby osoby postronne nie były tego świadome.

[4] **Steganoanaliza** - proces identyfikacji oraz dekodowania *komunikatu* zakodowanego w *steganogramie* przez procesy *steganograficzne*.

[5] **Informacja** - zmniejszenie niepewności związane z wystąpieniem konkretnego zdarzenia w systemie probabilistycznym kwantyfikowana przez pojęcie *entropii* zdefiniowane w *teorii informacji* Shannona *(Shannon, 1948; Anderson i Johannesson, 2006)*.

oraz analizował zagadnienie zniekształcenia sygnałów i wpływ *szumów*[6] na transmisję *informacji,* proponując metody ich minimalizacji poprzez stosowanie odpowiednich filtrów. Jego praca miała znaczący wpływ na rozwój telekomunikacji oraz *teorii informacji*, a opisane przez niego podstawowe zasady są nadal stosowane w nowoczesnych *systemach komunikacyjnych*[7].

C. Shannon w swojej fundamentalnej pracy *"A Mathematical Theory of Communication" (Shannon, 1948)* zdefiniował pojęcie *informacji* jako miarę redukcji niepewności, która występuje, gdy *odbiorca* otrzymuje od *nadawcy* wiadomość. W teorii Shannona *informacja* pozbawiona jest kontekstu, przez co jest ″semantycznie agnostyczna″.

Shannon wprowadził również pojęcie *entropii informacji*[8] kwantyfikującej ilość niepewności związanej z wyborem jednej z wielu możliwych wiadomości. W swojej pracy Shannon opisał podstawowe elementy *systemu komunikacyjnego*, którego schemat przedstawiony jest na rysunku 1. *System komunikacyjny* złożony jest z *nadawcy*[9], *nadajnika*[10], *kanału*[11], *odbiornika*[12] i *odbiorcy*.[13] *Nadawca* generuje wiadomość, która jest przekształcana przez *nadajnik* w sygnał przesyłany przez *kanał*. *Kanał* stanowi medium, przez które sygnał jest przesyłany, a *odbiornik* przekształca sygnał z powrotem w wiadomość, która jest dostarczana do *odbiorcy*.

---

[6] **Szum** - mogące pochodzić z różnych źródeł niepożądane zakłócenia w przekazie *informacji*, które obniżając jej jakość i dokładność, utrudniają lub zniekształcają jej odbiór oraz interpretację *(Shannon, 1948)*.

[7] **System komunikacyjny** - struktura, która umożliwia przesyłanie *informacji* między *nadawcą* a *odbiorcą* składająca się z takich elementów jak: *nadawca*, *kanał transmisji*, *odbiorca* oraz procesy kodowania, przesyłania i dekodowania *informacji (Nyquist, 1928; Shannon, 1948).*

[8] **Entropia** - miara nieuporządkowania *informacji* - im większa *entropia*, tym większa nieprzewidywalność i niepewność *informacji (Shannon, 1948)*.

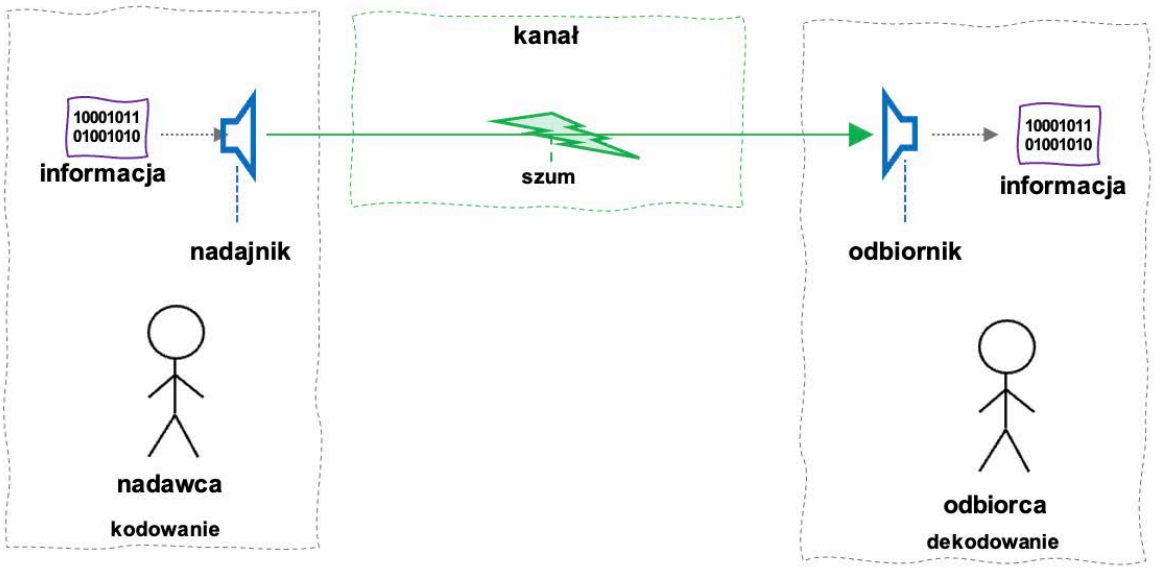
[9] **Nadawca** - element *systemu komunikacyjnego*, który generuje, koduje i przekazuje *informacje* poprzez wybrany *kanał* do *odbiorcy*.

[10] **Nadajnik** - element *systemu komunikacyjnego*, który przekształca zakodowaną *informację* od *nadawcy* w sygnał odpowiedni do przesłania przez *kanał.*

[11] **Kanał** - medium lub droga, przez którą sygnał zawierający zakodowaną *informację* jest przesyłany od *nadajnika* do *odbiornika*.

[12] **Odbiornik** - element *systemu komunikacyjnego*, który odbiera sygnał z *kanału*, dekoduje go i przekształca w *informację* zrozumiałą dla *odbiorcy*.

[13] **Odbiorca** - element *systemu komunikacyjnego*, który odbiera, dekoduje i interpretuje *informacje* przekazane przez *nadawcę* poprzez wybrany *kanał*.



Rysunek 1 - Schemat *systemu komunikacyjnego*
Źródło: *Opracowanie własne* na podstawie *(Shannon, 1948)*

Shannon wprowadził również pojęcie *pojemności kanału*[14] określające maksymalną ilość *informacji*, jaką można przesłać przez *kanał* z minimalnym prawdopodobieństwem błędu, pomimo obecności *szumów*. Shannon przypisał także terminowi *bit*[15] miarę dwustanowej, najmniejszej jednostki *informacji*.

Zależności pomiędzy danymi, *informacją*, wiedzą i mądrością można przedstawić również w sposób uwzględniający kontekst i semantykę, np. w formie klasycznej piramidy dane - *informacja* - wiedza - mądrość *DIKW*[16] opisanej przez R. Ackoffa *(Ackoff, 1989)*. Przedstawiona na rysunku 2 piramida *DIKW* obrazuje to, w jaki sposób surowe dane przekształcają się w *informację*, następnie w wiedzę, a na końcu w mądrość, stanowiąc różne poziomy zrozumienia i przetwarzania *informacji*. Ackoff zwraca uwagę na hierarchiczną

[14] **Pojemność kanału** - maksymalna ilość *informacji*, którą *kanał* może przesłać w jednostce czasu, określająca wartość przepustowości *systemu komunikacyjnego (Shannon, 1948; Cover i Thomas, 1991)*.

[15] **Bit** - podstawowa jednostka *informacji* w *systemie komunikacyjnym*, która przyjmuje jedną z dwóch możliwych wartości: 0 lub 1, używana do kodowania, przesyłania i przetwarzania *informacji* w *kanałach cyfrowych (Shannon, 1948)*.

[16] **DIKW (Data Information Knowledge Wisdom)** - klasyczna piramida: *dane, informacja, wiedza, mądrość*. Dane to surowe, nieprzetworzone fakty i liczby bez kontekstu. *Informacje* to dane zorganizowane i przetworzone, które nabierają znaczenia. Wiedza to *informacje*, które zostały zrozumiane i połączone z doświadczeniem i kontekstem. Mądrość to praktyczne zastosowanie wiedzy, prowadzące do mądrych decyzji *(Ackoff, 1989; Rowley, 2007)*.

naturę tych pojęć oraz na procesy, które pozwalają na transformację danych w wartościową wiedzę.

Rysunek 2 - Klasyczna piramida *DIKW*
Źródło: *Opracowanie własne* na podstawie *(Ackoff, 1989)*

Piramida *DIKW* pokazuje, że dane same w sobie nie mają istotnego znaczenia, dopóki nie zostaną przetworzone i zinterpretowane. W kontekście *steganologii* przekształcanie danych w *informację* ma kluczowe znaczenie, ponieważ ukrywane są nie surowe dane, ale dane, które nabrały znaczenia przez ich interpretację i nadanie im kontekstu. Piramida *DIKW* pomaga również zrozumieć to, jak przetwarzanie *informacji* prowadzi do tworzenia wiedzy, która jest podstawą mądrości, umożliwiającej podejmowanie świadomych decyzji. Takie podejście jest niezbędne dla skutecznego ukrywania i odkrywania *informacji*.

Teorię Shannona rozwinęli T. Cover i J. Thomas w książce *"Elements of Information Theory" (Cover i Thomas, 1991)*. Ich praca, która obejmuje m.in. podstawowe koncepcje takie jak *entropia*, *kompresja bezstratna*[17], *pojemność kanału* oraz kodowanie z korekcją błędów, podkreśla centralną rolę *entropii,* jako miary niepewności w zmiennej losowej, która jest kluczowa dla kwantyfikacji *informacji*. Autorzy analizują *kompresję bezstratną* oraz pokazują to, w jaki sposób założenia teorii *entropii* wyznaczają maksymalne granice

[17] **Kompresja bezstratna** - metoda redukcji rozmiaru danych, polegająca na transformacji *informacji* do postaci o zmniejszonym zapotrzebowaniu na liczbę *bitów*, przy jednoczesnym zapewnieniu możliwości pełnego odtworzenia oryginalnej treści bez jakichkolwiek strat.

kompresji bezstratnej. Omawiają także koncepcję *pojemności kanału* określającą maksymalną szybkość transmisji *informacji* i pokazują to, w jaki sposób kodowanie z korekcją błędów może zapewnić bezstratną komunikację nawet w obecności *szumów*. Autorzy przedstawiają teorię oceny zniekształceń analizując kompromis między kompresją danych, a utratą *informacji* oraz rozszerzają założenia *teorii informacji* na wykorzystanie w sieciach komunikacyjnych.

Z kolei L. Floridi w swojej pracy *"Information: A Very Short Introduction" (Floridi, 2010)* prezentuje kilka fundamentalnych tez dotyczących natury *informacji*, które mają szerokie implikacje praktyczne. Floridi argumentuje, że *informacja* nie jest jedynie abstrakcyjnym pojęciem, lecz stanowi fundamentalny aspekt rzeczywistości. Według Floridiego *informacja* odgrywa kluczową rolę w kształtowaniu współczesnego świata i ludzkiej percepcji rzeczywistości. Floridi podkreśla, że *informacja* istnieje na różnych poziomach: od surowych danych, które stanowią podstawowe fakty, przez *informacje*, które są danymi zinterpretowanymi w odpowiednim kontekście, aż po wiedzę, która integruje informacje z doświadczeniem i głębszym zrozumieniem. Taka struktura podkreśla złożoność i wieloaspektowość *informacji* jako kluczowego elementu rzeczywistości. Floridi zwraca także uwagę na relacyjny charakter *informacji* twierdząc, że *informacja* nie istnieje samoistnie, lecz powstaje w wyniku interakcji między danymi, kontekstem i interpretacją. W tej perspektywie *informacja* jest dynamicznym i zależnym od kontekstu bytem, co ma istotne konsekwencje dla sposobu, w jaki ją przetwarzamy i wykorzystujemy.

Informacja w kontekście *steganologii* jest zatem fundamentalnym pojęciem, który wymaga od *nadawcy* i *odbiorcy* precyzyjnej umowy, w jaki sposób interpretują i przetwarzają oni przesyłane do siebie dane. Interesującym aspektem jest fakt, iż brak danych, zwłaszcza gdy są one oczekiwane przez *odbiorcę*, również może stanowić *informację*. W *steganologii* aspekt ten jest szczególnie ważny, ponieważ ukrywanie *informacji* często polega na manipulacji obecnością lub brakiem obecności określonych danych w *nośniku*[18].

[18] **Nośnik** - oryginalny plik cyfrowy, w którym w wyniku zastosowania technik *steganograficznych* może być ukryty *komunikat*.

W niniejszej rozprawie stosowany jest termin *komunikat*[19], który oznaczać będzie wszelkie dane lub brak danych, które są lub mają być przetwarzane w sposób zrozumiały i jednoznacznie interpretowalny przez *nadawcę* i *odbiorcę*. *Komunikat* będzie zatem rozumiany jako *informacja*, którą *nadawca* będzie chciał przekazać w sposób sekretny *odbiorcy*.

2.2 Podstawy steganografii

Termin *steganografia* pochodzi z języka greckiego, gdzie *"steganos"* oznacza *"ukryty"* lub *"tajny"*, a *"graphein"* oznacza *"pisanie"* lub *"rysowanie"*. W literaturze przedmiotu *steganografia* jest definiowana na różne sposoby odzwierciedlające jej złożoność i wszechstronność, ale wszystkie te podejścia łączy jeden główny paradygmat: podstawowym założeniem *steganografii* jest ukrywanie tajnych *informacji* w jawnych *nośnikach* w taki sposób, aby osoby postronne nie były tego świadome.

W książce D. Kahna *"The Codebreakers: The Story of Secret Writing"* z 1967 roku *(Kahn, 1967) steganografia* przedstawiona jest jako praktyka i studium ukrywania *informacji*. Kahn podkreślił w niej praktyczną stronę *steganografii* uwypuklając historyczne aspekty jej rozwoju oraz pokazując różne techniki *steganograficzne* stosowane na przestrzeni tysiącleci. Jego praca dostarcza bogatego kontekstu historycznego pokazując to, w jaki sposób *steganografia* była używana od starożytności do współczesności. Autor omawia różne metody ukrywania *informacji*, np. takie jak atrament sympatyczny (technika *steganograficzna* polegająca na zapisywaniu *informacji* niewidocznych gołym okiem przy użyciu specjalnych roztworów chemicznych) oraz mikrokropki (technika ukrywania *informacji* polegająca na miniaturyzacji danych do postaci punktu o średnicy około 1 mm w skali 1:300 wykorzystując specjalne urządzenie łączące funkcje aparatu fotograficznego i mikroskopu). Książka Kahna jest często cytowana jako jedno z pierwszych kompleksowych opracowań na temat *steganografii*, które łączy omówienie aspektów technicznych z historycznymi przykładami ich zastosowania. Praca ta jest jednocześnie uznawana za jedno z najbardziej kompleksowych opracowań historii tej dziedziny nauki.

[19] **Komunikat** - *informacja*, którą *nadawca steganogramu* chce w sposób sekretny przekazać *odbiorcy*.

W drugiej połowie XX wieku rozwój technologii cyfrowych stał się istotnym impulsem do rozwoju technik *steganograficznych* wykorzystujących pliki cyfrowe jako *nośniki* ukrywanych *informacji*. Pojawiły się wówczas pierwsze kompleksowe przeglądowe publikacje naukowe z zakresu *steganologii* cyfrowej.

W 1996 roku po konferencji *"Information Hiding - First International Workshop"* opublikowano zbiorowe opracowanie pod redakcją R. Andersona *(Anderson R., Information Hiding, 1996)*, które jest zbiorem artykułów dotyczących technik ukrywania *informacji*, w tym głównie *steganografii*. Prace przedstawione w tym opracowaniu analizują możliwości i ograniczenia różnych metod *steganograficznych* oraz ich potencjalne zastosowania w bezpieczeństwie komputerowym.

W 1998 roku ukazały się dwa ważne dla rozwoju *steganografii* artykuły. Pierwszym była opublikowana przez N. Johnsona i S. Jajodii praca pod tytułem *"Exploring Steganography: Seeing the Unseen" (Johnson i Jajodia, Exploring Steganography: Seeing the Unseen, 1998)*, w której przedstawiono definicję *steganografii,* jako metody ukrywania *informacji* w sposób zapobiegający wykryciu jej obecności, podkreślając jej kluczową cechę - maskowanie samego faktu przekazywania *komunikatu*, a nie tylko jego treści. W swojej pracy Johnson i Jajodia zaprezentowali również historię *steganografii*, wyjaśnili jej podstawowe paradygmaty oraz opisali podstawowe techniki *steganograficzne* wykorzystujące modyfikację danych cyfrowych, ze szczególnym uwzględnieniem *steganografii obrazowej*. Ich analiza objęła również przegląd oprogramowania służącego do przetwarzania obrazów w kontekście kodowania *steganograficznego*. Z kolei artykuł R. Andersona i F. Petitcolasa *"On the limits of steganography" (Anderson i Petitcolas, On the limits of steganography, 1998)* badał ograniczenia efektywności *steganografii* w kontekście *niewykrywalności i odporności* na zniekształcenia sygnału. Autorzy porównują *steganografię* z *kryptografią* oraz technikami zapewniającymi bezpieczeństwo transmisji danych określając dla nich wspólną terminologię. W artykule omówiono też różne podejścia do *znakowania wodnego* w mediach cyfrowych, takich jak *audio* i *wideo*.

Z kolei w pracy F. Petitcolasa, R. Andersona i M. Kuhna *"Information hiding - a survey*" z 1999 roku *(Petitcolas, Anderson i Kuhn, 1999) steganografia* jest opisana jako nauka o ukrywaniu *informacji*, której celem jest zakamuflowanie samego faktu istnienia przekazu. Autorzy definiują *steganografię* jako metody ukrywania danych w różnych *nośnikach*, takich jak obrazy, dźwięki lub teksty, a *steganoanalizę* jako analizę użycia technik *steganograficznych* oraz opis sposobów ekstrakcji ukrytych danych.

Na potrzeby niniejszej rozprawy przyjmuje się, że *steganografia* jest umiejętnością ukrywania *komunikatu* w *nośnikach* cyfrowych, które na pierwszy rzut oka nie są z nimi powiązane, takich jak obrazy, pliki *audio*, *wideo*, dokumenty cyfrowe lub parametry transmisji sieciowej. Kluczowym założeniem *steganografii* jest zapewnienie, że obecność *komunikatu* pozostaje niewidoczna dla osób nieupoważnionych. Oznacza to, że chroniona jest nie tylko sama treść *komunikatu*, ale zamaskowany jest także fakt jego istnienia.

Proces *steganografii* prowadzi do powstania *steganogramu*[20], który jest kombinacją *nośnika* i *komunikatu*.

Jednym z podstawowych celów *steganografii* jest minimalizowanie różnic między *steganogramem* a *nośnikiem* tak, aby *komunikat* był trudny do wykrycia nawet przy użyciu zaawansowanych technik analizy.

Istnieje wiele różnych technik *steganograficznych*, które w zależności m.in. od rodzaju *nośnika,* kodują *komunikat* tworząc *steganogram*. W uproszczeniu każdą technikę *steganografii* cyfrowej można interpretować jako funkcję F, która przypisuje *komunikatowi* m (*informacji*, którą *nadawca* chce zakodować) oraz *nośnikowi* cyfrowemu c (w którym *nadawca* chce ten *komunikat* zakodować) nowy plik cyfrowy s będący *steganogramem*. Funkcja F jest zdefiniowana formułą (1).

$$s := F(m, c) \tag{1}$$

gdzie: m - *komunikat,*
c - *nośnik*
s - utworzony *steganogram.*

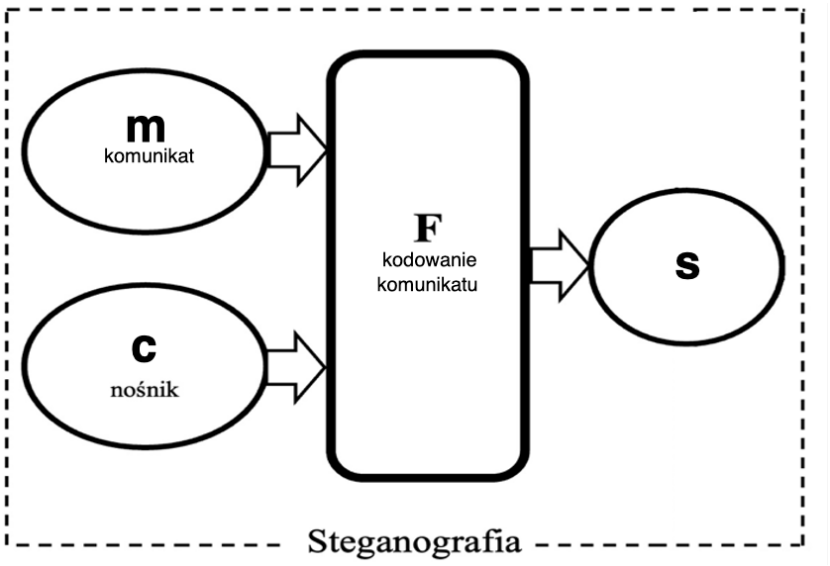
To uproszczenie nie obejmuje bardziej złożonych sytuacji, kiedy *nośnikiem* nie jest typowy plik cyfrowy, ale na przykład są to parametry transmisji sieciowej. W takich przypadkach funkcja F może przyjmować bardziej złożoną postać, zależną np. od uwarunkowań czasowych *nośnika* oraz *komunikatu*. Na przykład techniki ukrywania *informacji* w *wideo* mogą wykorzystywać do kodowania *komunikatu* różnice między kolejnymi *klatkami*[21], co wymaga dynamicznego dostosowywania funkcji F w trakcie kodowania kolejnych *klatek*.

---

[20] **Steganogram** - *nośnik*, w którym ukryto *komunikat*.

[21] **Klatka** - podstawowa jednostka w *wideo*, która przedstawia pojedynczy statyczny obraz, wyświetlany w określonym czasie, tworząc wrażenie ruchu po zestawieniu z kolejnymi *klatkami.*

Ogólny schemat funkcji *steganograficznej* przedstawiono na rysunku 3.



Rysunek 3 - Schemat funkcji *steganograficznej F*
Źródło: *(Pery, Zastosowanie steganologii w cyberbezpieczeństwie, 2024)*

Przykład 1 - działanie obrazowej funkcji steganograficznej LSB[22]

Argumentami przykładowej funkcji *steganograficznej F* są:

- *nośnik*, przedstawiony na rysunku 4, jakim jest wykonane przez autora zdjęcie *El Capitan*[23] w formacie *PNG*[24] o rozdzielczości 640×320 pikseli w przestrzeni kolorów *RGB*[25] i rozmiarze 520 KB,

[22] **LSB (Least Significant Bit)** - najmniej znaczący *bit* o najmniejszej wartości wagowej w binarnej reprezentacji liczby lub technika *steganograficzna* polegająca na kodowaniu kolejnych *bitów komunikatu* w najmniej znaczących *bitach* wybranych parametrów *nośnika (Chan i Cheng, 2004)*.

[23] **El Capitan** - granitowa formacja skalna w Yosemite w USA wznosząca się na 2307 metrów nad poziomem morza. Jego pionowa ściana o wysokości około 900 metrów z charakterystycznym nosem czyni go jednym z najtrudniejszych i jednocześnie najsłynniejszych miejsc wspinaczkowych na świecie. Źródło: https://www.peakbagger.com/peak.aspx?pid=2612.

[24] **PNG (Portable Network Graphics)** - rastrowy format plików graficznych i system *kompresji bezstratnej* danych graficznych *(ISO-IEC, 2004)*.

[25] **RGB (Red, Green, Blue)** - model kolorów używany w elektronice i technologii cyfrowej do reprezentacji i wyświetlania kolorów. Model *RGB* bazuje na mieszaniu trzech podstawowych kolorów światła: czerwonego (Red), zielonego (Green) i niebieskiego (Blue). Kombinacja tych trzech kolorów w różnych proporcjach pozwala na stworzenie szerokiej gamy kolorów *(Hirsch, 2004)*.

Rysunek 4 - Przykład: zdjęcie *El Capitan* jako *nośnik* użyty w *steganografii*
Źródło: *Opracowanie własne*

- *komunikat*, przedstawiony na rysunku 5, jakim jest zdjęcie najlepszego psiego przyjaciela autora niniejszej rozprawy o imieniu *Grom,* zapisane w formacie *PNG* o rozdzielczości 160×160 pikseli w przestrzeni kolorów *RGB* i rozmiarze 57 KB, które na potrzeby zakodowania w nośniku zamieniane jest na tablicę bajtów o wymiarach 160×160×3.

Rysunek 5 - Przykład: zdjęcie *Groma* będące *komunikatem* użytym w *steganografii LSB*
Źródło: *Opracowanie własne*

Wynikiem działania funkcji *steganograficznej F* jest *steganogram*, przedstawiony na rysunku 6, będący obrazem cyfrowym o rozdzielczości odpowiadającej rozdzielczości *nośnika* 640×320 pikseli zapisany w formacie *PNG* w przestrzeni kolorów *RGB* o rozmiarze 557 KB.

Rysunek 6 - Przykład: *steganogram* zakodowany techniką *LSB* będący połączeniem zdjęć *El Capitan* i *Groma*
Źródło: *Opracowanie własne*

W powyższym *steganogramie* został zakodowany *komunikat*, choć na pierwszy rzut oka obraz nie różni się od oryginalnego *nośnika*. Działanie funkcji *F* w tym przypadku zostało przedstawione schematycznie na rysunku 7.

Rysunek 7 - Przykład: działanie funkcji *steganograficznej F* typu *LSB* kodującej na trzech najmniej znaczących *bitach* obraz psa o imieniu *Grom* w obrazie *El Capitan*
Źródło: *Opracowanie własne*

Koniec przykładu 1

Przykład 2 - działanie obrazowej funkcji steganograficznej typu DCT[26]

Drugi przykład pokazuje zastosowanie *steganografii transformatowej DCT*. *Nośnik* pozostaje ten sam, jednakże ze względu na mniejszą *pojemność* informacyjną, użyto w tym przypadku innego *komunikatu*, którym jest zdjęcie głowy *Groma* w rozdzielczości 32×32 pikseli w odcieniach szarości zapisane w formacie *PNG* o rozmiarze 2 KB, przedstawione na rysunku 8. W tym przypadku *komunikat* jest tablicą bajtów o rozmiarze 32×32.

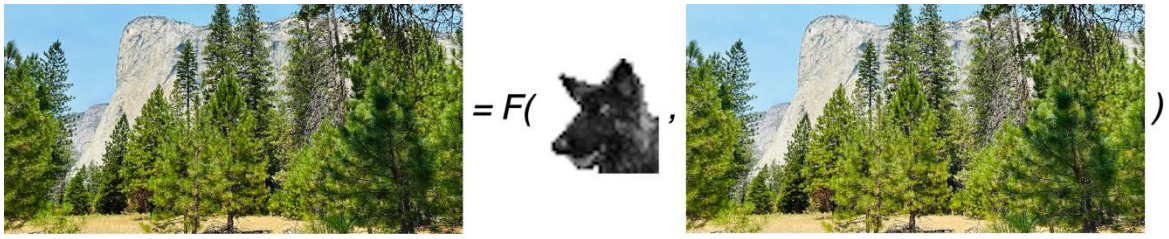
Rysunek 8 - Przykład: zdjęcie *Groma* będące *komunikatem* użytym w *steganografii DCT*
Źródło: *Opracowanie własne*

Wynikiem działania funkcji *F* jest w tym przypadku *steganogram* przedstawiony na rysunku 9, w formacie analogicznym do formatu *nośnika,* o rozmiarze 541 KB.

[26] **DCT (Discrete Cosine Transform)** - rodzaj *transformaty*, która przekształca sygnały w dziedzinie czasu lub przestrzeni na sygnały w dziedzinie częstotliwości przy użyciu funkcji cosinusowych. *DCT* jest szczególnie efektywna w analizie i kompresji danych, zwłaszcza obrazów i dźwięków, dzięki swojej zdolności do koncentracji energii sygnału w kilku współczynnikach. Metody wykorzystujące *DCT* są często stosowane w *steganografii obrazowej (Ahmed, Natarajan i Rao, 1974)*.

Rysunek 9 - Przykład: *steganogram* DCT będący połączeniem zdjęć *El Capitan* i *Groma*
Źródło: *Opracowanie własne*

W przypadku tego *steganogramu* zmiany wynikające z zakodowania *komunikatu* również na pierwszy rzut oka są niezauważalne. Działanie funkcji *F* w tym przypadku, przedstawione na rysunku 10, jest analogiczne, jak w poprzednim przykładzie.

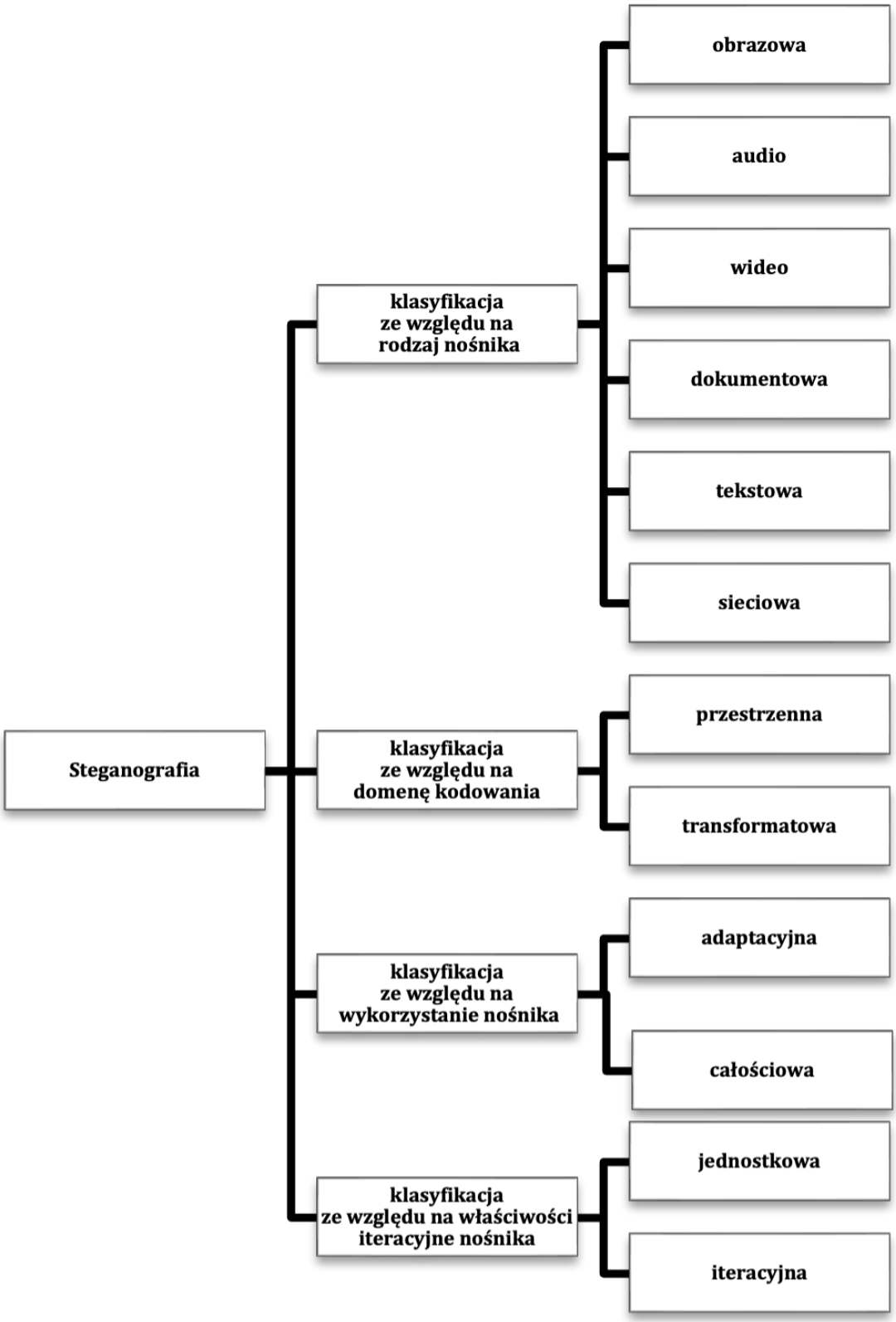


Rysunek 10 - Przykład: działanie funkcji *steganograficznej F* typu *DCT* kodującej obraz psa o imieniu *Grom* w obrazie *El Capitan*
Źródło: *Opracowanie własne*

Koniec przykładu 2

2.2.1 Klasyfikacje metod steganografii

W literaturze istnieje wiele podejść do zagadnienia klasyfikacji metod *steganograficznych*, np. (Petitcolas, Anderson i Kuhn, 1999; Fridrich, Steganography in Digital Media: Principles, Algorithms, and Applications, 2009; Johnson, Duric i Jajodia, Information Hiding: Steganography and Watermarking - Attacks and Countermeasures, 2001). Najczęściej spotykane klasyfikacje przedstawiono na rysunku 11.



Rysunek 11 - Klasyfikacje *steganografii*
Źródło: *Opracowanie własne*

2.2.1.1 Klasyfikacja steganografii ze względu na rodzaj nośnika

Jest to najczęściej występująca klasyfikacja metod *steganograficznych* i jednocześnie najbardziej naturalna, gdyż uzależniona jest od typu *nośnika*, który jest użyty do utworzenia *steganogramu*. W ramach tej klasyfikacji wyróżnia się następujące typy *steganografii*:

a) *steganografia obrazowa*[27] - polega na ukrywaniu *komunikatu* w cyfrowych obrazach np. przez manipulowanie wartościami kolorów lub jasności pikseli, zmianę niektórych właściwości skompresowanych obrazów lub wykorzystanie *metadanych* plików cyfrowych obrazów,

b) *steganografia audio*[28] - polega na ukrywaniu *komunikatu* w plikach dźwiękowych, np. przez subtelne zmiany w amplitudzie lub fazie sygnału dźwiękowego,

c) *steganografia wideo*[29] - zakłada umieszczanie *komunikatu* w *wideo*[30] np. poprzez zmianę poszczególnych *klatek* lub wykorzystanie właściwości *kodeków* i ukrycie *komunikatu* w *informacjach* o przejściach pomiędzy *klatkami*.

d) *steganografia dokumentowa*[31] - polega na umieszczaniu *komunikatu* w plikach różnych dokumentów np. przez bezpośrednią zmianę w treści dokumentu, w *metadanych* pliku cyfrowego bądź w dokumentach lub multimediach zagnieżdżonych w innym dokumencie,

e) *steganografia lingwistyczna*[32] - nazywana również *steganografią tekstową*[33], polega na ukrywaniu *komunikatu* w dokumentach tekstowych, np. poprzez manipulacje na

---

[27] **Steganografia obrazowa** - technika kodowania *komunikatu* w obrazach cyfrowych, która kolejne *bity komunikatu* osadza w parametrach obrazu, np. wartościach kolorów pikseli.

[28] **Steganografia audio** - technika kodowania *komunikatu* w plikach *audio*, która kolejne *bity komunikatu* osadza w parametrach dźwiękowych pliku, np. częstotliwościach dźwięków.

[29] **Steganografia wideo** - technika kodowania *komunikatu* w *wideo*, która kolejne *bity komunikatu* osadza w *wideo*, np. parametrach kolejnych *klatek*.

[30] **Wideo** - zapis ruchomych obrazów i dźwięku w formacie cyfrowym, skompresowany przy użyciu *kodeka*, np. *H.264* i zapisany w kontenerze, np. MP4, zarządzającym synchronizacją między obrazem, dźwiękiem i napisami *(Hanzo, Cherriman i Streit, 2007)*.

[31] **Steganografia dokumentowa** - technika kodowania *komunikatu* w dokumentach lub plikach cyfrowych różnych formatów, która kolejne *bity komunikatu* osadza w parametrach tych plików, np. w ich *metadanych*.

[32] **Steganografia lingwistyczna** - technika kodowania *komunikatu* w tekstach pisanych, która kolejne *bity komunikatu* osadza w modyfikacjach językowych lub zmianie formy bądź treści tekstu.

[33] **Steganografia tekstowa** - inaczej: *steganografia lingwistyczna*.

poziomie zamiany znaków, formatowania, struktur językowych lub wykorzystania znaków niewidocznych,

f) *steganografia sieciowa*[34] - polega na ukrywaniu *komunikatu* w ruchu sieciowym, czyli w danych przesyłanych przez sieć komputerową np. poprzez wykorzystanie różnorodnych aspektów protokołów sieciowych, *nagłówków* pakietów oraz innych właściwości sieci *(Mazurczyk, Wendzel, Zander, Houmansadr i Szczypiorski, 2016)*.

2.2.1.2 Klasyfikacja steganografii ze względu na domenę kodowania

Metody *steganografii* można także sklasyfikować ze względu na to, czy bazują one na bezpośredniej podmianie wartości danych *nośnika,* czy operują w przestrzeni *transformat*[35] częstotliwościowych obliczanych dla tego *nośnika*. Ze względu na to kryterium wyróżnia się następujące dwa typy metod:

a) *steganografia przestrzenna*[36] - operująca w *domenie przestrzennej*[37], polega na ukrywaniu *komunikatu* bezpośrednio w danych *nośnika*, np. w obrazie lub *klatkach* poprzez modyfikację wartości w przestrzeni kolorów lub jasności ich pikseli,

b) *steganografia transformatowa*[38] - operująca w *domenie częstotliwościowej*[39], polega na manipulacji wartościami parametrów *transformat*, które tworzone są dla danego *nośnika* zmieniając np. *domenę przestrzenną* w *domenę częstotliwościową*.

---

[34] **Steganografia sieciowa** - technika kodowania *komunikatu* w ruchu sieciowym, która kolejne *bity komunikatu* osadza w parametrach protokołów sieciowych, *nagłówkach* pakietów lub innych elementach transmisji danych *(Mazurczyk, Wendzel, Zander, Houmansadr i Szczypiorski, 2016)*.

[35] **Transformata** - operacja matematyczna przekształcająca jedną funkcję w inną funkcję upraszczając problem i ułatwiając jego analizę i obliczenia *(Polyanin i Manzhirov, 2008)*.

[36] **Steganografia przestrzenna -** typ metod *steganograficznych* kodujących *komunikaty* poprzez bezpośrednią modyfikację wartości poszczególnych pikseli obrazu cyfrowego.

[37] **Domena przestrzenna** - sposób reprezentacji *nośnika*, w którym operacje analizy lub modyfikacji są wykonywane poprzez bezpośrednie manipulowanie jego wartościami składowymi, np. pikselami.

[38] **Steganografia transformatowa** - typ metod *steganograficznych* kodujących *komunikaty* w parametrach *transformat* obliczonych dla *nośnika* po przekształceniu go do *domeny częstotliwościowej*.

[39] **Domena częstotliwościowa** - sposób reprezentacji *nośnika*, dla którego najpierw wyznaczane są *transformaty* częstotliwościowe, a następnie są one poddawane analizie lub modyfikacji.

2.2.1.3 Klasyfikacja steganografii ze względu na użycie lokalnych własności nośnika

Niektóre algorytmy *steganograficzne* abstrahują od konkretnych lokalnych cech wykorzystywanego *nośnika*, a niektóre wykorzystują właśnie te lokalne właściwości w celu kodowania *komunikatu* jedynie w niektórych wybranych obszarach *nośnika*. Ze względu na to kryterium można wyróżnić dwa typy metod:

a) *steganografia adaptacyjna*[40] - polega na dostosowywaniu się do lokalnych właściwości *nośnika* w celu zwiększenia *pojemności* i *niewykrywalności* poprzez wynajdywanie lokalnych nieciągłości lub wyróżnionych obszarów *nośnika*, np. w przypadku obrazów algorytm szuka obszarów z dużą liczbą szczegółów, tekstur, krawędzi lub obszarów ciemniejszych i w tych miejscach wprowadzane są zmiany, które są mniej zauważalne dla ludzkiego oka,
b) *steganografia całościowa*[41] - polega na traktowaniu wszystkich obszarów *nośnika* jednakowo.

2.2.1.4 Klasyfikacja steganografii ze względu na właściwości iteracyjne nośnika

Większość znanych prostych *nośników* składa się z jednego, całościowego elementu, takiego jak pojedynczy tekst lub cyfrowy obraz. Są też nośniki, które charakteryzują się budową iteracyjną, złożoną z wielu powtarzalnych w czasie elementów. Ze względu na tę własność można wyróżnić dwa typy metod:

a) *steganografia jednostkowa*[42] - polega na traktowaniu *nośnika* jako jednego, całościowego obiektu, w którym jest kodowany cały *komunikat*,

---

[40] **Steganografia adaptacyjna** - typ metod *steganograficznych* kodujących *komunikaty* z dostosowaniem się do lokalnych charakterystyk *nośnika*, takich jak np. struktura obrazu, w tym tekstury, krawędzie, itp.

[41] **Steganografia całościowa** - typ metod *steganograficznych* kodujących *komunikaty* bez uwzględnienia lokalnych charakterystyk *nośnika*.

[42] **Steganografia jednostkowa** - typ metod *steganograficznych* kodujących od razu całe *komunikaty* w *nośniku* przy wykorzystaniu faktu, iż *nośnik* ma charakter jednostkowy, całościowy, niepodzielny na części.

b) *steganografia iteracyjna*[43] - polega na kodowaniu informacji stopniowo, po podzieleniu *nośnika* na jednorodne segmenty, z których każdy zawiera częściowo zakodowane dane *(Pery i Waszkowski, A Model and Quantitative Framework for Evaluating Iterative Steganography, 2024)*. Pełne odczytanie ukrytego *komunikatu* jest możliwe dopiero po przeprowadzeniu wielu iteracji algorytmu dekodującego. Każda kolejna iteracja powoduje dodanie jakiejś porcji *informacji* i dopiero po osiągnięciu pewnego progu osiąga ona ilość wystarczającą do pełnego zdekodowania ukrytego *komunikatu*. Przykładami *steganografii iteracyjnej* może być *steganografia wideo* lub wybrane techniki *steganografii sieciowej*.

Należy podkreślić, że wymienione powyżej klasyfikacje nie wykluczają się wzajemnie i istnieje możliwość przynależności jednego *steganogramu* do wielu kategorii jednocześnie. Na przykład *steganografia przestrzenna* może być jednocześnie *steganografią adaptacyjną* i *steganografią iteracyjną*.

2.2.2 Charakterystyka podstawowych technik steganograficznych

Dobór odpowiedniej techniki *steganograficznej* jest kluczowy dla skutecznego ukrycia *komunikatu* i zależy od wielu czynników. Główne aspekty, które należy wziąć pod uwagę, to rodzaj *nośnika* danych oraz pożądane efekty końcowe, takie jak wartości miar *pojemności*, *niewykrywalności* i *odporności*. Na przykład dla ukrycia dużych ilości danych szczególnie efektywne są techniki *steganografii wideo*, gdzie komunikaty mogą być ukrywane w poszczególnych *klatkach* lub w danych definiujących przejścia pomiędzy kolejnymi *klatkami*.

Każda technika *steganograficzna* ma swoje unikalne zalety i ograniczenia. Na przykład *steganografia obrazowa* zapewnia równowagę między *pojemnością* a *niewykrywalnością*, podczas gdy *steganografia sieciowa* może charakteryzować się większą *niewykrywalnością*, ale oferuje mniejszą *pojemność*.

---

[43] **Steganografia iteracyjna** - typ metod *steganograficznych* kodujących *komunikaty* iteracyjnie, wykorzystując właściwości iteracyjne *nośnika (Pery i Waszkowski, A Model and Quantitative Framework for Evaluating Iterative Steganography, 2024)*.

Poniżej przedstawione są najczęściej spotykane podstawowe techniki *steganograficzne*. W praktyce bardzo często spotyka się różne modyfikacje tych podstawowych technik oraz stosowanie technik mieszanych.

2.2.2.1 Technika steganograficzna LSB (Least Significant Bit)

Technika *LSB* jest jedną z najprostszych i najczęściej stosowanych metod *steganografii* polegającą na zastępowaniu kolejnymi *bitami komunikatu* najmniej znaczących *bitów* w *nośniku*, np. w pikselach obrazu lub próbkach dźwięku. Na rysunku 12 przedstawiono przykład najmniej znaczących *bitów* liczb 8-bitowych.

liczby dziesiętne	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
215	**1**	**1**	**0**	**1**	**0**	**1**	**1**	**1**
129	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
79	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**
2	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**
25	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**1**
254	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**
								LSB

Rysunek 12 - Przykład: liczby 8-*bitowe* z wyróżnionymi najmniej znaczącymi *bitami LSB*
Źródło: *Opracowanie własne*

W artykule C. Chana i L. Chenga *"Hiding data in images by simple LSB substitution"* z 2004 roku *(Chan i Cheng, 2004)* przedstawiono technikę *steganografii obrazowej* bazującą na *LSB*. Artykuł szczegółowo opisuje proces ukrywania i wydobywania *informacji*, analizuje efektywność i niezawodność metody oraz omawia jej zastosowania i ograniczenia.

W przypadku algorytmów *LSB* działających w *domenie przestrzennej komunikaty* kodowane są w *bitach* reprezentujących najmniej istotne parametry definiujące np. kolor lub jasność pikseli bądź częstotliwość dźwięku i mających najmniejszy wpływ na percepcję zmian w *steganogramie*. Jeżeli obraz zapisany jest w przestrzeni kolorów *RGB*, a każdy piksel obrazu jest reprezentowany przez określoną liczbę *bitów*, np. ośmiobitowy *kanał* dla koloru czerwonego, zielonego i niebieskiego, to wykorzystanie najmniej znaczącego *bitu* jednego wybranego koloru umożliwia zapisanie jednego *bitu komunikatu* na 24 *bity nośnika*.

Przykładowy proces kodowania tekstowego *komunikatu* polegałby w tym przypadku na konwersji *komunikatu* na formę binarną, a następnie wprowadzeniu kolejnych *bitów*

komunikatu zamiast najmniej znaczących *bitów* wybranego koloru kolejnych pikseli *nośnika*. Metoda *LSB* może wykorzystywać również rozpraszanie kodowanego *komunikatu* pomiędzy różnymi *kanałami* kolorów, co zwiększa *pojemność*, ale zmniejsza jednocześnie *niewykrywalność* wynikowego *steganogramu*.

Techniki *LSB* można użyć również w *domenie częstotliwościowej*. Wtedy modyfikacji ulegają np. najmniej znaczące *bity* parametrów wybranych *transformat* obliczonych dla *nośnika*.

Algorytmy implementujące metody *LSB* mogą być zarówno technikami *steganografii całościowej*, jak też *steganografii adaptacyjnej*. W przypadku *steganografii adaptacyjnej* kodowanie kolejnych *bitów komunikatu* może odbywać się w wybranych lokalnych obszarach *nośnika* i to zarówno w *domenie przestrzennej*, jak też w *domenie częstotliwościowej*.

Kod 1 zawiera przykład algorytmu *LSB* działającego w *domenie przestrzennej* kodującego *komunikat* na najmniej znaczących *bitach* wszystkich trzech kolorów z przestrzeni *RBG*.

```
def LSB_code(cover_image, message):
    """
    Koduje komunikat w nośniku przy użyciu metody LSB (Least Significant Bit).

    Argumenty:
    cover_image (np.array): nośnik
    message (np.array): komunikat
    Wyniki:
    np.array: steganogram
    """

    # Tworzenie kopii obrazu, aby nie modyfikować oryginału
    steganogram = np.copy(cover_image)

    # Obliczenie pojemności steganogramu
    capacity = steganogram.size

    # Konwersja komunikatu do bitów
    bits = np.unpackbits(np.array(message, dtype=np.uint8))

    if len(bits) <= capacity:
        # Dodanie wypełnienia zerami
        padded = np.pad(bits, (0, capacity - len(bits)), 'constant')

        # Maska najmniej znaczących bitów pikseli obrazu
        mask = np.ones_like(steganogram, dtype=np.uint8) * 0b11111110

        # Zastąpienie najmniej znaczącego bitu bitami komunikatu
        steganogram = (steganogram & mask) | padded.reshape(steganogram.shape)

    return steganogram
```

```
def LSB_decode(steganogram_image, size):
    """
    Dekoduje komunikat ze steganogramu zakodowanego przy użyciu metody LSB.

    Argumenty:
    steganogram_image (np.array): steganogram
    size (int): rozmiar komunikatu do dekodowania
    Wyniki:
    np.array: zdekodowany komunikat
    """

    # Obliczenie liczby bitów do zdekodowania
    num_bits = size * 8

    # Wyodrębnienie najmniej znaczących bitów z obrazu
    bits = steganogram_image & 1

    # Spłaszczenie tablicy bitów i wybranie tylko potrzebnych bitów
    message_bits = bits.flatten()[:num_bits]

    # Konwersja bitów z powrotem do bajtów
    message_bytes = np.packbits(message_bits)

    return message_bytes
```

Kod 1 - Algorytmy kodowania i dekodowania w *technice LSB*
Źródło: *Opracowanie własne*

Ciekawym rozwinięciem prostej metody *LSB* jest technika zaproponowana przez W. Luo'ego, F. Huanga i J. Huanga w artykule *"Edge adaptive image steganography based on LSB matching revisited"* z 2010 roku *(Luo, Huang i Huang, 2010)*. Proponowana metoda polega na adaptacyjnym dostosowaniu metody *LSB*. Autorzy proponują schemat adaptacyjny, który selektywnie wybiera obszary do ukrywania *informacji*, którymi są znalezione w *nośniku* krawędzie i w zależności od wielkości *komunikatu* oraz różnicy między dwoma kolejnymi pikselami koduje kolejne *bity komunikatu*. Autorzy wykorzystują w proponowanej metodzie fakt, iż ludzkie oko jest mniej wrażliwe na zmiany w zakresie krawędzi wewnątrz obrazów, co pozwala na zwiększenie *pojemności* przy jednoczesnym zachowaniu dużej *niewykrywalności steganogramu*. Wyniki eksperymentalne uzyskane przez autorów pokazują m.in. to, że metoda ta przewyższa inne techniki *steganograficzne* pod względem *niewykrywalności* w przypadku wykorzystania do *steganoanalizy* metody z użyciem funkcji charakterystycznej *histogramu HCF*[44].

[44] **HCF (Histogram Characteristic Function)** - technika *steganoanalizy* wykorzystująca analizę *histogramu* do identyfikacji potencjalnych anomalii w *steganogramie* mogących wskazywać na zastosowanie *steganografii (Harmsen i Pearlman, 2003)*.

2.2.2.2 Technika steganograficzna Patchwork

Technika *Patchwork*[45] polega na ukryciu *komunikatu* w obrazie poprzez wprowadzenie niewielkich, statystycznie rozłożonych modyfikacji wartości pikseli, które są niezauważalne dla ludzkiego oka. Metoda ta wybiera losowo pary pikseli w różnych miejscach obrazu. W jednej parze nieznacznie zwiększa się wartość jasności pikseli, a w drugiej odpowiednio ją zmniejsza. Dzięki temu średnia jasność obrazu pozostaje praktycznie niezmieniona, ale wprowadzone są subtelne różnice statystyczne. Osoba znająca algorytm i klucz losowania pikseli może odczytać ukrytą *informację* poprzez analizę statystyczną tych niewielkich odchyleń. Do określenia, które piksele są zmieniane, wykorzystywany jest generator liczb pseudolosowych, inicjowany tajnym kluczem znanym *nadawcy* i *odbiorcy komunikatu*.

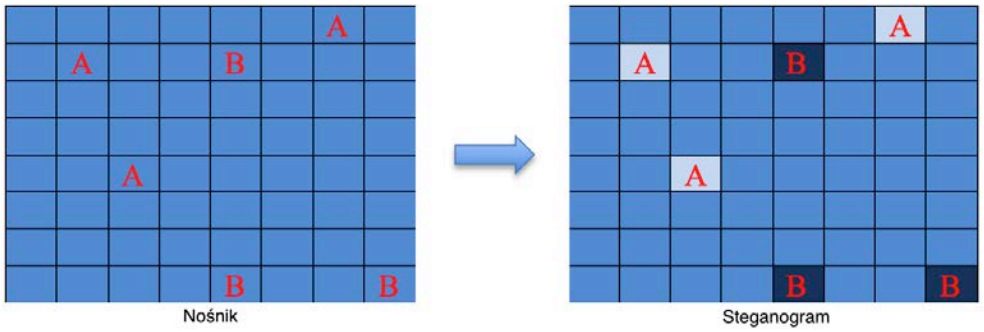
Technika *Patchwork* została szczegółowo opisana w artykule *"Techniques for data hiding"* autorstwa W. Bendera, D. Gruhla, N. Morimoto i A. Lu z 1996 roku *(Bender, Gruhl, Morimoto i Lu, 1996)*. Praca ta była i jest nadal często cytowana w literaturze dotyczącej *steganografii*, *steganoanalizy* i metod *znakowania wodnego*. Zawiera opisy zarówno tradycyjnych, jak i nowatorskich na tamte czasy metod *steganograficznych* w mediach cyfrowych. Autorzy analizują te techniki w kontekście trzech głównych zastosowań: ochrony praw autorskich, zabezpieczania przed manipulacją oraz osadzania ukrytych dodatkowych *informacji*. W artykule omówiono również różne metody *steganografii*, w tym *steganografii przestrzennej* i *steganografii transformatowej* oraz ich zastosowania w praktyce.

Kluczowym aspektem metody *Patchwork* jest fakt, że w niemodyfikowanym *nośniku* oczekiwana wartość sumy różnic między odpowiadającymi sobie pikselami z zestawów A i B wynosi zero. Jednakże po zastosowaniu techniki *Patchwork* wartość ta zostaje przesunięta o określoną wartość. Wykrycie tego przesunięcia pozwala stwierdzić, czy obraz został zmodyfikowany tą metodą. Istotną cechą *Patchwork* jest możliwość przypisania poziomu pewności do procesu kodowania, który reprezentuje wiarygodność

[45] **Patchwork** - technika *steganograficzna* polegająca na wprowadzaniu zmian w jasności wybranych w sposób pseudolosowy par pikseli obrazu tak, aby zakodować kolejne *bity komunikatu (Bender, Gruhl, Morimoto i Lu, 1996)*.

odpowiedzi dekodowania. Zwiększanie liczby modyfikowanych pikseli zwiększa *odporność steganogramu*, ale jednocześnie zmniejsza jego *niewykrywalność*.

Schemat działania algorytmu *Patchwork* dla obrazu cyfrowego reprezentowanego przez dwuwymiarową tablicę pikseli pokazano na rysunku 13.



Rysunek 13 - Przykład: działanie techniki *Patchwork*
Piksele A są rozjaśniane, a piksele B są przyciemniane.
Źródło: *Opracowanie własne*

Kod 2 realizuje prosty przykładowy algorytm kodowania *steganograficznego* techniką *Patchwork*.

```
def Patchwork_code(cover_image, message):
    """
    Koduje komunikat w nośniku przy użyciu techniki Patchwork.
    Argumenty:
    cover_image (np.array): nośnik
    message (np.array): komunikat
    Wyniki:
    np.array: steganogram.
    """

    # Tworzenie kopii obrazu, aby nie modyfikować oryginału
    steganogram = np.copy(cover_image)

    # Sprawdzenie czy steganogram ma wystarczającą pojemność
    if steganogram.size > len(message) * 2:

        # Konwersja komunikatu do bitów
        message_bits = np.unpackbits(np.array(message, dtype=np.uint8))
        bit_index = 0
        h, w, d = steganogram.shape

        # Iteracja przez piksele obrazu
        for i in range(0, h - 1, 2):
            for j in range(0, w - 1, 2):
                for k in range(d):
                    if bit_index >= len(message_bits):
                        return steganogram

                    pixel1 = np.int32(steganogram[i, j, k])
                    pixel2 = np.int32(steganogram[i + 1, j + 1, k])
                    diff = pixel1 - pixel2
```

```
                    bit = message_bits[bit_index]

                    # Modyfikacja różnicy, aby zakodować bit komunikatu
                    if (bit == 0 and diff >= 0) or (bit == 1 and diff < 0):
                        diff += -5 if bit == 0 else 5

                    avg = (pixel1 + pixel2) // 2
                    steganogram[i,j,k] = np.clip(avg + (diff + 1) // 2, 0, 255)
                    steganogram[i+1, j+1, k] = np.clip(avg - diff // 2, 0, 255)
                    bit_index += 1

    return steganogram

def Patchwork_decode(steganogram_image, size):
    """
    Dekoduje komunikat ze steganogramu zakodowanego przy użyciu metody
Patchwork.
    Argumenty:
    steganogram_image (np.array): steganogram
    size (int): rozmiar komunikatu do zdekodowania
    Wyniki:
    np.array: zdekodowany komunikat
    """

    # Obliczenie liczby bitów do zdekodowania
    bits_needed = size * 8
    message_bits = []

    # Iteracja przez piksele obrazu
    for i in range(0, steganogram_image.shape[0] - 1, 2):
        for j in range(0, steganogram_image.shape[1] - 1, 2):
            if len(message_bits) >= bits_needed:
                break
            for k in range(steganogram_image.shape[2]):
                pixel1 = np.int32(steganogram_image[i, j, k])
                pixel2 = np.int32(steganogram_image[i + 1, j + 1, k])
                diff = pixel1 - pixel2
                # Odczytanie zakodowanego bitu z różnicy pikseli
                message_bits.append('0' if diff < 0 else '1')
                if len(message_bits) >= bits_needed:
                    break

    # Konwersja bitów z powrotem do bajtów
    message_bytes = bytearray(
        int("".join(message_bits[i:i+8]), 2) for i in range(0, bits_needed, 8))

    return np.array(message_bytes, dtype=np.uint8)
```

Kod 2 - Algorytmy kodowania i dekodowania w *technice Patchwork*
Źródło: *Opracowanie własne*

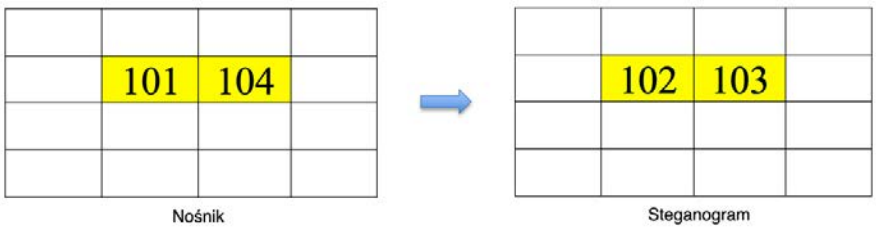
Metoda *Patchwork* charakteryzuje się stosunkowo niską *pojemnością* informacyjną. Jej podstawowym celem jest odpowiedź na pytanie, czy obraz został zakodowany przy użyciu konkretnej łaty, czy też nie. Z drugiej strony technika ta wykazuje dużą *odporność* na pewne rodzaje przekształceń, np. redukcja rozmiaru obrazu prowadzi jedynie do logarytmicznego zmniejszenia dokładności detekcji. *Patchwork* jest jednak podatny na

transformacje afiniczne, takie jak translacja, rotacja lub skalowanie. Ze względu na swoją stosunkowo dobrą *odporność* i niską *pojemność*, metoda *Patchwork* jest szczególnie przydatna w dziedzinie cyfrowego *znakowania wodnego*[46].

2.2.2.3 Technika steganograficzna PVD (Pixel Value Differencing)

Technika *PVD*[47] została zaprezentowana przez D. Wu'a i W. Tsai'a w artykule *"A steganographic method for digital images using pixel-value differencing"* z 2003 roku *(Wu i Tsai, 2003)*. Polega ona na ukrywaniu *komunikatu* w obrazach poprzez modyfikację wartości jasności sąsiadujących pikseli tak, aby zmiany te były niewidoczne dla ludzkiego oka i jednocześnie umożliwiały zakodowanie kolejnych *bitów komunikatu*. Obraz jest dzielony na bloki, zazwyczaj składające się z dwóch sąsiednich pikseli. Dla każdego bloku oblicza się różnicę wartości jasności między dwoma sąsiednimi pikselami, a różnica ta służy do określenia zakresu, w którym można modyfikować wartości pikseli bez widocznych zmian w obrazie. Ukrywany *komunikat* jest zakodowany w różnicy między wartościami pikseli *steganogramu*. Wartości jasności kolejnych pikseli są odpowiednio modyfikowane tak, aby różnica między wartościami sąsiednich pikseli po modyfikacji była nadal w akceptowalnym zakresie co sprawia, że ukryty *komunikat* pozostaje niewidoczny.

Technika *PVD* charakteryzuje się stosunkowo dużą *niewykrywalnością*, ponieważ dzięki modyfikowaniu różnic między pikselami zmiany w *steganogramie* są mniej zauważalne dla ludzkiego oka.



Rysunek 14 - Przykład: działanie techniki *PVD* na dwóch sąsiadujących pikselach
Źródło: *Opracowanie własne*

[46] **Znakowanie wodne** - technika zabezpieczania danych polegająca na osadzaniu ukrytej *informacji (znaku wodnego)* w pliku multimedialnym, np. obrazie, *wideo* lub dźwięku w sposób zauważalny lub niezauważalny, służąca ochronie praw autorskich, zapewnieniu autentyczności lub śledzeniu nieautoryzowanego użycia *(Shih, 2017)*.

[47] **PVD (Pixel Value Differencing)** - adaptacyjna technika *steganograficzna*, która ukrywa *komunikat* w *steganogramie* poprzez modyfikację różnic jasności między sąsiadującymi pikselami *(Wu i Tsai, 2003)*.

Kod 3 zawiera przykład prostego algorytmu realizującego technikę *steganografii obrazowej PVD*.

```
def PVD_code(cover_image, message):
    """
    Koduje komunikat przy użyciu metody PVD (Pixel Value Differencing).

    Argumenty:
    cover_image (np.array): nośnik
    message (np.array): komunikat

    Wyniki:
    np.array: steganogram
    """

    # Konwersja komunikatu do bitów
    message_bits = np.unpackbits(message)
    bit_index = 0

    # Tworzenie kopii obrazu, aby nie modyfikować oryginału
    steganogram = np.copy(cover_image)
    h, w, d = steganogram.shape

    # Iteracja przez piksele obrazu
    for i in range(h - 1):
        for j in range(w):
            for k in range(d):
                if bit_index >= len(message_bits):
                    return steganogram

                pixel1 = int(steganogram[i, j, k])
                pixel2 = int(steganogram[i + 1, j, k])
                diff = pixel1 - pixel2

                # Modyfikacja różnicy, aby zakodować bit wiadomości
                if message_bits[bit_index] == 0:
                    if diff % 2 != 0:
                        if pixel1 > pixel2:
                            pixel1 -= 1
                        else:
                            pixel2 -= 1
                else:
                    if diff % 2 == 0:
                        if pixel1 > pixel2:
                            pixel1 -= 1
                        else:
                            pixel2 -= 1

                # Przypisanie zmodyfikowanych wartości pikseli
                steganogram[i, j, k] = np.clip(pixel1, 0, 255)
                steganogram[i + 1, j, k] = np.clip(pixel2, 0, 255)
                bit_index += 1

    return steganogram


def PVD_decode(steganogram_image, size):
```

```
    """
    Dekoduje komunikat ze steganogramu zakodowanego przy użyciu techniki PVD.

    Argumenty:
    steganogram_image (np.array): steganogram
    size (int): rozmiar komunikatu do zdekodowania

    Wyniki:
    np.array: zdekodowany komunikat
    """

    # Obliczenie liczby bitów do zdekodowania
    bits_needed = size * 8
    message_bits = []
    h, w, d = steganogram_image.shape

    # Iteracja przez piksele obrazu
    for i in range(h - 1):
        for j in range(w):
            for k in range(d):
                if len(message_bits) >= bits_needed:
                    break

                pixel1 = int(steganogram_image[i, j, k])
                pixel2 = int(steganogram_image[i + 1, j, k])
                diff = pixel1 - pixel2

                # Odczytanie zakodowanego bitu z różnicy pikseli
                message_bits.append(0 if diff % 2 == 0 else 1)

    # Konwersja bitów z powrotem do bajtów
    return np.packbits(message_bits[:bits_needed])[:size]
```

Kod 3 - Algorytmy kodowania i dekodowania w *technice PVD*
Źródło: *Opracowanie własne*

2.2.2.4 Techniki steganograficzne częstotliwościowe

Steganografia transformatowa polega na manipulacji wartościami częstotliwości *transformat* obliczonych dla *nośnika*. Proces ten rozpoczyna się od transformacji *nośnika* z *domeny przestrzennej* do *domeny częstotliwościowej*, a następnie zakodowaniu *komunikatu* w zmianach parametrów utworzonych *transformat*.

Znanych jest wiele technik *steganografii transformatowej* wykorzystujących różne *transformaty*, w tym *transformatę* Fouriera *FT*[48], dyskretną *transformatę* cosinusową *DCT* oraz dyskretną *transformatę* falkową *DWT*[49].

Najczęściej wykorzystywana technika *steganografii transformatowej* bazuje na transformacie *DCT (Chang, Chen i Chung, 2002)*, która jest również szeroko stosowana w kompresji obrazów, np. w formacie *JPEG*[50].

Działanie *transformaty DCT* rozpoczyna się od podziału obrazu na bloki pikseli, zazwyczaj 8 x 8 pikseli. Następnie następuje przekształcenie bloków pikseli z *domeny przestrzennej* do *domeny częstotliwościowej* poprzez wyliczenie macierzy 64 współczynników *DCT* reprezentujących występowanie różnych częstotliwości w danym bloku pikseli. Kodowanie kolejnych *bitów komunikatu* w tej metodzie polega na modyfikacji wybranych współczynników *DCT*. Zazwyczaj wybierane są współczynniki odpowiadające średnim częstotliwościom, ponieważ modyfikacje tych współczynników są mniej widoczne dla ludzkiego oka i mniej podatne na degradację podczas ewentualnej kompresji. W celu zdekodowania *komunikatu odbiorca* może użyć odwrotnej *transformaty IDCT*[51], która przekształca współczynniki *DCT* z powrotem do *domeny przestrzennej*, a bloki pikseli są łączone w celu uzyskania końcowego obrazu z zakodowanym przez *nadawcę komunikatem*.

---

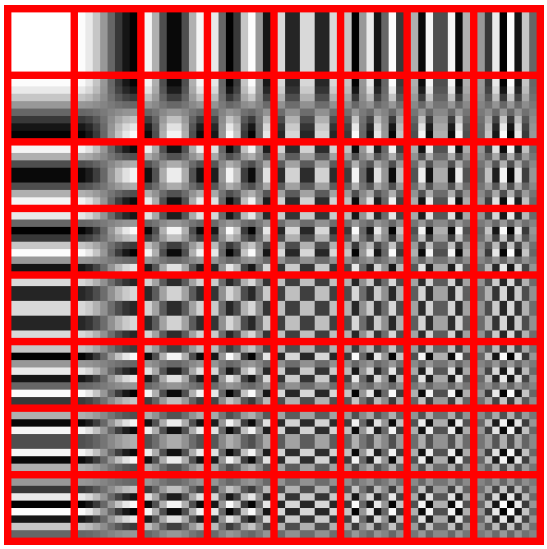
[48] **FT (Fourier Transform)** - Transformata Fouriera, matematyczne narzędzie używane do analizy funkcji lub sygnałów w dziedzinie częstotliwości. *FT* przekształca funkcję czasową lub przestrzenną na reprezentację w dziedzinie częstotliwości, umożliwiając analizę jej składowych harmonicznych. Jest szeroko stosowana w wielu dziedzinach, takich jak inżynieria, fizyka, przetwarzanie sygnałów, analiza obrazu *(Bracewell, 1986; Pinsky, 2009)*.

[49] **DWT (Discrete Wavelet Transform)** - metoda przekształcania sygnału z dziedziny czasu na reprezentację w dziedzinie częstotliwości i czasu przy użyciu falek. *DWT* jest używana w analizie sygnałów, kompresji danych, przetwarzaniu obrazów i innych dziedzinach inżynierii i nauki *(Daubechies, 1992; Akansu i Haddad, 2000)*.

[50] **JPEG (Joint Photographic Experts Group)** - standard *kompresji stratnej* dla obrazów cyfrowych, który zmniejsza rozmiar plików poprzez odrzucenie niektórych danych minimalizując utratę jakości. Proces kompresji obejmuje konwersję kolorów, podział obrazu na bloki 8x8 pikseli, zastosowanie dla nich dyskretnej *transformaty* cosinusowej *DCT* wraz z *kwantyzacją* i *kodowaniem entropijnym* (ITU-T, JPEG (T.81), 1992).

[51] **IDCT (Inverse Discrete Cosine Transform)** - odwrotna *transformata* do *DCT*, która przekształca dane z *domeny częstotliwościowej* do *domeny przestrzennej*. *IDCT* jest stosowana w celu rekonstrukcji oryginalnych wartości pikseli po ich przetworzeniu przez *DCT (Ahmed, Natarajan i Rao, 1974).*

Na rysunku 15 przedstawiono schemat układu 64 współczynników *DCT* dla bloków pikseli 8x8.



Rysunek 15 - Układ współczynników transformaty DCT
Źródło: Public Domain, https://commons.wikimedia.org/w/index.php?curid=1599304

Kod 4 przedstawia przykład prostej implementacji algorytmu *steganografii transformatowej DCT*.

```
# Macierz kwantyzacji dla DCT
matrix = np.asarray([
    [16, 11, 10, 16, 24, 40, 51, 61],
    [12, 12, 14, 19, 26, 58, 60, 55],
    [14, 13, 16, 24, 40, 57, 69, 56],
    [14, 17, 22, 29, 51, 87, 80, 62],
    [18, 22, 37, 56, 68, 109, 103, 77],
    [24, 36, 55, 64, 81, 104, 113, 92],
    [49, 64, 78, 87, 103, 121, 120, 101],
    [72, 92, 95, 98, 112, 100, 103, 99]
], dtype=np.float32)

def scan(matrix):
    """
    Skanowanie macierzy w kolejności zigzag.

    Argumenty:
    matrix (np.array): macierz do zeskanowania

    Wyniki:
    np.array: tablica zeskanowanych elementów
    """

    vmax, hmax = matrix.shape
    output = np.zeros(vmax * hmax)
    v, h, i = 0, 0, 0

    # Skanowanie w kolejności zigzag
```

```
    while v < vmax and h < hmax:
        output[i] = matrix[v, h]
        i += 1
        if (v + h) % 2 == 0:
            if h < hmax - 1:
                if v == 0:
                    h += 1
                else:
                    v -= 1
                    h += 1
            else:
                v += 1
        else:
            if v < vmax - 1:
                if h == 0:
                    v += 1
                else:
                    v += 1
                    h -= 1
            else:
                h += 1
    return output

def scan_inverse(array, vmax, hmax):
    """
    Odwrotne skanowanie zigzag, aby przywrócić macierz.

    Argumenty:
    array (np.array): tablica zeskanowanych elementów
    vmax (int): maksymalna liczba wierszy w macierzy
    hmax (int): maksymalna liczba kolumn w macierzy

    Wyniki:
    np.array: odwrócona zeskanowana macierz
    """

    output = np.zeros((vmax, hmax))
    v, h, i = 0, 0, 0

    # Odwrotne skanowanie zigzag
    while v < vmax and h < hmax:
        output[v, h] = array[i]
        i += 1
        if (v + h) % 2 == 0:
            if h < hmax - 1:
                if v == 0:
                    h += 1
                else:
                    v -= 1
                    h += 1
            else:
                v += 1
        else:
            if v < vmax - 1:
                if h == 0:
                    v += 1
                else:
                    v += 1
                    h -= 1
```

```
            else:
                h += 1
        if v == vmax - 1 and h == hmax - 1:
            output[v, h] = array[i]
            break
    return output



def DCT_code(cover_image, message):
    """
    Koduje komunikat w nośniku przy użyciu metody DCT.

    Argumenty:
    cover_image (np.array): nośnik
    message (np.array): komunikat

    Wyniki:
    np.array: steganogram
    """

    height, width = cover_image.shape[:2]
    while height % 8:
        height += 1
    while width % 8:
        width += 1

    # Przygotowanie obrazu do kodowania
    cover_padded_image = cv2.resize(cover_image, (width, height))
    cover_image_f32 = np.float32(cover_padded_image)
    cover_image_YCC = cv2.cvtColor(cover_image_f32, cv2.COLOR_BGR2YCrCb)

    # Konwersja komunikatu do bitów
    bits = np.unpackbits(np.array(message, dtype=np.uint8))
    bit_count, data_len = 0, len(bits)

    steganogram = np.empty_like(cover_image_f32)
    for channel_index in range(3):

        # Podział kanału na bloki 8x8
        channel = [
            horiz_slice for vert_slice in np.vsplit(
                cover_image_YCC[:, :, channel_index],
                int(cover_image_YCC[:, :, channel_index].shape[0] / 8)
            )
            for horiz_slice in np.hsplit(
                vert_slice,
                int(cover_image_YCC[:, :, channel_index].shape[1] / 8)
            )
        ]

        # Obliczenie DCT dla każdego bloku
        dct_blocks = [cv2.dct(block) for block in channel]
        dct_quants = [np.around(np.divide(item,matrix)) for item in dct_blocks]

        # Skanowanie zigzag
        coefficients = [scan(m) for m in dct_quants]

        if channel_index == 0:
```

```
            # Kodowanie bitów komunikatu w współczynniki DCT
            for block in coefficients:
                for i in range(1, len(block)):
                    if bit_count >= data_len:
                        break
                    if np.int32(block[i]) > 1:
                        if bits[bit_count]:
                            block[i]=np.float32(np.uint8(block[i])| 0b00000001)
                        else:
                            block[i]=np.float32(np.uint8(block[i])& 0b11111110)
                        bit_count += 1
            if bit_count < data_len:
                raise ValueError("Message too big.")

        # Odwrócony zigzag scan
        desorted_coeff = [scan_inverse(m, 8, 8) for m in coefficients]

        # Odwrócenie kwantyzacji i IDCT
        dct_dequants=[np.multiply(data,matrix) for data in desorted_coeff]
        idct_blocks = [cv2.idct(block) for block in dct_dequants]

        # Łączenie bloków w obraz
        image_rows = []
        temp = []
        for i, block in enumerate(idct_blocks):
            if i > 0 and not (i % int(cover_image_YCC.shape[1] / 8)):
                image_rows.append(temp)
                temp = [block]
            else:
                temp.append(block)
        image_rows.append(temp)
        steganogram[:, :, channel_index] = np.block(image_rows)

    # Konwersja z powrotem do przestrzeni BGR
    steganogram_BGR = cv2.cvtColor(steganogram, cv2.COLOR_YCR_CB2BGR)
    steganogram_image = np.uint8(np.clip(steganogram_BGR, 0, 255))

    return steganogram_image

def DCT_decode(steganogram_image, size):
    """
    Dekoduje komunikat ze steganogramu zakodowanego przy użyciu metody DCT.

    Argumenty:
    steganogram_image (np.array): steganogram
    size (int): rozmiar komunikatu do zdekodowania

    Wyniki:
    np.array: zdekodowany komunikat
    """

    steganogram_f32 = np.float32(steganogram_image)
    steganogram_YCC = cv2.cvtColor(steganogram_f32, cv2.COLOR_BGR2YCrCb)

    # Podział kanału na bloki 8x8
    channel = [
        horiz_slice for vert_slice in np.vsplit(
            steganogram_YCC[:, :, 0],
```

```
            int(steganogram_YCC[:, :, 0].shape[0] / 8))
        for horiz_slice in np.hsplit(
            vert_slice,
            int(steganogram_image_YCC[:, :, 0].shape[1] / 8)) ]

    # Obliczenie DCT dla każdego bloku
    dct_blocks = [cv2.dct(block) for block in channel]
    dct_quants = [np.around(np.divide(item, matrix)) for item in dct_blocks]

    # Skanowanie zigzag
    coefficients = [scan(m) for m in dct_quants]

    # Odczytanie zakodowanych bitów z współczynników DCT
    bits = [
        np.uint8(block[i]) & 0x01
        for block in coefficients
        for i in range(1, len(block))
        if np.int32(block[i]) > 1 ]

    # Złożenie bitów w komunikat
    message = np.packbits(bits[:size * 8])

    return message
```

Kod 4 - Algorytmy kodowania i dekodowania w *technice DCT*
Źródło: *Opracowanie własne*

Technika *DCT* charakteryzuje się dużą *odpornością* na *kompresję stratną*, ponieważ modyfikacje są wprowadzane bezpośrednio w *domenie częstotliwościowej* co sprawia, że *komunikat* jest mniej podatny na degradację podczas działania *kodeków JPEG*. Modyfikacje współczynników *DCT*, zwłaszcza tych odpowiadających średnim częstotliwościom, są trudne do wykrycia przez ludzkie oko, ponieważ mają minimalny wpływ na jakość wizualną obrazu *(Watson, 1993)*.

Przykładem techniki *steganograficznej* bazującej na modyfikacjach współczynników *transformaty DCT* jest technika QIM (Quantization Index Modulation) zaproponowana w 2001 roku przez B. Chena i G. Wornella w artykule "Quantization Index Modulation: A Class of Provably Good Methods for Digital Watermarking and Information Embedding" *(Chen i Wornell, 2001)*. Technika ta polega na modyfikacji indeksów kwantyzacji, co pozwala na ukrywanie *komunikatów* w sposób zapewniający zarówno dużą *niewykrywalność,* jak i *odporność*. Autorzy przedstawiają teoretyczne podstawy proponowanej metody oraz dowody potwierdzające jej skuteczność w kontekście kompromisu między *niewykrywalnością* a *pojemnością*. Praca omawia zastosowania QIM w *znakowaniu wodnym* oraz *steganografii* podkreślając jej odporność na kompresję, *szum* i inne formy zniekształceń.

Kolejnym przykładem techniki wykorzystującej modyfikacje *DCT* jest praca *"A steganographic method based upon JPEG and quantization table modification"* z 2002 roku autorstwa C. Changa, T. Chena i L. Chunga *(Chang, Chen i Chung, 2002)* przedstawiająca metodę *steganograficzną*, która wykorzystuje modyfikację tabeli *kwantyzacji JPEG* do kodowania *komunikatów* w obrazach cyfrowych poprzez modyfikację współczynników *DCT* w zakresie średnich częstotliwości. Dzięki takiemu podejściu metoda charakteryzuje się dużą *pojemnością* oraz *odpornością* m.in. na *kompresję stratną*, co zwiększa jej skuteczność i trudność wykrycia przez techniki *steganoanalizy statystycznej*.

2.2.2.5 Techniki steganograficzne lingwistyczne

Steganografia lingwistyczna jest metodą ukrywania *komunikatów* w tekstach naturalnych poprzez subtelne modyfikacje niewidoczne lub nieznaczące dla ludzkiego oka. Technika ta obejmuje manipulacje na poziomie znaków, formatowania, struktur językowych oraz wykorzystanie niewidocznych znaków. Metoda została opisana przez K. Bennetta w pracy "Linguistic Steganography: Survey, Analysis, and Robustness Concerns for Hiding Information in Text" z 2004 roku *(Bennett K. , 2004)*, w której szczegółowo analizuje różne techniki *steganografii lingwistycznej,* koncentrując się na ich przeglądzie, analizie i ocenie *niewykrywalności*. Bennett porównuje różne algorytmy *steganografii lingwistycznej* podkreślając wyzwania specyficzne dla analizy języka naturalnego. Autor omawia różne metody, takie jak modyfikacja składni, zamiana słów na synonimy i użycie akronimów oraz techniki *steganoanalizy*, w tym w szczególności *steganoanalizy statystycznej*. Szczególną uwagę poświęca kwestii *niewykrywalności* metod *steganograficznych* zarówno przez automatyczne techniki *steganoanalizy*, jak i percepcyjne testy ludzkie. W zakończeniu artykułu wskazane są wyzwania i propozycje kierunków przyszłych badań, które mogą poprawić *niewykrywalność steganografii lingwistycznej*.

Na poziomie znaków *komunikaty* mogą być ukrywane poprzez wymianę znaków na podobne wizualnie. Na przykład, litera *"o"* może zostać zastąpiona cyfrą *"0"* o podobnym wyglądzie, ale różnym kodzie binarnym. W tym celu mogą być wykorzystane również różne kodowania znaków, np. ASCII (American Standard Code for Information Interchange) - opracowany w latach 60-tych XX wieku standard kodowania znaków wykorzystujący do reprezentacji znaków 7 *bitów* oraz Unicode - uważany dzisiaj za powszechny standard kodowania tekstu, przypisujący unikalny kod o wielkości do 32 *bitów* każdemu znakowi niezależnie od platformy i języka.

Manipulacje formatowaniem obejmują np. dodawanie dodatkowych spacji, tabulatorów lub znaków końca linii w miejscach niewidocznych podczas normalnego czytania. Dodatkowe spacje mogą być dodawane np. po niektórych słowach lub ukryte w odstępach między akapitami. Do ukrywania komunikatów może służyć również użycie różnych czcionek, rozmiarów i stylów bez zmiany ogólnego wyglądu tekstu, np. zamiana w wybranych miejscach czcionki Times New Roman na *Times New Roman Italic*.

Manipulacje strukturą tekstu obejmują np. użycie synonimów do zamiany słów w sposób niezmieniający znaczenia tekstu oraz zmiany struktury zdań poprzez dodawanie lub usuwanie przysłówków i przyimków.

Steganografia lingwistyczna wykorzystuje również znaki specjalne, takie jak niewidoczne spacje oraz znaki kontrolne, które nie wpływają widocznie na tekst, ale mogą być odczytane przez odpowiednie oprogramowanie, np. między kolejnymi literami słów mogą być dodane znaki o zerowej szerokości.

2.2.2.6 Techniki steganograficzne nadmiarowe

Steganografia nadmiarowa[52] wykorzystuje dostępne w wielu typach *nośników* cyfrowych pola *metadanych*, które nie są niezbędne do przekazania podstawowych istotnych *informacji* zawartych w *nośniku*. Przykładem są pliki obrazów, w których dostępne są pola EXIF (Exchangeable Image File Format) - standard *metadanych* używany w plikach graficznych zawierający informacje techniczne o zdjęciu, w tym model aparatu, ustawienia ekspozycji, czas migawki, przysłonę, ISO, datę, czas wykonania fotografii i dane geolokalizacyjne oraz pola IPTC (International Press Telecommunications Council) - standard *metadanych* używany w zarządzaniu i publikacji treści medialnych pozwalający na zapisanie informacji opisowych, w tym tytuł zdjęcia, słowa kluczowe, dane autora i opis praw autorskich. Natomiast w przypadku różnych formatów dokumentów, takich jak np. DOCX, XLSX, PDF, *nagłówki* pliku obejmują takie *metadane* jak np. dane o autorze, tytule, słowach kluczowych, komentarze.

Techniki *steganografii nadmiarowej (Castiglione, De Santis i Soriente, 2007)* polegają na użyciu dodatkowych pól danych plików cyfrowych do zakodowania

[52] **Steganografia nadmiarowa** - metoda *steganograficzna* wykorzystująca do zakodowania *komunikatu* w przestrzeni *metadanych, w tym nagłówków*, co pozostaje bez wpływu na podstawową funkcjonalność pliku cyfrowego *steganogramu (Castiglione, De Santis i Soriente, 2007).*

komunikatów. Mogą być one zapisane wprost, na przykład poprzez dodanie ukrytego tekstu do pola *metadanych* lub dodatkowo zakodowane przy wykorzystaniu innych technik *steganograficznych* takich jak np. *steganografia lingwistyczna*.

2.2.2.7 Techniki steganograficzne wykorzystujące uczenie maszynowe

Jednym z głównych podejść w *steganografii maszynowej*[53] jest wykorzystanie metod *uczenia maszynowego*[54], w tym *sieci neuronowych*[55], takich jak konwolucyjne sieci neuronowe *CNN*[56], lub generacyjne sieci przeciwstawne *GAN*[57].

CNN są często wykorzystywane do analizowania i modyfikowania obrazów na poziomie pikseli w celu kodowania *komunikatów (Khan, Sohail, Zahoora i Qureshi, 2020)*. Z kolei *GAN*, składające się z dwóch *sieci neuronowych* (generatora i dyskryminatora) uczą się wspólnie: generator tworzy obrazy z ukrytym *komunikatem*, a dyskryminator stara się odróżnić *steganogramy* z ukrytymi *komunikatami* od *nośników* bez zakodowanych *komunikatów*. Proces ten prowadzi do generowania *steganogramów* charakteryzujących się dużą *niewykrywalnością (Zhang, Dong i Liu, 2018)*.

---

[53] **Steganografia maszynowa** - klasa metod *steganograficznych* wykorzystujących *sieci neuronowe* i mechanizmy *uczenia maszynowego*.

[54] **Uczenie maszynowe** - dziedzina sztucznej inteligencji, w której systemy uczą się na podstawie danych, zamiast być bezpośrednio programowane. Modele przetwarzają dane i dostosowują swoje parametry, aby poprawić wydajność w zadaniach takich jak klasyfikacja, prognozowanie lub rozpoznawanie wzorców *(Goodfellow, Bengio i Courville, Deep Learning, 2016)*.

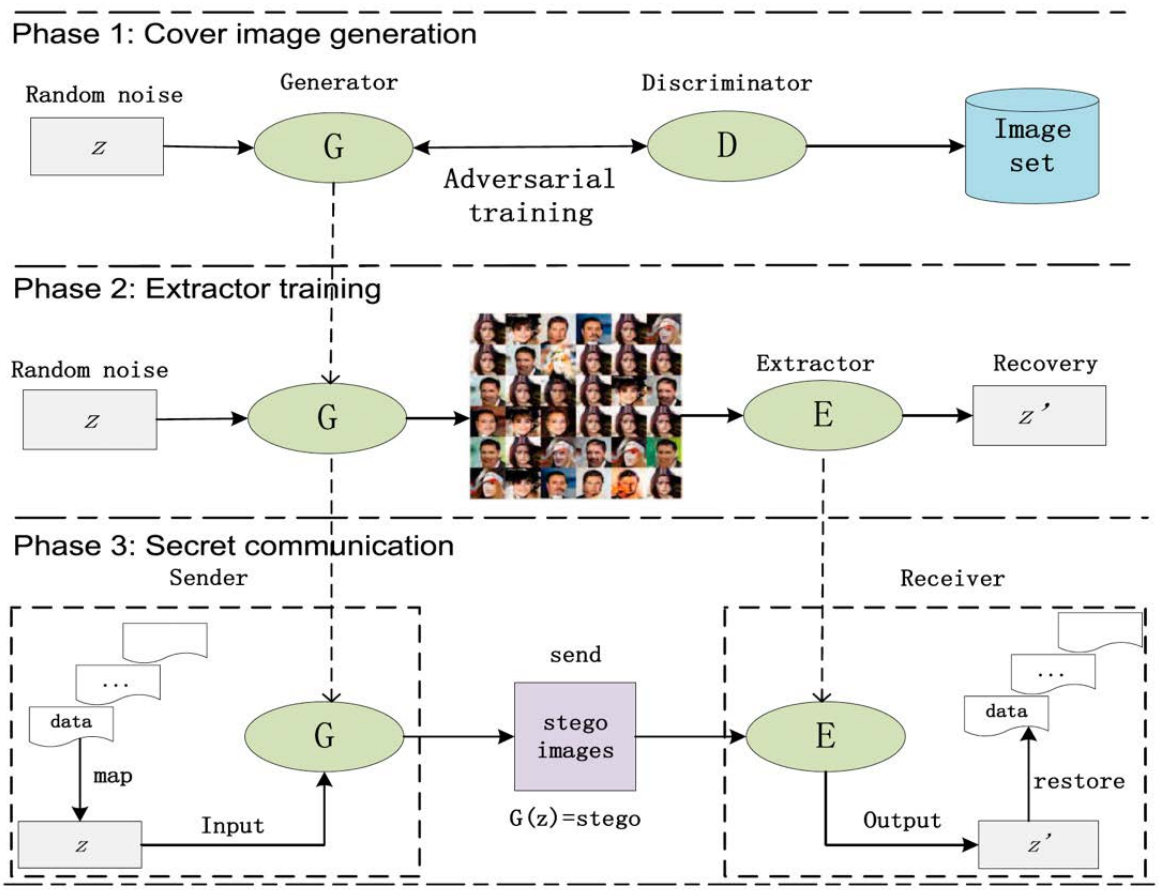
[55] **Sieć neuronowa** - model obliczeniowy inspirowany mózgiem, stosowany w obszarach sztucznej inteligencji. Składa się z warstw neuronów, które uczą się rozpoznawać wzorce poprzez dostosowywanie wag połączeń. *Sieć neuronowa* jest używana m.in. do rozpoznawania obrazów ucząc się generalizować na dużych zbiorach obrazów *(Goodfellow, Bengio i Courville, Deep Learning, 2016)*.

[56] **CNN (Convolutional Neural Network)** - konwolucyjne *sieci neuronowe*, specjalistyczne *sieci neuronowe* efektywne w przetwarzaniu danych o strukturze siatki, np. obrazów. Składają się z warstw konwolucyjnych, które filtrują obraz, warstw aktywacji wprowadzających nieliniowość oraz warstw splotowych redukujących wymiarowość mapy cech. Na końcu znajdują się warstwy w pełni połączone do klasyfikacji lub regresji. *CNN* są powszechnie stosowane w rozpoznawaniu obrazów, analizie *wideo*, przetwarzaniu języka naturalnego i generowaniu treści. Wymagają dużych zbiorów danych do skutecznego trenowania *(Krizhevsky, Sutskever i Hinton, 2017; LeCun, Bengio i Hinton, 2015)*.

[57] **GAN (Generative Adversarial Networks)** - generatywne sieci przeciwstawne, technika głębokiego uczenia składająca się z dwóch rywalizujących *sieci neuronowych*: generatora, który tworzy dane, oraz dyskryminatora, który ocenia, czy dane są prawdziwe czy wygenerowane. Proces ten prowadzi do coraz lepszego generowania danych przez generator *(Goodfellow i inni, 2014)*.

W *steganografii maszynowej* proces ukrywania *komunikatu* polega na wprowadzeniu danych do *nośnika* multimedialnego za pomocą wytrenowanego modelu *sieci neuronowej*. Model ten jest zdolny do znajdowania optymalnych miejsc w *nośniku*, które mogą być zmodyfikowane tak, aby efekty były niewidoczne dla ludzkiego oka. Dekodowanie *komunikatu* polega natomiast na odczytywaniu ukrytych danych ze *steganogramu* za pomocą odwrotnego procesu, wykorzystując ten sam lub inny model *sieci neuronowej*. Modele te są trenowane na dużych zbiorach danych, aby nauczyć się efektywnych wzorców kodowania i dekodowania *komunikatów*, co zwiększa ich *niewykrywalność* i *pojemność*. Dzięki możliwościom uczenia się wzorców, modele te mogą przewidywać i unikać zmian, które mogłyby być wykryte przez różne analizy statystyczne. Efekt ten sprawia, że *steganografia maszynowa* może osiągać większe wartości wskaźnika *niewykrywalności* w porównaniu do tradycyjnych metod.

Jednym z przykładów *techniki steganograficznej,* wykorzystującej jednocześnie sieci *CNN* i *GAN,* jest technika zaproponowana w artykule *"A Novel Image Steganography Method via Deep Convolutional Generative Adversarial Networks"* autorstwa D. Hu'a, L. Wanga, W. Jianga, S. Zhenga i B. Li'ego z 2018 roku *(Hu, Wang, Jiang, Zheng i Li, 2018)*. Zaproponowana przez autorów metoda wykorzystuje głębokie konwolucyjne generacyjne sieci przeciwstawne *CNN-GAN* do kodowania *komunikatów* w obrazach cyfrowych. Proponowana metoda łączy możliwości głębokiego uczenia *CNN* i sieci *GAN* w celu wygenerowania *steganogramów* zachowując jednocześnie ich dużą *niewykrywalność*. Autorzy szczegółowo opisują proponowaną architekturę sieci, proces szkolenia modelu oraz eksperymentalne wyniki, które według autorów potwierdzają skuteczność i przewagę proponowanej metody nad innymi technikami *steganograficznymi* w kontekście zarówno *pojemności* informacyjnej, jak i *niewykrywalności*. Na rysunku 16 przedstawiono schemat działania proponowanej sieci *CNN-GAN*.



Rysunek 16 - Przykład: schemat działania sieci *steganograficznej CNN-GAN*
Źródło: *(Hu, Wang, Jiang, Zheng i Li, 2018)*

W tym samym roku W. Zhang, H. Wang, D. Wen i Y. Dai zaproponowali w artykule *"Invisible steganography via generative adversarial networks"* technikę *steganografii obrazowej* przy użyciu generacyjnych sieci przeciwstawnych *GAN (Zhang, Dong i Liu, 2018)*. Główne założenie tej metody polega na wykorzystaniu sieci *GAN* do generowania obrazów, które zawierają ukryte *informacje* w sposób niewidoczny dla ludzkiego oka. W tym celu sieć generacyjna tworzy *steganogramy*, podczas gdy dyskryminator ocenia ich jakość, co sukcesywnie w ramach procesu trenowania sieci prowadzi do poprawy jakości *steganogramów* i zwiększenia ich *niewykrywalności* zarówno przez osoby trzecie, jak też automatyczne systemy detekcji. Praca ta szczegółowo opisuje architekturę używanych sieci *GAN*, proces treningowy oraz wyniki eksperymentów przeprowadzonych na różnych zestawach danych. Wyniki badań przedstawionych w pracy pokazują, że proponowana metoda jest efektywna w ukrywaniu *informacji* oferując wysoki poziom *niewykrywalności* w porównaniu do innych technik *steganograficznych*. Autorzy stawiają w swojej pracy tezę,

że techniki *uczenia maszynowego* w obszarze *steganografii* stanowią bardzo obiecujący kierunek dalszych badań nad *steganologią*.

Aktualne badania koncentrują się na rozwijaniu algorytmów *uczenia maszynowe*go, które zwiększają *pojemność* i *niewykrywalność* procesu *steganografii maszynowej* dla różnych typów *nośników*. Uważa się *(Hu, Wang, Jiang, Zheng i Li, 2018)*, że dalszy rozwój w dziedzinie *uczenia maszynowego*, w szczególności w rozwoju *sieci neuronowych*, może odgrywać kluczową rolę w ewolucji *steganografii* i jej aplikacji w nowoczesnych systemach informacyjnych.

2.2.3 Podobieństwa oraz różnice pomiędzy steganografią i kryptografią

Zarówno *steganografia*, jak i *kryptografia*[58] odgrywają istotną rolę w zapewnianiu bezpieczeństwa w *systemach komunikacyjnych*. Pomimo, że oba podejścia mają wspólny cel ochrony *informacji* przed nieautoryzowanym dostępem, różnią się one celami i metodami działania. *Steganografia* koncentruje się na ukrywaniu samego faktu istnienia *informacji* w różnych *nośnikach*, a jej głównym celem jest takie zakodowanie *komunikatu*, aby osoba trzecia nie była świadoma jej istnienia. Skuteczność *steganografii* polega więc na tym, żeby ukryty *komunikat* pozostawał niewidoczny dla wszystkich poza *nadawcą* i *odbiorcą*. Z kolei *kryptografia* zajmuje się zabezpieczaniem treści *komunikatu* przed nieautoryzowanym dostępem. W przeciwieństwie do *steganografii*, w *kryptografii komunikat* jest zazwyczaj widoczny dla każdego, ale jego treść jest zakodowana w taki sposób, że tylko osoby posiadające odpowiednie *klucze kryptograficzne*[59] mogą ją odczytać. Proces *szyfrowania*[60]

---

[58] **Kryptografia** - dziedzina nauki zajmująca się zabezpieczaniem *informacji* poprzez ich *szyfrowanie*, czyli przekształcanie w formę nieczytelną dla osób nieuprawnionych, z użyciem algorytmów i *kluczy kryptograficznych*, umożliwiająca poufną komunikację i ochronę danych *(Blackledge, 2011; Beckett, 1988)*

[59] **Klucz kryptograficzny** - tajny ciąg *bitów* używany w algorytmach *kryptograficznych* do *szyfrowania* i deszyfrowania informacji, zapewniający bezpieczeństwo komunikacji poprzez kontrolowanie dostępu do zaszyfrowanych danych *(Chandra, Paira, Alam i Sanyal, 2014)*.

[60] **Szyfrowanie** - proces przekształcania *informacji* w sposób, który uniemożliwia jej odczytanie bez odpowiedniego *klucza kryptograficznego*, stosowany w *kryptografii* w celu ochrony danych przed nieautoryzowanym dostępem *(Beckett, 1988)*.

przekształca zawartość *komunikatu* w formę niezrozumiałą dla osób nieuprawnionych, nie ukrywając jednak samego faktu jej przesyłania.

Jedną z pierwszych systematycznych prób opisu współistnienia *steganografii* z *kryptografią* była książka *"Top Secret: A Handbook of Codes, Ciphers, and Secret Writing"* autorstwa P. Janeczko z 2004 roku *(Janeczko, 2004)*. Jest to kompleksowy przewodnik po sztuce *szyfrowania* i ukrywania *informacji*. Stanowi praktyczne wprowadzenie do świata *kryptografii* prezentując szeroki zakres technik *szyfrowania*, ale też ukrywania i kodowania *informacji* - od prostych metod po bardziej zaawansowane systemy. Autor szczegółowo omawia różnice między kodami a szyframi, przedstawia historyczne konteksty ich stosowania oraz oferuje praktyczne wskazówki dotyczące tworzenia i łamania szyfrów. Janeczko proponuje tworzenie własnego zestawu technik kodowania, a publikacja zawiera liczne ćwiczenia praktyczne oraz ciekawostki historyczne, które ilustrują praktyczne znaczenie algorytmów kodowania i dekodowania *informacji*.

Z kolei w książce *"Cryptography and Steganography: New Algorithms and Applications"* z 2011 roku J. Blackledge połączył teorię *kryptografii* i *steganografii*, opisując różne algorytmy z obu dziedzin i przykłady ich praktycznego zastosowania analizując metody zapewniania bezpiecznej wymiany *informacji* począwszy od tradycyjnych technik wojskowych po nowoczesne zastosowania w różnych sektorach przemysłu *(Blackledge, 2011)*. Blackledge szczegółowo opisuje techniki *kryptograficzne* i *steganograficzne*, które mogą być stosowane m.in. do ochrony danych oraz przedstawia różne podejścia do zabezpieczania *informacji* uwzględniając ówczesne wyzwania i zagrożenia związane z *cyberbezpieczeństwem*[61].

W niniejszej pracy przyjęto założenie, że funkcja *steganograficzna* nie posiada dodatkowego argumentu w postaci *klucza kryptograficznego*, co oznacza brak zaimplementowanej w ramach procesu *steganografii* funkcjonalności *szyfrowania.* W literaturze *(Rahmani, Arora i Pal, 2014; Ahn i Hopper, 2004)* rozważa się jednak również inne podejście, w którym *steganografia* jest wyposażona w funkcjonalność *kryptograficzną*

[61] **Cyberbezpieczeństwo** - ochrona informacji i systemów informacyjnych przed nieautoryzowanym dostępem, wykorzystaniem, ujawnieniem, zakłóceniem, modyfikacją lub zniszczeniem w celu zapewnienia poufności, integralności i dostępności *(Nieles, Dempsey i Pillitteri, 2017)*.

i określana jest wówczas jako *steganografia z kluczem*[62]. W takim podejściu *klucz kryptograficzny* jest używany do *szyfrowania komunikatu*, co dodaje dodatkową warstwę ochrony, uniemożliwiając odczytanie nieautoryzowanym osobom *komunikatu* nawet w przypadku wykrycia faktu jego istnienia i zdekodowania jego zaszyfrowanej postaci.

Pomimo tego, że tej pracy przyjęto prostszą definicję *steganografii* bez uwzględnienia elementu *kryptografii*, warto podkreślić, że integracja *steganografii* z *kryptografią* jest jak najbardziej możliwa i często wręcz pożądana. Takie podejście łączy zalety obu technik, oferując zarówno ukrycie istnienia *komunikatu*, jak i jego *szyfrowanie*. Połączenie tych dwóch technik może być szczególnie istotne wtedy, gdy sama *steganografia* może nie być wystarczająca do zapewnienia pełnego bezpieczeństwa ukrywania *informacji*.

W literaturze można znaleźć liczne przykłady wykorzystania *steganografii z kluczem*, które demonstrują skuteczność takiego podejścia. L. Ahn wraz ze współautorami w opublikowanym w 2004 roku artykule *"Public-Key Steganography" (Ahn i Hopper, 2004)* przedstawia koncepcję *steganografii z kluczem* z wykorzystaniem publicznego *klucza kryptograficznego*. Autorzy m.in. opisują protokół, który umożliwia dwóm stronom, które nigdy wcześniej nie wymieniały *informacji*, przesyłanie *komunikatów* przez publiczny *kanał* w sposób niezauważalny dla potencjalnej osoby podsłuchującej.

Z kolei w artykule *"A Crypto-Steganography: A Survey"* z 2014 roku autorstwa K. Rahmaniego, K. Arory'ego i N. Pala *(Rahmani, Arora i Pal, 2014)* przedstawiony jest przegląd różnych technik *kryptografii* i *steganografii*. Autorzy analizują to, jak *steganografia z kluczem* zwiększa *odporność* na potencjalne ataki, np. kradzieży danych. Jednocześnie w artykule przedstawiono analizę porównująca najważniejsze cechy *steganografii* i *kryptografii*, której podsumowanie umieszczono w tabeli 1.

[62] **Steganografia z kluczem (krypto-steganografia)** - metoda ukrywania *informacji* w taki sposób, że ich obecność jest trudna do wykrycia, przy czym do ukrycia i odczytania tej *informacji* używany jest dodatkowo *klucz kryptograficzny*, który zapewnia dodatkowy poziom bezpieczeństwa *(Rahmani, Arora i Pal, 2014; Ahn i Hopper, 2004)*.

Tabela 1 - Różnice pomiędzy *steganografią* a *kryptografią*
Źródło: *Opracowanie własne* na podstawie *(Rahmani, Arora i Pal, 2014)*

	Steganografia	***Kryptografia***
Definicja	ukrywanie *informacji - komunikatu*	*szyfrowanie informacji - komunikatu*
Cel	utrzymanie **faktu istnienia** *komunikatu* w tajemnicy	utrzymaniu **treści** *komunikatu* w tajemnicy
Klucze kryptograficzne	Brak	Konieczne
Nośnik	Dowolne media cyfrowe lub parametry transmisji sieciowej	Zazwyczaj oparte na tekście
Widoczność	Nigdy	Zawsze
Usługi bezpieczeństwa	Poufność, uwierzytelnianie	Poufność, dostępność, integralność, niezaprzeczalność
Ataki	*Steganografia* zostanie złamana, gdy atakujący wykryje jej użycie i/lub odczyta treść *komunikatu* **(*steganoanaliza*)**	Kryptografia zostanie złamana, gdy atakujący odczyta treść *komunikatu* **(kryptoanaliza)**
Wynik działania	*Steganogram*	Zaszyfrowany *komunikat*

W kontekście oceny bezpieczeństwa systemów *steganograficznych* ciekawym podejściem, wykorzystującym analogię do oceny systemów *kryptograficznych,* jest opracowany w 1998 roku przez J. Zöllnera i innych autorów model matematyczny zaproponowany w artykule *"Modeling the Security of Steganographic Systems" (Zöllner i inni, 1998)*. Główne założenie proponowanego przez nich modelu polega na tym, że potencjalna osoba atakująca system *steganograficzny* powinna posiadać jak najmniejszą możliwą wiedzę o tym, w jaki sposób *komunikat* jest ukrywany w *steganogramie*. Model ten wykorzystuje założenia *teorii informacji* do wykazania, że takie podejście zwiększa bezpieczeństwo systemu, analogicznie jak w przypadku znanych ataków na systemy *kryptograficzne*.

W 2004 roku C. Lin oraz W. Tsai zaproponowali podejście rozszerzające *steganografię* o mechanizm uwierzytelniania. Proponowaną metodę przedstawili w artykule *"Secret Image Sharing with Steganography and Authentication" (Lin i Tsai, 2004)*, w którym opisali pomysł kodowania *komunikatów* będących obrazami cyfrowymi z dodatkowymi możliwościami ich uwierzytelniania. *Komunikaty* są dzielone na bloki, które są uwierzytelniane i ukrywane w wybranych przez użytkownika *nośnikach*. Autorzy omawiają w swojej pracy szczegóły techniczne swojej metody, w tym algorytmy wykorzystywane do podziału i ukrywania *komunikatów* oraz wyniki przeprowadzonych eksperymentów potwierdzających skuteczność proponowanego podejścia.

2.3 Podstawy steganoanalizy

Steganoanaliza jest dziedziną badawczą koncentrującą się na identyfikacji oraz dekodowaniu *komunikatów* zakodowanych przez procesy *steganograficzne* w różnych typach *nośników*. W swoim najszerszym zastosowaniu *steganoanaliza* nie wymaga a priori wiedzy o metodach *steganograficznych* zastosowanych do zakodowania *komunikatów* ani o własności *nośników*. Techniki stosowane w *steganoanalizie* często opierają się na detekcji anomalii statystycznych, strukturalnych, a także innych nietypowych wzorców danych, które mogą wskazywać na obecność ukrytych *komunikatów* w analizowanych *nośnikach*.

Literatura przedmiotowa (Fridrich, Steganography in Digital Media: Principles, Algorithms, and Applications, 2009; Johnson i Jajodia, Exploring Steganography: Seeing the Unseen, 1998; Johnson, Duric i Jajodia, Information Hiding: Steganography and Watermarking - Attacks and Countermeasures, 2001; Pery, Zastosowanie steganologii w cyberbezpieczeństwie, 2024) wskazuje na kluczową rolę *steganoanalizy* w kontekście zapewnienia *cyberbezpieczeństwa,* jako procesu umożliwiającego wykrywanie tajnych *komunikatów* ukrytych przy użyciu różnorodnych technik *steganograficznych*.

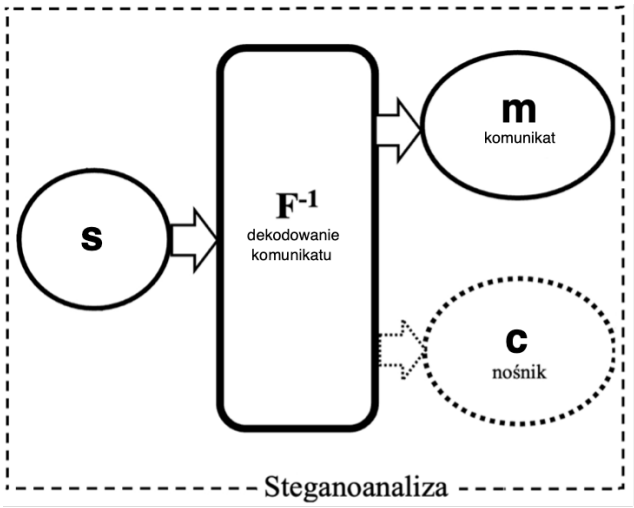
Można założyć, że dla każdej funkcji *steganograficznej* F istnieje odpowiednia funkcja *steganoanalityczna* F^{-1}, będąca funkcją odwrotną do funkcji F, która dekoduje *komunikat* ze *steganogramu*, czyli dla danego *steganogramu* s przypisuje zakodowany w nim *komunikat* m, co zdefiniowano formułą (2).

$$m := F^{-1}(s) \qquad lub \qquad (m, c) := F^{-1}(s) \tag{2}$$

gdzie: m - zdekodowany *komunikat,*
c - odtworzony *nośnik,*
s- *steganogram*.

Proces ten może odbywać się z równoczesnym odtworzeniem pierwotnego *nośnika* danych c lub bez jego odtwarzania. Odtworzenie pierwotnego *nośnika* może przyczynić się do doskonalenia systemów *steganoanalitycznych*, szczególnie w kontekście systemów wykorzystujących techniki *uczenia maszynowego*. W takich przypadkach odtworzony oryginalny *nośnik* może np. stanowić cenne źródło uczące dla głębokich *sieci neuronowych*.

Schemat funkcji *steganoanalitycznej* F^{-1} przedstawiono na rysunku 17.



Rysunek 17 - Schemat funkcji *steganoanalitycznej* F^{-1}
Źródło: *(Pery, Zastosowanie steganologii w cyberbezpieczeństwie, 2024)*

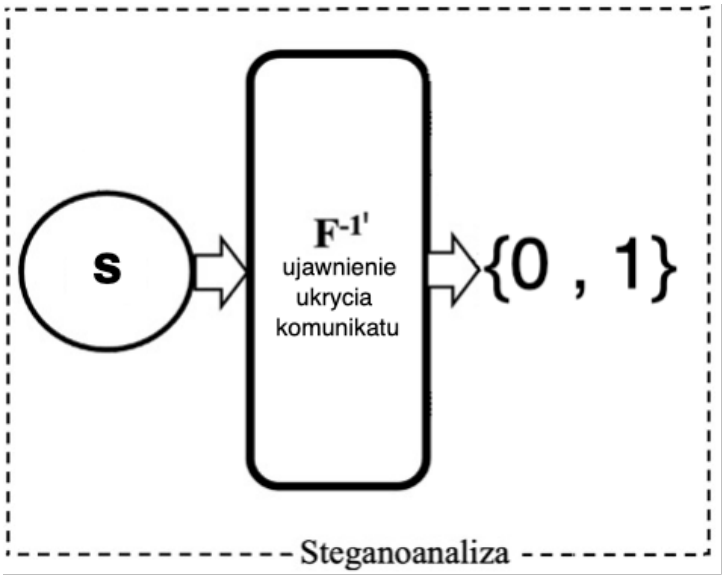
Steganoanaliza może być postrzegana jako proces odwrotny w stosunku do *steganografii*, przy czym *steganogram* stanowi dla *steganoanalizy* wartość wejściową. Wynikiem tego procesu może być albo pełne zdekodowanie *komunikatu* ukrytego w *steganogramie,* zgodnie z definicją funkcji F^{-1}, albo prosty wniosek dotyczący obecności lub braku obecności ukrytego *komunikatu* w *steganogramie*. W tym drugim przypadku możemy przyjąć, że istnieje taka uproszczona funkcja *steganoanalityczna* $F^{-1'}$, która danemu *steganogramowi* s przypisuje wartość 0 wtedy, gdy w analizowanym *nośniku* nie wykryto obecności ukrytego *komunikatu* oraz 1 wtedy, gdy w *steganogramie* wykryto obecność ukrytego *komunikatu*, co pokazano w formule (3).

$$F^{-1'}(s) \in \{0,1\} \tag{3}$$

gdzie: s- *steganogram*.

Warto podkreślić, w nawiązaniu do wcześniejszych uwag na temat podobieństw i różnic między *steganografią* a *kryptografią*, że dekodowany w procesie *steganoanalizy komunikat* nie musi zawierać jawnej *informacji*. *Komunikat* może być np. zaszyfrowany i wydobycie z niego właściwej informacji może wymagać dalszych działań, co zdefiniowano uprzednio jako *steganografię z kluczem*. Jednakże dla celów niniejszej pracy zakłada się, że celem działania funkcji dekodującej jest (tylko i aż) poprawne zdekodowanie *komunikatu* w takiej postaci, w jakiej został on zakodowany w *steganogramie*.

Schemat uproszczonej funkcji *steganoanalitycznej* $F^{-1'}$ przedstawiono na rysunku 18.



Rysunek 18 - Schemat uproszczonej funkcji *steganoanalitycznej* $F^{-1'}$
Źródło: *Opracowanie własne*

Przykład 3 - działanie funkcji odwrotnej do steganografii z przykładu 1

Funkcja *steganoanalityczna* F^{-1} odzyskuje ukryty w *steganogramie komunikat*, czyli zdjęcie psa *Groma*. Przykład funkcji odwrotnej F^{-1} do zaprezentowanego wcześniej przykładu prostej funkcji *F steganografii obrazowej LSB* w *domenie przestrzennej* przedstawiono na rysunku 19.

Rysunek 19 - Przykład: działanie funkcji *odwrotnej* F^{-1} do funkcji *F LSB*
Źródło: *Opracowanie własne*

W powyższym przykładzie odzyskany *komunikat* ze *steganogramu* jest identyczny z *komunikatem* oryginalnym.

Koniec przykładu 3

Przykład 4 - działanie funkcji odwrotnej do steganografii z przykładu 2

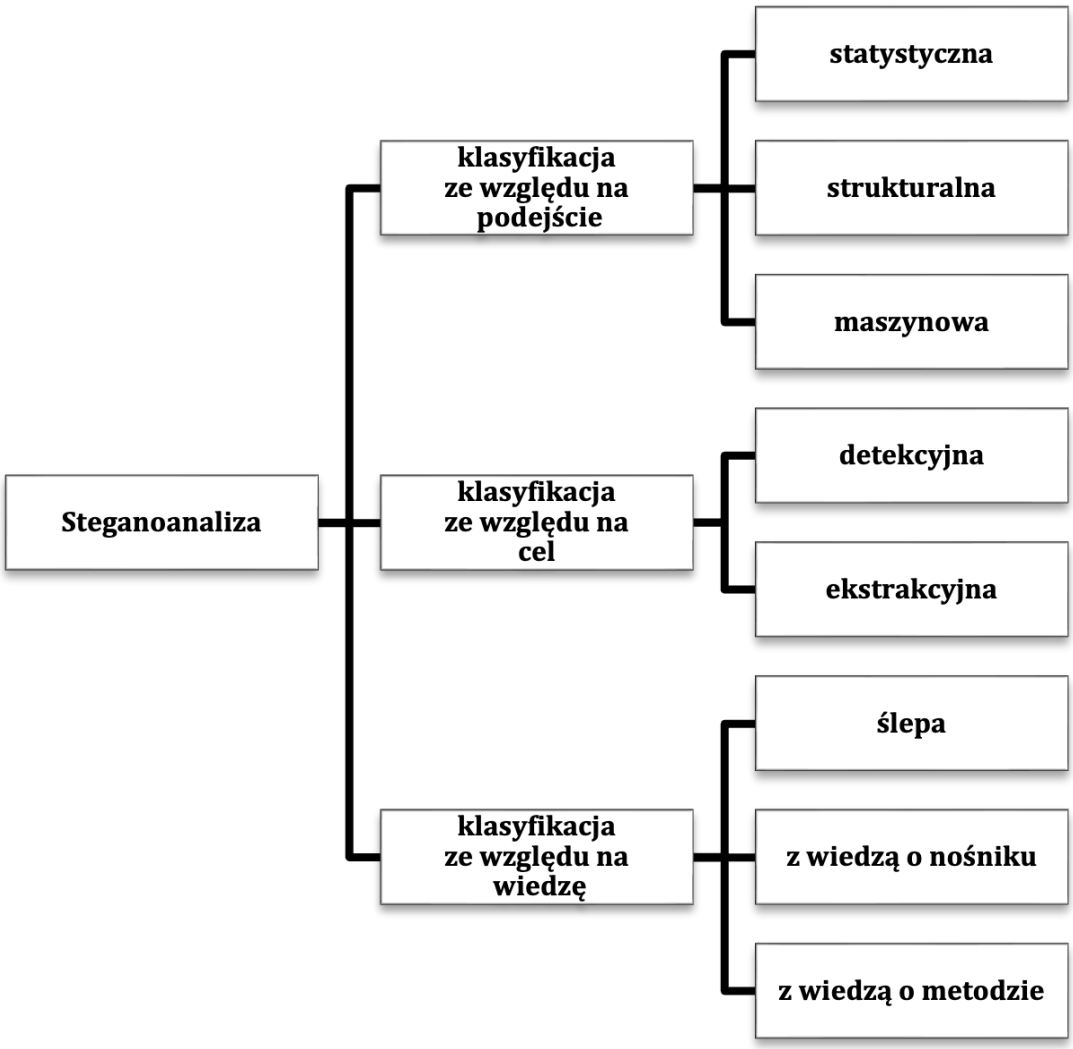
W kolejnym przykładzie funkcja odwrotna do *steganografii obrazowej DCT* w domenie *transformatowej* nie dekoduje już *komunikatu* identycznego z oryginałem. Można zauważyć gorszą jakość odzyskanego obrazu w porównaniu z oryginalnym *komunikatem*, co przedstawiono na rysunku 20.

Rysunek 20 - Przykład: działanie funkcji *odwrotnej* F^{-1} do funkcji *F DCT*
Źródło: Opracowanie własne

Koniec przykładu 4

2.3.1 Klasyfikacje metod steganoanalizy

Podobnie jak w przypadku *steganografii,* istnieje wiele różnych sposobów klasyfikowania *steganoanalizy* (Petitcolas, Anderson i Kuhn, 1999; Fridrich, Steganography in Digital Media: Principles, Algorithms, and Applications, 2009; Li, He, Huang i Shi, 2011; Fridrich, Goljan i Kodovský, Practical Steganalysis of Digital Images - State of the Art, 2002), a najpopularniejsze przedstawione zostały na rysunku 21.



Rysunek 21 - Klasyfikacje *steganoanalizy*
Źródło: *Opracowanie własne*

2.3.1.1 Klasyfikacja steganoanalizy ze względu na podejście

W tej klasyfikacji technik *steganoanalitycznych* pod uwagę bierze się metodę przeprowadzania analiz *steganogramów*. W ramach tej klasyfikacji wyróżnia się następujące typy technik:

a) Steganoanaliza statystyczna

Steganoanaliza statystyczna[63] wykrywa ukryte *komunikaty* w *nośnikach,* takich jak obrazy, dźwięki lub *wideo*, analizując statystyczne właściwości *steganogramu*.

[63] **Steganoanaliza statystyczna** - metoda *steganoanalizy* wykorzystująca analizę statystycznych właściwości *nośnika* w celu identyfikacji anomalii lub odchyleń od oczekiwanego rozkładu, np. *rozkładu normalnego (Patel i Read, 1996)*, które mogą wskazywać na obecność *steganografii*.

Polega np. na porównywaniu rozkładów statystycznych cech *steganogramu* zawierającego ukryty *komunikat* z *nośnikami* bez zakodowanych *komunikatów* - oryginalnymi lub podobnymi. Ukrywanie *komunikatów* w *nośniku* zwykle wprowadza subtelne zmiany w rozkładzie wartości pikseli, próbek dźwięku lub *klatek*, które różnią się od naturalnych wzorców. Techniki *steganoanalizy statystycznej* obejmują m.in. analizę *histogramów*, *analizę spektralną*[64] oraz metody *uczenia maszynowego*. Wykrycie potencjalnych anomalii w danych wskazuje na zwiększone prawdopodobieństwo obecności ukrytych *komunikatów*. *Steganoanaliza statystyczna* często wymaga dużych mocy obliczeniowych.

Jedną z publikacji dotyczących *steganoanalizy statystycznej* jest książka *"Advanced Statistical Steganalysis"* autorstwa R. Böhme'a opublikowana w 2010 roku *(Böhme, 2010)*, która koncentruje się na zaawansowanych metodach *steganoanalizy* w plikach cyfrowych przy wykorzystaniu m.in. analizy statystycznej. Publikacja ta przedstawia teoretyczne podstawy *steganografii* i *steganoanalizy* oraz przedstawia dyskusję wykorzystania różnych technik. Böhme proponuje strukturalne podejście do *steganoanalizy*, opierając się m.in. na wykorzystaniu szczególnych właściwości *nośników*. Książka porusza również kwestie heterogeniczności rozkładów różnych parametrów *nośników*, co jest kluczowe dla procesu przeprowadzania analiz. Dodatkową wartością jest fakt, iż autor wspiera swoje teoretyczne rozważania licznymi przykładami praktycznymi oraz propozycją ujednoliconej terminologii i notacji.

b) Steganoanaliza strukturalna

Steganoanaliza strukturalna[65] wykrywa ukryte *komunikaty* w *steganogramach* poprzez analizę potencjalnych anomalii w ich strukturze i formatach mogących powstać w wyniku zastosowania *steganografii*. Zajmuje się ona m.in. badaniem *nagłówków* i segmentów kompresji, wykrywaniem nieprawidłowości strukturalnych, takich jak nadmiarowe dane lub nieoczekiwane segmenty danych,

---

[64] **Analiza spektralna** - metoda badania sygnałów poprzez rozkład na składowe częstotliwościowe umożliwiająca identyfikację dominujących częstotliwości i amplitud *(Stoica i Moses, 2005)*.

[65] **Steganoanaliza strukturalna** - metoda *steganoanalizy* polegająca na analizie struktury *nośnika*, np. formatu pliku lub jego metadanych, w celu zidentyfikowania nieprawidłowości lub modyfikacji, które mogą sugerować zastosowanie *steganografii*.

a także analizowaniem potencjalnych *redundancji*, które mogą służyć do ukrywania danych. Zaletą *steganoanalizy strukturalnej* jest dokładność i wszechstronność. Wymaga ona jednak bardzo szczegółowej wiedzy na temat formatów *nośników*.

c) Steganoanaliza wykorzystująca uczenie maszynowe

Steganoanaliza maszynowa[66] wykorzystuje techniki sztucznej inteligencji do wykrywania ukrytych *komunikatów* w *steganogramach* utworzonych z takich *nośników* jak obrazy, pliki *audio* i *wideo*. Polega na trenowaniu modeli *sieci neuronowych* na zbiorach danych zawierających przykłady *steganogramów* z ukrytymi *komunikatami* oraz *nośników* bez zakodowanych *komunikatów*. Modele te uczą się rozpoznawać wzorce charakterystyczne dla ukrytych *komunikatów*. Proces obejmuje zbieranie danych, ekstrakcję cech, trenowanie modelu, weryfikację i walidację oraz wykrywanie ukrytych *komunikatów*.

Zaletą tych metod jest potencjalnie duża skuteczność w wykrywaniu ukrytych *komunikatów* oraz adaptacyjność do nowych technik *steganografii*. Wyzwaniem jest w tym przypadku pozyskiwanie dużych, dobrze oznaczonych zbiorów danych do trenowania modeli oraz radzenie sobie z zaawansowanymi technikami *steganograficznymi* unikającymi tworzenia powtarzalnych wzorców.

Techniki *steganoanalizy maszynowej* mogą być stosowane wszędzie tam, gdzie metody *steganoanalityczne* bazujące na algorytmach deterministycznych nie mogą być wykorzystywane np. ze względu na niewystarczającą moc obliczeniową.

2.3.1.2 Klasyfikacja steganoanalizy ze względu na cel

W tej klasyfikacji istotne jest rozróżnienie tego, czy technika *steganoanalityczna* ma jedynie odpowiedzieć na pytanie, czy w *steganogramie* jest ukryty *komunikat*, czy dodatkowo ma przeprowadzić głębszą analizę i zdekodować ukryty *komunikat*. W ramach tej klasyfikacji wyróżniamy dwa typy metod:

[66] **Steganoanaliza maszynowa** - klasa metod *steganoanalitycznych* wykorzystujących *sieci neuronowe* i mechanizmy *uczenia maszynowego*.

a) Steganoanaliza detekcyjna

   Celem *steganoanalizy detekcyjnej*[67] jest jedynie wykrycie tego, czy w *steganogramie* jest ukryty *komunikat*.

b) Steganoanaliza ekstrakcyjna

   Celem *steganoanalizy ekstrakcyjnej*[68] jest wykrycie tego, czy w *steganogramie* jest ukryty *komunikat* oraz odzyskanie ukrytego *komunikatu* ze *steganogramu*.

2.3.1.3 Klasyfikacja steganoanalizy ze względu na wiedzę

a) Steganoanaliza ślepa

   Steganoanaliza ślepa[69] *(Fridrich, Goljan i Kodovský, Practical Steganalysis of Digital Images - State of the Art, 2002)* polega na wykrywaniu ukrytych komunikatów w *steganogramie* bez żadnej wiedzy na temat oryginalnego *nośnika,* jak też użytej metody *steganograficznej*. Jest to trudne zadanie, ponieważ analiza musi polegać wyłącznie na analizie *nośnika*, który może, ale nie musi zawierać ukrytego *komunikatu*. Taka analiza często skupia się na badaniu statystycznych właściwościach *nośnika*, np. analizach *histogramów* wartości pikseli lub *analizach spektralnych* sygnałów *audio*.

   Steganoanaliza ślepa często wykorzystuje techniki wykrywania anomalii metodami *uczenia maszynowego*. Wykorzystywane w tym celu *sieci neuronowe* są trenowane na zbiorach danych zawierających zarówno *steganogramy* z ukrytymi *komunikatami*, jak i oryginalne *nośniki*, ucząc się rozpoznawać różnice wskazujące na obecność *steganografii*. W *steganoanalizie ślepej* można również wykorzystywać przekształcenia *steganogramów* do *domeny częstotliwościowej*, co może ułatwiać ujawnianie ukrytych *komunikatów*, które są trudniejsze do wykrycia w *domenie przestrzennej*.

   Steganoanaliza ślepa jest uznawana za 'najczystszą' formę *steganoanalizy,* będąc jednocześnie najtrudniejszą jej odmianą.

---

[67] **Steganoanaliza detekcyjna** - metoda *steganoanalizy*, która koncentruje się na identyfikacji obecności *steganografii* w *steganogramie*, bez konieczności wydobywania *komunikatu*.

[68] **Steganoanaliza ekstrakcyjna** - technika steganoanalizy, która nie tylko identyfikuje obecność *steganografii*, ale również próbuje wydobyć zakodowany *komunikat* ze *steganogramu* rekonstruując oryginalną *informację*.

[69] **Steganoanaliza ślepa** - metoda *steganoanalizy*, która analizuje *steganogram* bez wcześniejszej znajomości oryginalnego *nośnika* oraz użytej techniki *steganograficznej* opierając się wyłącznie na cechach *steganogramu*.

b) Steganoanaliza z wiedzą o oryginalnym nośniku

Steganoanaliza z wiedzą o nośniku[70] polega na wykrywaniu ukrytych *komunikatów* poprzez analizę i porównanie dwóch wersji tego samego pliku: oryginalnego *nośnika* bez ukrytego *komunikatu* i *steganogramu* z potencjalnie zakodowanym *komunikatem*. Dzięki dostępowi do obu wersji możliwe jest precyzyjne zidentyfikowanie różnic wskazujących na obecność ukrytego *komunikatu*. Proces ten obejmuje porównywanie plików na różnych poziomach, takich jak analiza różnic pikseli w obrazach, różnic w próbkach dźwięku w plikach *audio* lub różnic w tekstach. Dzięki dokładnemu porównaniu z oryginalnym *nośnikiem* w *steganogramie* mogą być wykryte nawet niewielkie zmiany. Dodatkowe wykorzystanie narzędzi statystycznych do analizy różnic umożliwia identyfikację ukrytych informacji na podstawie rozkładów wartości, częstotliwości występowania pewnych cech oraz innych właściwości statystycznych. Metoda ta jest efektywna, gdyż umożliwia bezpośrednie wykrywanie różnic między dwoma wersjami pliku, co znacząco zwiększa szanse na zidentyfikowanie *komunikatu*.

Przykładem zastosowania tej techniki może być analiza dwóch obrazów: jednego oryginalnego i drugiego z potencjalnie ukrytym *komunikatem*. Przeprowadzenie szczegółowego porównania *"piksel po pikselu"* pozwala na wykrycie nawet najmniejszych zmian w *steganogramie*, takich jak zmiana wartości koloru dowolnego piksela, ponieważ znamy dokładną wartość tego piksela w oryginalnym *nośniku*. Ta metoda stanowi jedną z najdokładniejszych, najskuteczniejszych i jednocześnie najprostszych technik wykrywania ukrytych *komunikatów*, zapewniając wysoki poziom pewności w identyfikacji *steganografii*. Ponieważ jednak wymaga dostępu do oryginalnego *nośnika* przed zastosowaniem procesu *steganografii*, jest jedną z metod najmniej praktycznych i najrzadziej stosowanych.

[70] **Steganoanaliza z wiedzą o nośniku** - metoda *steganoanalizy*, która korzysta z porównania *steganogramu* z oryginalnym *nośnikiem*, umożliwiając bardziej precyzyjne wykrycie i analizę *komunikatu* dzięki znajomości pierwotnych cech *nośnika*.

c) Steganoanaliza z wiedzą o metodzie steganografii

Steganoanaliza z wiedzą o metodzie[71] *steganografii* polega na wykrywaniu ukrytych komunikatów w *steganogramie* przy założeniu, że analityk posiada wiedzę na temat użytej techniki *steganograficznej*. Taka analiza umożliwia bardziej ukierunkowane i skuteczniejsze badanie zwiększając prawdopodobieństwo wykrycia ukrytych *komunikatów* w przeciwieństwie do *steganoanalizy ślepej*. Wiedza ta może obejmować znajomość specyficznego algorytmu kodującego, jego parametrów lub sposobu modyfikacji danych w *nośniku*. Na przykład w przypadku techniki *LSB*, polegającej na ukrywaniu danych w najmniej znaczących *bitach* pikseli obrazu, analiza może skoncentrować się na badaniu tylko tych *bitów* w celu wykrycia nieregularności wskazujących na obecność ukrytych *komunikatów*.

Znajomość szczegółów metody *steganograficznej* pozwala analitykowi na stosowanie odpowiednich narzędzi i metod detekcji dostosowanych do specyfiki algorytmu kodującego. Może to obejmować analizę statystyczną cech *nośnika*, takich jak *histogramy* rozkładu pikseli lub *histogramy* rozkładu częstotliwości, które mogą ujawniać charakterystyczne ślady pozostawione przez technikę *steganograficzną*. Dodatkowo analityk może symulować proces *steganograficzny* tworząc próbki *steganogramów* z ukrytymi *komunikatami* przy użyciu tej samej techniki, a następnie porównywać te próbki z analizowanym *steganogramem*. Tego rodzaju *steganoanaliza* jest wysoce skuteczna, ponieważ pozwala na skoncentrowanie wysiłków na specyficznych aspektach *nośnika* najbardziej podatnych na zmiany wynikające z wykorzystania konkretnej techniki *steganograficznej*, co umożliwia potencjalnie nie tylko wykrycie obecności ukrytego *komunikatu*, ale także jego odtworzenie.

Kategorie w *steganoanalizie* nie są wzajemnie wykluczające się i dana metoda może być jednocześnie przypisana do wielu kategorii. Przykładowo, *steganoanaliza maszynowa* może równocześnie być klasyfikowana jako metoda *steganoanalizy ślepej*.

Wybór odpowiedniej metody *steganoanalizy* jest determinowany przez szereg czynników, w tym typ *nośnika*, a także dostępność zasobów obliczeniowych.

---

[71] **Steganoanaliza z wiedzą o metodzie** - metoda *steganoanalizy*, która wykorzystuje wiedzę o technice *steganograficznej* użytej do zakodowania *komunikatu* w *steganogramie*, co pozwala na bardziej precyzyjne i skuteczne identyfikowanie oraz wydobywanie *komunikatu*.

2.3.2 Charakterystyka podstawowych technik steganoanalitycznych

W *steganoanalizie*, analogicznie jak w *steganografii*, stosuje się różnorodne techniki obejmujące m.in. analizy statystyczne, *analizy spektralne*, analizy *transformat* w *domenie częstotliwości* oraz zaawansowane algorytmy *uczenia maszynowego* wykorzystujące *sieci neuronowe* trenowane do rozpoznawania wzorców charakterystycznych dla *steganogramów*.

Wiele technik *steganoanalitycznych* bazuje na analizie *histogramów*[72] utworzonych na podstawie analizowanych *steganogramów*. W przypadku prób wykrycia technik *steganografii przestrzennej* są to najczęściej *histogramy* kolorów *RGB* lub jasności poszczególnych pikseli obrazu. Natomiast, gdy *steganoanaliza* próbuje wykryć *steganografię transformatową,* są to np. *histogramy* współczynników *transformaty DCT*. Analiza różnych anomalii występujących w obliczonych *histogramach* może wskazać fakt zastosowania jakiejś techniki *steganograficznej* i tym samym odkryć istnienie zakodowanego *komunikatu*.

Kod 5 zawiera przykład algorytmu obliczającego dla zadanego obrazu z przestrzeni *RGB* proste *histogramy* kolorów.

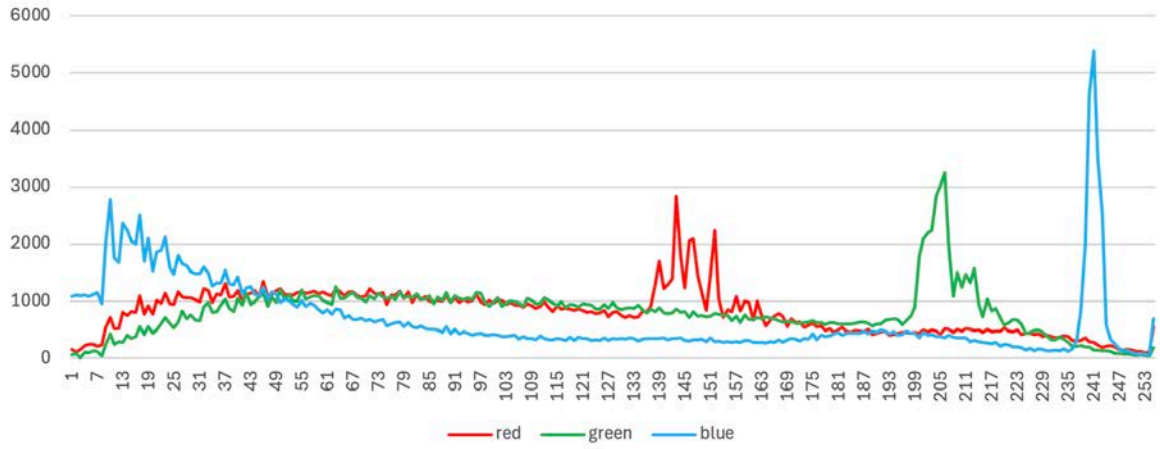
```
def RGB_histogram(image_file_path):
    """
    Oblicza histogramy dla poszczególnych kanałów kolorów (R, G, B) obrazu.
    Argumenty:
    image_file_path (string): ścieżka do pliku obrazu
    Wyniki:
    np.array: histogramy dla kanałów (R, G, B)
    """
    image = Image.open(image_file_path)
    image = image.convert('RGB')
    histograms = np.zeros((3, 256), dtype=int)  # Kanały: R, G, B
    for pixel in image.getdata():
        red, green, blue = pixel
        histograms[0, red] += 1
        histograms[1, green] += 1
        histograms[2, blue] += 1
    return histograms
```

Kod 5 - Algorytm obliczania *histogramów* kolorów z przestrzeni *RGB*
Źródło: *Opracowanie własne*

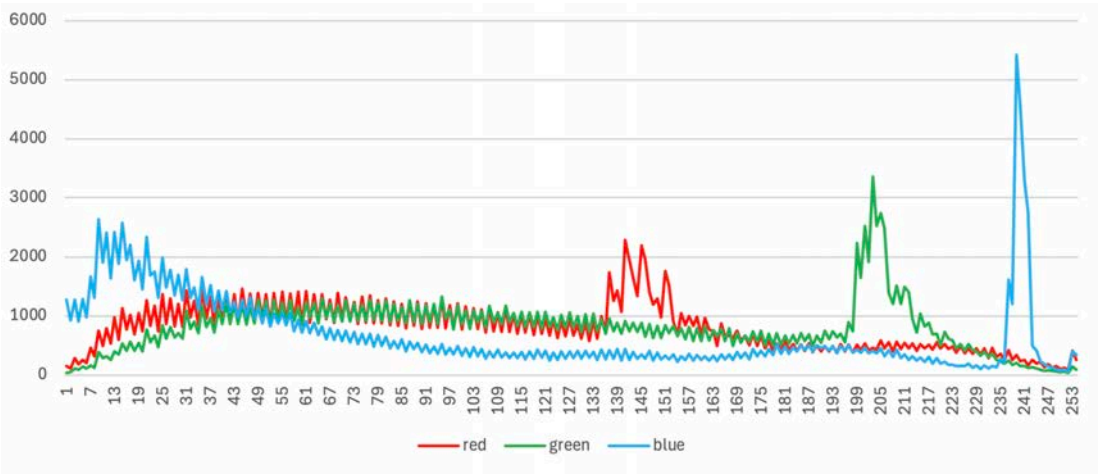
[72] **Histogram** - graficzny wykres słupkowy przedstawiający rozkład danych, gdzie oś X pokazuje przedziały wartości, a oś Y - częstotliwość ich występowania. Jest powszechnie stosowany w statystyce i analizie danych umożliwiając wizualizację rozkładu danych, identyfikację koncentracji wartości oraz wykrywanie anomalii *(Howitt i Cramer, 2007)*.

Przykład 5 - histogramy RGB nośników i steganogramów

Poniżej przedstawione są dwa *histogramy obrazów RGB*. Pierwszy, przedstawiony na rysunku 22, to *histogram nośnika* zdjęcia *El Capitan*. Drugi, przedstawiony na rysunku 23, to *histogram steganogramu* utworzonego na podstawie tego *nośnika* oraz zdjęcia *Groma* przy zastosowaniu techniki *steganograficznej LSB*.



Rysunek 22 - Przykład: *histogram RGB* przykładowego *nośnika*
Źródło: *Opracowanie własne*



Rysunek 23 - Przykład: *histogram RGB* przykładowego *steganogramu LSB*
Źródło: *Opracowanie własne*

Porównując oba *histogramy,* już na pierwszy rzut oka widać różnice pomiędzy rozkładami częstotliwości wartości poszczególnych kolorów i anomalie występujące w *histogramie steganogramu.* Zaobserwowane anomalie wskazują na istnienie zakodowanego w nim ukrytego *komunikatu*.

Koniec przykładu 5

Wykorzystywanie różnych analiz *histogramów* pikseli jest podstawą wielu technik *steganoanalitycznych* dedykowanych zwłaszcza do wykrywania użycia technik *steganograficznych* bazujących na metodzie *LSB*. Przykładem może być technika RQP do wykrywania *steganografii* w obrazach kolorowych zaproponowana przez J. Fridrich, R. Du'a i L. Menga w 2000 roku w publikacji *"Steganalysis of LSB Encoding in Color Images" (Fridrich, Du i Meng, Steganalysis of LSB encoding in color images, 2000)*. Zaproponowana przez autorów technika polega na analizie statystycznej par sąsiadujących pikseli w celu wykrycia anomalii. Opublikowane przez autorów badania wykazały, że *RQP* jest skuteczna, gdy liczba unikalnych kolorów w obrazie nie przekracza 30% liczby pikseli, a jej dokładność maleje przy większej liczbie unikalnych kolorów. Zaproponowanej metody nie można użyć w przypadku obrazów zapisanych w formatach skali szarości. Rozwinięciem tej techniki jest zaproponowana w artykule *"Detecting LSB Steganography in Color and Gray-Scale Images"* z 2001 roku *(Fridrich, Goljan i Rui, Detecting LSB steganography in color, and gray-scale images, 2001)* przez J. Fridrich, M. Goljan oraz D. Rui metoda wykrywania *steganografii* wykorzystującej *LSB* zarówno w obrazach kolorowych, jak też wykorzystujących skalę szarości. Autorzy opracowali technikę, która analizuje różnice między regularnymi i pojedynczymi grupami pikseli w płaszczyźnie *LSB* oraz przesuniętej płaszczyźnie *LSB*. Metoda ta umożliwia wykrycie obecności ukrytych *informacji* poprzez badanie tych różnic, co teoretycznie pozwala na identyfikację zakodowanych *komunikatów* nawet w przypadku losowego rozproszenia pikseli w obrazie.

Kolejnym ciekawym rozwinięciem wykorzystania analiz *histogramów* pikseli jest autorska technika *steganoanalityczna* zaprezentowana w roku 2010 roku przez T. Pevný'ego, P. Basa i J. Fridrich w artykule *"Steganalysis by Subtractive Pixel Adjacency Matrix" (Pevný, Bas i Fridrich, Steganalysis by Subtractive Pixel Adjacency Matrix, 2010)*. Przedstawiono w nim metodę, która bazuje na macierzy różnic sąsiednich pikseli SPAM (Subtractive Pixel Adjacency Matrix). Metoda ta jest zaprojektowana do wykrywania technik *steganograficznych* używanych w *domenie przestrzennej*. Autorzy pokazują, że różnice między sąsiadującymi pikselami obrazu można modelować za pomocą *procesów Markowa* pierwszego i drugiego rzędu, co pozwala na efektywne wykrywanie zakodowanych *komunikatów*. W artykule szczegółowo opisano sposób modelowania tych różnic oraz ich wykorzystania do wykrywania nietypowych wzorców, które mogą sugerować obecność ukrytych *informacji*. Badania empiryczne przeprowadzone przez autorów pokazują, że proponowana przez nich metoda jest skuteczna w wykrywaniu

różnych technik *steganograficznych* oferując dużą dokładność detekcji przy jednoczesnym minimalizowaniu fałszywych pozytywnych wskazań.

Unikalne podejście do zagadnień *steganoanalizy* zostało zaprezentowane w artykule G. Kesslera *"An Overview of Steganography for the Computer Forensics Examiner"* zaktualizowanym w 2015 roku *(Kessler, 2015)*. Przedstawiono w nim przegląd technik *steganograficznych* oraz metod *steganoanalitycznych* z perspektywy analityka kryminalistycznego. Kessler omawia znaczenie *steganografii* w kontekście komunikacji cyfrowej, gdzie *informacje* są ukrywane np. w plikach obrazów lub *audio*. Artykuł koncentruje się na praktycznych aspektach technik *steganoanalitycznych* oferując przykłady narzędzi do ukrywania i wykrywania zakodowanych *komunikatów* oraz wskazówki dla analityków w zakresie identyfikacji potencjalnych przypadków użycia *steganografii* w dochodzeniach kryminalistycznych. Kessler podkreśla, że mimo braku dokładnych statystyk dotyczących powszechności *steganografii*, jej znaczenie wzrasta, a analitycy muszą być przygotowani do jej identyfikacji i analizy w swoich badaniach.

2.3.2.1 Technika steganoanalityczna HCF (Histogram Characteristic Function)

W 2003 roku J. Harmsen i W. Pearlman opublikowali artykuł *"Steganalysis of Additive Noise Modelable Information Hiding" (Harmsen i Pearlman, 2003)* przedstawiający nowatorskie podejście do *steganoanalizy* polegające na wykrywaniu ukrytych *komunikatów* w obrazach cyfrowych za pomocą analizy funkcji charakterystycznej *histogramu HCF*. Autorzy wprowadzają koncepcję modelowania *steganografii* jako addytywnego *szumu*, co pozwala na zastosowanie zaawansowanych metod statystycznych do jej detekcji. Kluczowym elementem proponowanej metody jest analiza zmian w *histogramie* obrazu, które powstają w wyniku kodowania *komunikatu*. Harmsen i Pearlman wykazują, że funkcja charakterystyczna *histogramu HCF* jest szczególnie czuła na te modyfikacje, umożliwiając skuteczne wykrywanie *steganografii* nawet przy małych *pojemnościach steganogramów*. W pracy tej autorzy prezentują teoretyczne podstawy swojej metody, a następnie demonstrują jej skuteczność przeprowadzając różne eksperymenty na wybranych typach obrazów i przy użyciu różnych technik *steganograficznych,* w tym technik *steganografii adaptacyjnej*.

Podstawą techniki *HCF* jest analiza statystyczna *histogramu* obrazu, która pozwala na identyfikację subtelnych zmian w rozkładzie pikseli spowodowanych przez zastosowanie *steganografii*. *Histogram* obrazu przedstawia rozkład wartości pikseli, a funkcja charakterystyczna *histogramu HCF* analizuje te dane, koncentrując się na lokalnych

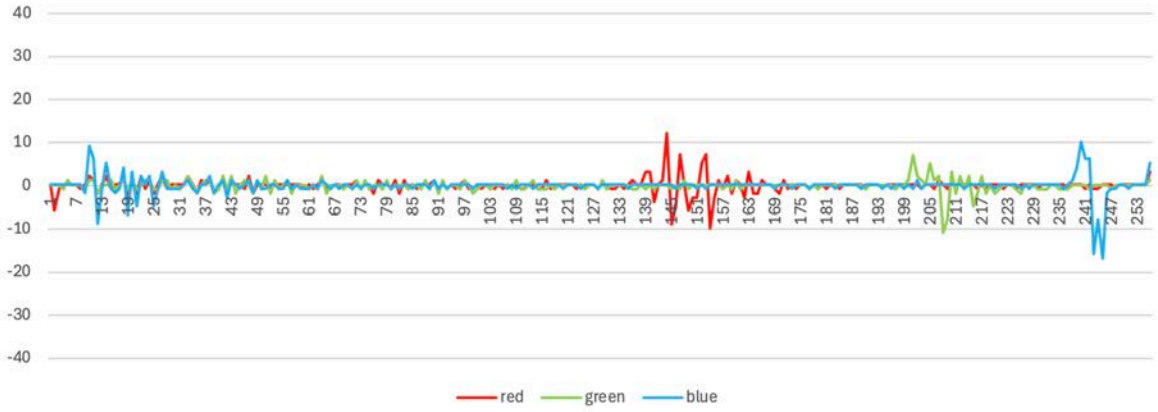
minimach i maksimach, które mogą ulec zmianie w wyniku procesu kodowania *komunikatu*. Metoda ta jest szczególnie skuteczna w wykrywaniu *steganografii* wykorzystującej techniki *LSB*. Kod 6 zawiera przykładowy algorytm wyliczający wartości funkcji charakterystycznej *histogramu HCF* dla zadanego obrazu.

```
def HCF_analysis(image_file_path):
    """
    Oblicza wartości funkcji charakterystycznej HCF obrazu.

    Argumenty:
    image_file_path (string): ścieżka do pliku obrazu

    Wyniki:
    np.array: wartości HCF
    """

    # Obliczanie histogramów dla kanałów za pomocą funkcji RGB_histogram
    histograms = RGB_histogram(image_file_path)
    hcf = []

    for hist in histograms:
        # Konwersja histogramu na tablicę NumPy i normalizacja
        hist = np.array(hist, dtype=float)
        hist = hist / hist.sum()  # Normalizacja do sumy 1

        hcf_channel = np.zeros_like(hist)

        # Obliczanie różnicy między kolejnymi wartościami histogramu
        for j in range(1, len(hist)):
            hcf_channel[j] = hist[j] - hist[j - 1]

        # Dodanie wyniku HCF dla tego kanału do listy
        hcf.append(hcf_channel)

    hcf = np.array(hcf)

    # Normalizacja HCF do zakresu 0-255
    arrmax, arrmin = hcf.max(), hcf.min()
    hcf_normalized = np.uint8(255 * (hcf - arrmin) / (arrmax - arrmin))

    # Obliczanie średniej wartości HCF z pominięciem 0 i 255
    mask = (hcf_normalized > 0) & (hcf_normalized < 255)
    mean_hcf = hcf_normalized[mask].mean()

    # Obliczanie różnicy względem średniej wartości HCF
    hcf_final = hcf_normalized - mean_hcf

    return hcf_final
```

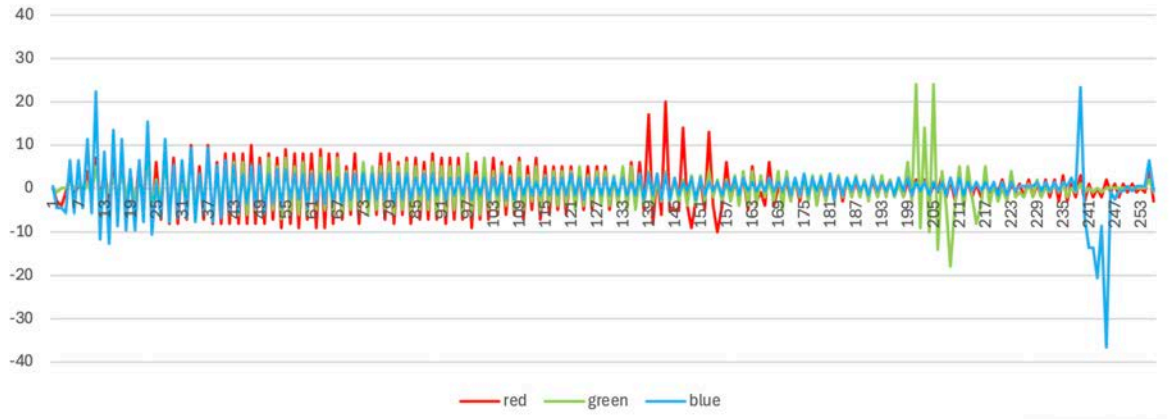
Kod 6 - Algorytm obliczania funkcji charakterystycznej *histogramu HCF*
Źródło: *Opracowanie własne*

Przykład 6 - wykresy funkcji charakterystycznych nośnika i steganogramów

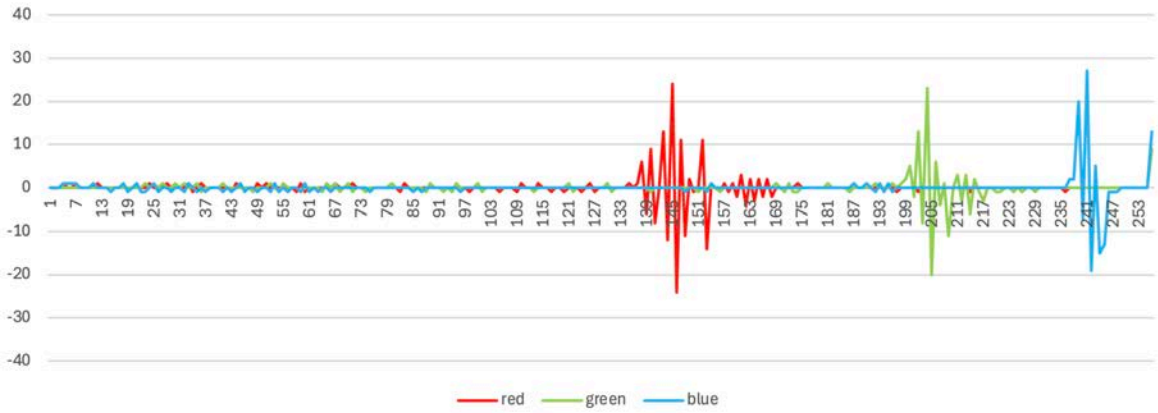
Poniżej przedstawione są wykresy funkcji charakterystycznej *histogramu HCF* dla *nośnika* - rysunek 24, *steganogramu* utworzonego przy pomocy techniki *steganografii LSB* - rysunek 25 oraz *steganogramu* utworzonego przy pomocy techniki *DCT* - rysunek 26.



Rysunek 24 - Przykład: wartości funkcji *HCF* dla *nośnika*
Źródło: *Opracowanie własne*



Rysunek 25 - Przykład: wartości funkcji *HCF* dla *steganogramu LSB*
Źródło: *Opracowanie własne*



Rysunek 26 - Przykład: wartości funkcji *HCF* dla *steganogramu DCT*
Źródło: *Opracowanie własne*

Wartości funkcji charakterystycznej *histogramu HCF* są największe dla *steganogramu LSB*, co uprawdopodabnia istnienie ukrytego *komunikatu* właśnie w tym nośniku. Interesujące są też wartości funkcji charakterystycznej *histogramu HCF* dla steganogramu *DCT*, gdyż różnią się one istotnie od wartości funkcji charakterystycznej *histogramu HCF* dla oryginalnego *nośnika*, choć na ich podstawie nie da się wprost powiedzieć, że świadczą one o ukryciu w *steganogramie DCT komunikatu*.

Koniec przykładu 6

W 2005 roku A. Ker w publikacji *"Steganalysis of LSB matching in grayscale images" (Ker, 2005)* wprowadził udoskonalenia do metody *HCF*. Autor wprowadza dwie nowe metody wykorzystania funkcji charakterystycznej *histogramu HCF*, pierwotnie zaproponowanej przez Harmsena *(Harmsen i Pearlman, 2003)* dla obrazów kolorowych, ale nieskutecznej w przypadku obrazów w skali szarości. Pierwsza metoda polega na kalibracji wyników przy użyciu obrazu o zmniejszonej rozdzielczości, druga natomiast wykorzystuje *histogram* sąsiedztwa zamiast standardowego *histogramu*. Kalibracja polega na porównaniu *histogramu* analizowanego obrazu z *histogramem* obrazu referencyjnego, który nie zawiera ukrytego *komunikatu*. Dzięki temu możliwe jest wykrycie nawet bardzo subtelnych różnic w rozkładzie wartości pikseli, ponieważ *histogram* sąsiedztwa uwzględnia korelacje między sąsiadującymi pikselami. Jako wskaźnik obecności ukrytych informacji jest używany COM (Center of Mass) - centrum masy *steganogramu* w *histogramie HCF* obliczane jako średnia ważona wartości pikseli. Wskazuje on centralny punkt *histogramu steganogramu*, którego przesunięcie może wskazywać na zastosowanie technik *steganograficznych*. Ker prezentuje obszerne wyniki eksperymentalne dowodzące, że proponowana technika *steganoanalityczna* jest bardziej skuteczna niż wcześniej znane metody wykrywania *steganografii LSB* w obrazach w skali szarości. Autor szczegółowo omawia problem dużej zmienności funkcji *HCF* dla różnych *nośników*, co stanowiło główne wyzwanie w zastosowaniu tej metody do obrazów w skali szarości.

Technika *HCF* jest ceniona za swoją dużą skuteczność w wykrywaniu *steganografii*, zwłaszcza w kontekście technik *LSB*. Jej zastosowanie w praktyce obejmuje zarówno obrazy kolorowe, jak i monochromatyczne, co czyni ją uniwersalnym narzędziem *steganoanalitycznym*.

2.3.2.2 Technika steganoanalityczna chi-kwadrat

W 1999 roku A. Westfeld i A. Pfitzmann przedstawili w artykule *"Attacks on Steganographic Systems" (Westfeld i Pfitzmann, 1999)* metodę analizy *chi-kwadrat*[73] do wykrywania zastosowania *steganografii obrazowej* wykorzystującej metodę *LSB*. Autorzy opisują to, jak tworzenie *histogramów* wartości wybranych pikseli oraz zastosowanie testu *chi-kwadrat* pozwalają na identyfikację anomalii sugerujących obecność *komunikatu*. Wyniki eksperymentalne pokazane przez autorów wykazują skuteczność tej metody, szczególnie w obrazach o małej liczbie kolorów, ale także wskazują na jej ograniczenia, np. wrażliwość na *szum*. Praca porównuje też metodę *chi-kwadrat* z innymi technikami *steganoanalizy*.

Technika *chi-kwadrat* jest jedną z bardziej popularnych metod wykrywania ukrytych *komunikatów* w obrazach cyfrowych, szczególnie w kontekście *steganografii* opartych na algorytmach wykorzystujących modyfikacje *LSB*. Technika ta polega na analizie *histogramów* wartości pikseli dla obrazu podejrzewanego o zawieranie ukrytego *komunikatu*, który następnie dzielony jest na dwa odrębne *histogramy*: jeden dla parzystych wartości pikseli i drugi dla nieparzystych wartości pikseli. Do porównania tych dwóch *histogramów* stosowany jest test *chi-kwadrat*, który mierzy to, jak bardzo rozkład wartości pikseli w potencjalnym *steganogramie* różni się od oczekiwanego rozkładu w oryginalnym *nośniku*. W przypadku obecności ukrytych danych, wartości *chi-kwadrat* odbiegają od oczekiwanych, wskazując na anomalię.

Kod 7 zawiera przykładowy algorytm implementujący technikę *steganoanalityczną chi-kwadrat*.

```
def CHI2_analysis(image_file_path):
    """
    Test chi-kwadrat histogramów parzystych i nieparzystych pikseli (R, G, B).

    Argumenty:
    image_file_path (string): ścieżka do pliku obrazu
```

[73] **Chi-kwadrat** - test statystyczny stosowany do oceny zależności między zmiennymi kategorycznymi umożliwiający porównanie rozkładu danych empirycznych z rozkładem teoretycznym, aby sprawdzić, czy istnieje istotna różnica między tymi rozkładami. W *steganoanalizie* wykorzystywany jest do porównywania oczekiwanego i obserwowanego rozkładu danych w *steganogramie*, który pomaga wykryć potencjalne anomalie mogące sugerować obecność zakodowanego *komunikatu (Abramowitz i Stegun, 1965)*.

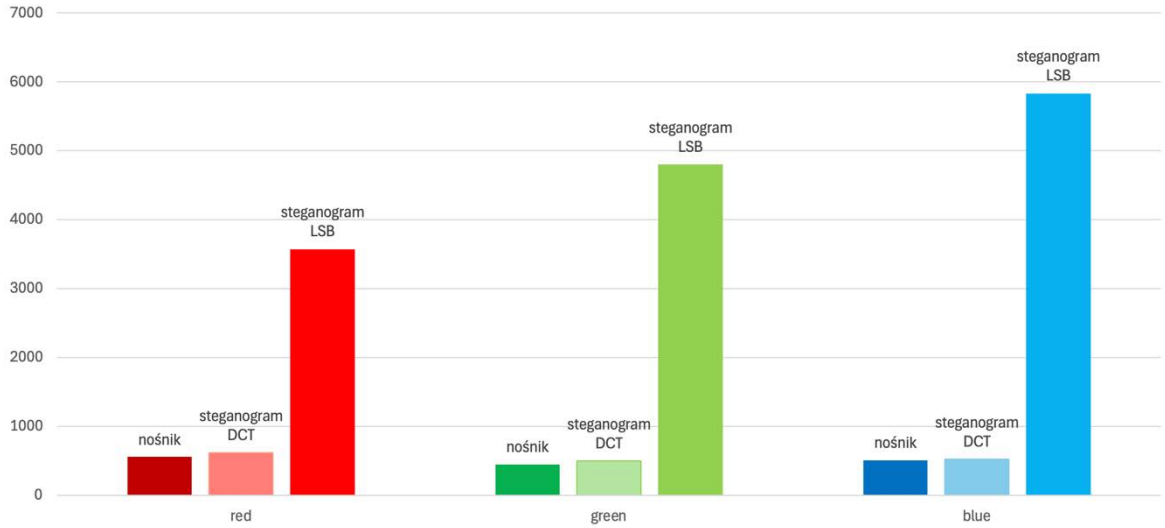
```
    Wyniki:
    np.array: wyniki testu chi-kwadrat dla kanałów R, G, B.
    """

    # Otwórz obraz i konwertuj na przestrzeń kolorów RGB
    image = Image.open(image_file_path)
    image = image.convert('RGB')

    # Inicjalizacja histogramów dla parzystych i nieparzystych pikseli
    hist_even = np.zeros((3, 256), dtype=int)  # Kanały: R, G, B
    hist_odd = np.zeros((3, 256), dtype=int)   # Kanały: R, G, B

    # Przechodzenie przez każdy piksel obrazu
    for i, pixel in enumerate(image.getdata()):
        red, green, blue = pixel
        if i % 2 == 0:  # Parzyste piksele
            hist_even[0, red] += 1
            hist_even[1, green] += 1
            hist_even[2, blue] += 1
        else:  # Nieparzyste piksele
            hist_odd[0, red] += 1
            hist_odd[1, green] += 1
            hist_odd[2, blue] += 1

    # Obliczenie chi-kwadrat
    chi_square_values = np.zeros(3)  # Wyniki dla kanałów R, G, B
    for channel in range(3):  # Dla każdego kanału (R, G, B)
        obs = hist_even[channel]
        exp = hist_odd[channel]
        nonzero_mask = exp > 0
        chi_square_values[channel] = np.sum(
            (obs[nonzero_mask]-exp[nonzero_mask])**2/exp[nonzero_mask] )

    return chi_square_values
```

Kod 7 - Algorytm obliczania wartości testu w metodzie *steganoanalitycznej chi-kwadrat*
Źródło: *Opracowanie własne*

Technika *chi-kwadrat* cechuje się prostotą i skutecznością, przez co jest stosunkowo łatwa do implementacji i może być bardzo efektywna w wykrywaniu *steganografii LSB*. Technika ta ma szerokie zastosowanie mogąc być używana do analizy różnych typów obrazów, zarówno kolorowych, jak i w skali szarości. Niemniej jednak, jej skuteczność może być ograniczona w obrazach zawierających dużo *szumu* lub losowych elementów, które mogą wpływać na rozkład wartości pikseli. Bardziej zaawansowane metody *steganografii*, które nie polegają na prostych modyfikacjach *LSB*, mogą być trudniejsze do wykrycia za pomocą tej techniki.

Przykład 8 - wykorzystanie metody steganoanalitycznej chi-kwadrat

Na rysunku 27 przedstawione są uzyskane wyniki zastosowania metody *steganoanalitycznej chi-kwadrat* na używanym wcześniej przykładowym *nośniku* i *steganogramach* utworzonych przy pomocy technik *steganograficznych LSB* oraz *DCT*.



Rysunek 27 - Przykład: wartości testu *chi-kwadrat* dla *nośnika* i *steganogramów LSB i DCT*
Źródło: *Opracowanie własne*

Uzyskane w metodzie *steganoanalitycznej chi-kwadrat* wartości testu dla poszczególnych *histogramów* kolorów są istotnie większe dla *steganogramu LSB* niż *nośnika,* jak też *steganogramu DCT*. Wskazuje to na anomalie występujące w *histogramie steganogramu LSB* i uprawdopodabnia zakodowanie w nim ukrytego *komunikatu*. W przypadku steganogramu *DCT,* wprawdzie uzyskane wartości testu *chi-kwadrat* są trochę większe niż w oryginalnym *nośniku*, to różnice nie są na tyle duże, aby istotnie uprawdopodobniało to zakodowanie w nim *komunikatu*. Pokazuje to, że metoda *steganoanalizy chi-kwadrat* dla *steganografii DCT* nie da w tym przypadku oczekiwanych rezultatów.

Koniec przykładu 8

2.3.2.3 Technika steganoanalityczna DCT Coefficient Analysis

Analiza współczynników *DCT (Pevný i Fridrich, Merging Markov and DCT features for multi-class JPEG steganalysis, 2007)* jest jedną z zaawansowanych technik wykorzystywanych w *steganoanalizie* do wykrywania ukrytych *komunikatów* w obrazach cyfrowych, zwłaszcza skompresowanych w formacie *JPEG*. Technika ta koncentruje się na analizie zmian w *domenie częstotliwości* obrazu, które są wprowadzane przez proces *steganografii*. Głównym celem jest identyfikacja anomalii w rozkładzie współczynników

DCT, które mogą wskazywać na obecność ukrytych *komunikatów*. W tym celu porównuje się np. *histogramy* współczynników *DCT* obrazu podejrzanego o zastosowanie *steganografii* z obrazem referencyjnym lub modelem teoretycznym *nośnika*, gdyż zakodowanie *komunikatu* powoduje powstanie w *steganogramie* odchyleń od standardowego modelu kompresji *JPEG*.

Kod 8 zawiera przykładowy algorytm obliczający *histogramy* dla wartości współczynników *transformaty* w metodzie *steganoanalitycznej* bazującej na analizie współczynników *DCT*.

```
def DCT_analysis(image_file_path):
    """
    Tworzy histogram współczynników DCT obrazu.

    Argumenty:
    image_file_path (string): ścieżka do pliku obrazu

    Wyniki:
    np.array: histogramy współczynników DCT
    """

    # Otwórz obraz i konwertuj na skalę szarości
    image = Image.open(image_file_path).convert('L')
    image_array = np.array(image)
    height, width = image_array.shape

    # Inicjalizacja histogramu DCT dla bloków 8x8
    dct_histogram = np.zeros((8, 8, 256))

    # Przechodzenie przez obraz blokami 8x8
    for i in range(0, height, 8):
        for j in range(0, width, 8):
            block = image_array[i:i + 8, j:j + 8]

            # Implementacja dwuwymiarowej transformacji DCT
            N = block.shape[0]
            dct_block = np.zeros_like(block, dtype=float)
            for u in range(N):
                for v in range(N):
                    sum_val = 0.0
                    for x in range(N):
                        for y in range(N):
                            sum_val += block[x, y] * np.cos(
                                np.pi * u * (2 * x + 1) / (2 * N)) * np.cos(
                                np.pi * v * (2 * y + 1) / (2 * N))
                    cu = np.sqrt(1 / N) if u == 0 else np.sqrt(2 / N)
                    cv = np.sqrt(1 / N) if v == 0 else np.sqrt(2 / N)
                    dct_block[u, v] = cu * cv * sum_val

            # Kwantyzacja wartości do przedziału 0-255 z ograniczeniem
            quantized_dct_block = np.clip(
                np.round(dct_block).astype(int) + 128, 0, 255 )
```

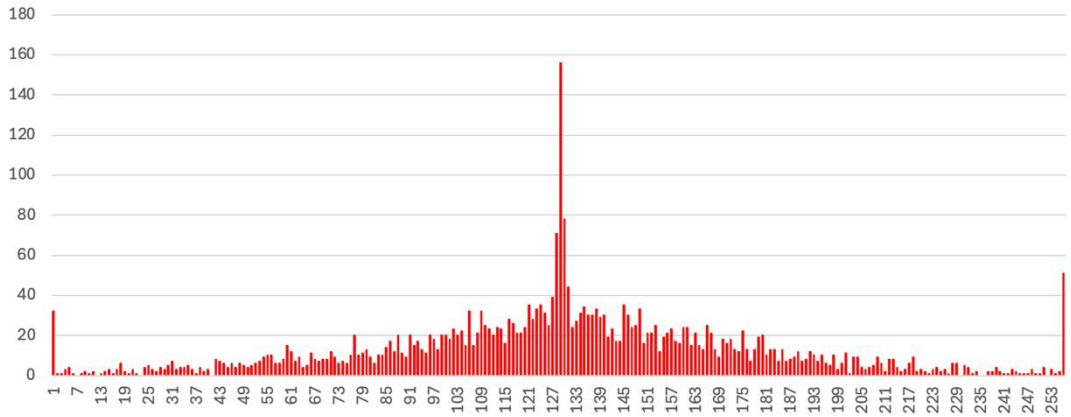
```
            # Zliczanie wystąpień każdej wartości współczynnika DCT
            for x in range(8):
                for y in range(8):
                    value = quantized_dct_block[x, y]
                    dct_histogram[x, y, value] += 1

    return dct_histogram
```

Kod 8 - Algorytm obliczania *histogramów* w *steganoanalitycznej* metodzie analizy współczynników *DCT*
Źródło: *Opracowanie własne*

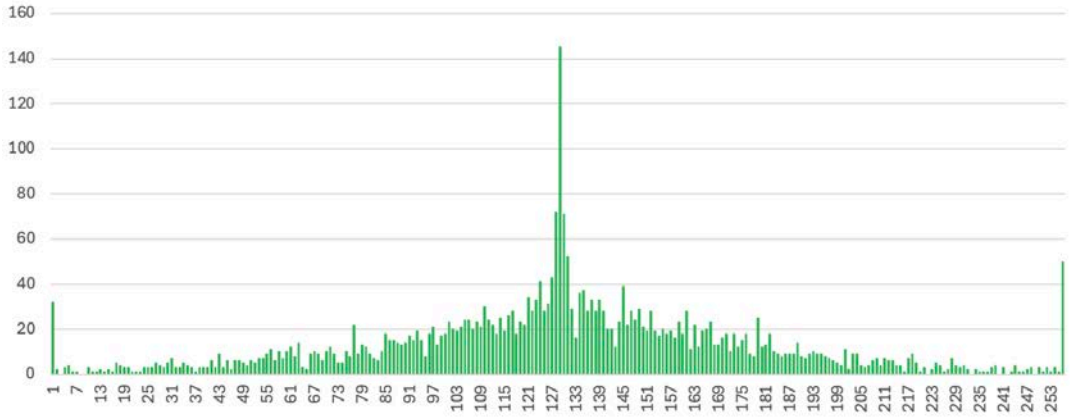
Analiza współczynników *DCT* charakteryzuje się dużą skutecznością szczególnie w wykrywaniu *steganografii* w formacie *JPEG*, który jest powszechnie używany do kompresji obrazów. Analiza współczynników *DCT* pozwala na dokładne wykrywanie nawet niewielkich zmian w obrazie, które są wynikiem kodowania *komunikatów*. Złożoność obliczeniowa tej metody jest relatywnie duża, co może być wadą w przypadku analizy dużych zbiorów danych. Jednocześnie metoda ta jest przydatna głównie dla obrazów kompresowanych przy użyciu technik *DCT*, co ogranicza jej stosowalność. Pomimo tych ograniczeń analiza współczynników *DCT* pozostaje zaawansowaną i skuteczną techniką w dziedzinie *steganoanalizy*.

Przykład 9 - histogramy współczynników DC dla nośnika i steganogramów

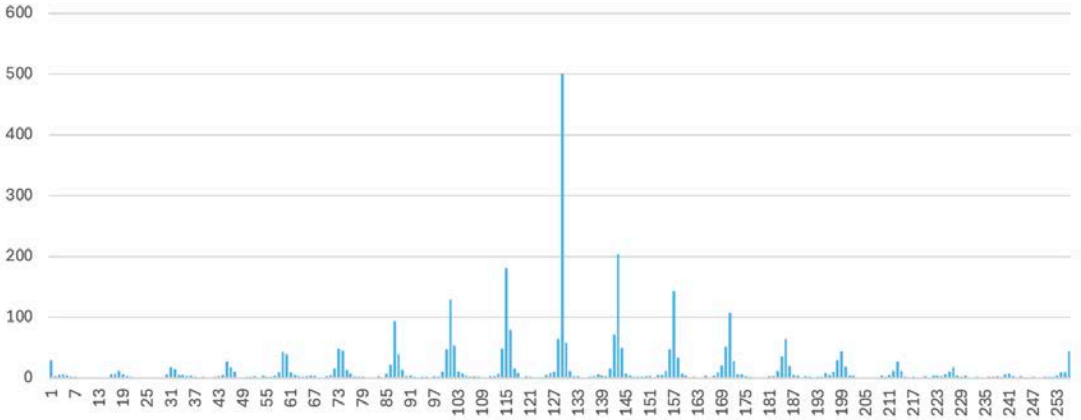
Poniżej przedstawione są *histogramy współczynników DCT* dla *nośnika*- rysunek 28, *steganogramu* utworzonego przy pomocy techniki *steganografii LSB* - rysunek 29 *oraz steganogramu* utworzonego przy pomocy techniki *steganografii DCT* - rysunek 30.



Rysunek 28 - Przykład: *histogram* współczynników *DCT* dla *nośnika*
Źródło: *Opracowanie własne*



Rysunek 29 - Przykład: *histogram* współczynników *DCT* dla *steganogramu* zakodowanego metodą *LSB*
Źródło: *Opracowanie własne*



Rysunek 30 - Przykład: *histogram* współczynników *DCT* dla *steganogramu* zakodowanego metodą *DCT*
Źródło: *Opracowanie własne*

Na pierwszy rzut oka nie widać istotnych różnic pomiędzy *histogramem* dla oryginalnego *nośnika* a *histogramem* dla *steganogramu LSB*. Obydwa *histogramy* są regularne i nie dają powodów do przypuszczeń, że zakodowany jest w nich *komunikat.* Pokazuje to, że metoda analizy współczynników *DCT* nie daje w analizowanym przykładzie rozstrzygnięcia w przypadku *steganografii przestrzennej*. Natomiast *histogram* dla *steganogramu* zakodowanego *steganografią transformatową DCT* wyraźnie pokazuje występowanie anomalii i wskazuje na ukrycie w nim *komunikatu*.

Koniec przykładu 9

Ciekawym rozwinięciem metody analizy współczynników *DCT* jest zaproponowane w artykule *"A Markov Process Based Approach to Effective Attacking JPEG Steganography"* autorstwa Y. Shi'ego, C. Chena i W. Chena z 2006 roku *(Shi, Chen i Chen, 2006)* wykorzystanie *procesów Markowa* do modelowania statystycznych właściwości *kwantowanych* współczynników *transformaty DCT*, które są wykorzystywane w *kodekach JPEG*. Metoda ta skupia się na analizie dwuwymiarowych tablic utworzonych z wartości tych współczynników *DCT*. Poprzez tworzenie różnicowych dwuwymiarowych tablic *JPEG* w różnych kierunkach (poziomym, pionowym i diagonalnym) metoda ta zwiększa wykrywalność subtelnych zmian wprowadzonych przez zastosowanie technik *steganograficznych*. Wyprowadzone z tych tablic macierze prawdopodobieństw przejść są następnie używane do uchwycenia statystyk drugiego rzędu, których analiza może prowadzić do identyfikacji odchyleń w danych, a w konsekwencji do uprawdopodobnienia faktu zakodowania w *steganogramie komunikatu*. Technika *steganoanalizy* opisana w artykule wykazuje według autorów większą skuteczność w porównaniu do innych metod w wykrywaniu *steganografii obrazowej* w plikach *JPEG* zarówno pod względem dokładności wykrywania, jak też zmniejszenia rozmiaru przestrzeni badanych cech, co zwiększa jej wydajność obliczeniową.

Podobne podejście do rozwinięcia metody analizy współczynników *DCT* omówione zostało przez T. Pevny'ego i J. Fridrich w 2007 roku w artykule *"Merging Markov and DCT features for multi-class JPEG steganalysis" (Pevný i Fridrich, Merging Markov and DCT features for multi-class JPEG steganalysis, 2007)*. Zaproponowali oni technikę *steganoanalityczną* dedykowaną dla obrazów *JPEG* wykorzystująca analizę współczynników *DCT* w połączeniu z *procesami Markowa* do definiowania statystycznych zależności między pikselami oraz współczynników *DCT*. Przeprowadzone przez autorów eksperymenty wykazują, że proponowana metoda znacząco poprawia dokładność w porównaniu do metod opartych na pojedynczych typach cech.

2.3.2.4 Techniki steganoanalizy wykorzystujące uczenie maszynowe

Badania nad wykrywaniem ukrytych danych w *nośnikach* cyfrowych z wykorzystaniem mechanizmów *uczenia maszynowego,* w tym głębokich *sieci neuronowych*, pokazują dużą skuteczność oraz zdolność do adaptacji wobec dynamicznie rozwijających się technik *steganograficznych*. Techniki *steganoanalizy* wykorzystujące głębokie uczenie *sieci neuronowych* wykazują dużą skuteczność i czułość w rozpoznawaniu subtelnych wzorców wskazujących na zastosowanie *steganografii*. Wnioski z wielu badań

sugerują, że głębokie *sieci neuronowe* mają znaczny potencjał w zakresie *steganoanalizy* i mogą przewyższać tradycyjne metody deterministyczne, szczególnie w przypadku wykorzystania zaawansowanych technik *steganograficznych*.

W artykule W. Saleha, H. Alqahtani'ego i S. Saleha *"Digital Image Steganalysis: Current Methodologies and Future Challenges"* opublikowanego w 2022 roku *(Eid, Alotaibi, Alqahtani i Saleh, 2022)* dokonano przeglądu obecnie stosowanych metod *steganoanalitycznych* dotyczących wykrywania zastosowania *steganografii obrazowej* oraz przedstawiono przewidywane przyszłe wyzwania związane z tą dziedziną. Autorzy omawiają zastosowanie zaawansowanych technik *uczenia maszynowego*, w tym *CNN*. W artykule poruszono również problemy związane z wykrywaniem *steganogramów* o małej *pojemności* oraz optymalizacją procesów uczenia *sieci neuronowych* wykorzystywanych w metodach *steganoanalitycznych*.

Z kolei w opracowaniu *"Deep learning for steganalysis of diverse data types: A review of methods, taxonomy, challenges and future directions"* opublikowanym w 2024 roku przez H. Kheddara i innych *(Kheddar, Hemis, Himeur, Megías i Amira, 2024)* przedstawiono wszechstronny przegląd technik *steganoanalizy* opartych na *uczeniu maszynowym*. W pracy omówiono metody wykrywania zakodowanych *komunikatów* w różnych rodzajach mediach cyfrowych takich jak obrazy, *audio* i *wideo*, podkreślając znaczenie w procesie *steganoanalizy* zaawansowanych technik *uczenia maszynowego* takich jak np. DTL (Deep Transfer Learning) - technika uczenia maszynowego, która przenosi wiedzę w *sieciach neuronowych* pomiędzy zadaniami zmniejszając zapotrzebowanie na zasoby obliczeniowe oraz DRL (Deep Reinforcement Learning) - technika łącząca głębokie uczenie (deep learning) z uczeniem przez wzmacnianie (reinforcement learning), gdzie *sieci neuronowe* uczy się optymalnych strategii poprzez odpowiednio skonstruowany system nagród i kar. Autorzy przeanalizowali najnowsze badania w tej dziedzinie, w tym zestawy danych i metryki oceny stosowane w badaniach. W artykule przedstawiono także ocenę wydajności omawianych technik na przykładach różnych zestawów danych. Przegląd kończy się omówieniem obecnego stanu *steganoanalizy* opartej na *uczeniu maszynowym* oraz potencjalnych wyzwań i możliwych przyszłych kierunków badań.

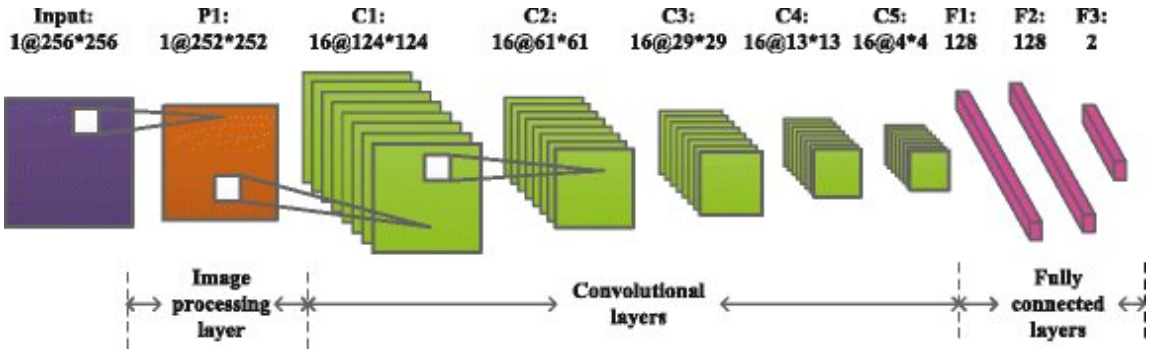
Jednymi z najczęściej stosowanych metod *steganoanalitycznych* opartych na głębokim *uczeniu maszynowym* są techniki wykorzystujące konwolucyjne *sieci neuronowe CNN* i ich duże zdolności do automatycznej ekstrakcji cech z obrazów. Sieci te są dobrze znane ze swojej skuteczności w zadaniach związanych z przetwarzaniem obrazów, co czyni je odpowiednim narzędziem do detekcji ukrytych *komunikatów* w *steganogramach*

obrazowych. Model *CNN* jest trenowany na dużym zbiorze danych zawierającym zarówno obrazy z ukrytymi *komunikatami*, jak i obrazy bez zakodowanych *komunikatów*, co pozwala sieci nauczyć się rozpoznawania różnic wskazujących na fakt zastosowania technik *steganograficznych*. Przykładem metody *steganoanalitycznej* wykorzystującej sieci *CNN* jest technika zaproponowana w artykule *"Feature learning for steganalysis using convolutional neural networks"* przez Y. Qiana, J. Donga i W. Wanga w 2018 roku *(Qian, Dong, Wang i Tan, 2018)*. Wykorzystuje ona sieci *CNN* do automatycznej ekstrakcji cech z obrazów. Wyniki eksperymentów pokazują skuteczność proponowanego modelu *CNN*, który przewyższa metody deterministyczne, zwłaszcza w przypadku zaawansowanych technik *steganograficznych,* wykazując dużą precyzję i czułość w rozpoznawaniu wzorców charakterystycznych dla *steganografii*.

Architektura wykorzystywanych sieci *CNN* sieci składa się m.in. z kilku warstw konwolucyjnych, z których każda wykrywa różnorodne wzorce i cechy. Sieć jest optymalizowana za pomocą funkcji straty, która penalizuje błędne klasyfikacje, umożliwiając poprawę detekcji w miarę postępu procesu treningowego. Proces treningu obejmuje zgromadzenie dużego zbioru *steganogramów* oraz *nośników*. Dane te są starannie oznaczone, aby model mógł się uczyć rozpoznawania wzorców charakterystycznych dla zastosowania *steganografii*. W trakcie treningu stosowane są techniki *augmentacji danych*[74], aby zwiększyć różnorodność danych treningowych i poprawić ogólną zdolność generalizacji modelu. Walidacja modelu odbywa się na niezależnym zbiorze danych, co pozwala ocenić jego skuteczność i zapewnić, że nie jest *przeuczony*[75]. Na rysunku 31 przedstawiono model sieci *CNN* składający się z warstw neuronów realizujących różne funkcje w procesie *steganoanalizy*.

[74] **Augmentacja danych** - technika polegająca na sztucznym zwiększaniu zbioru danych treningowych poprzez stosowanie transformacji, takich jak obrót, skalowanie, przesunięcie lub zmiana jasności, w celu zwiększenia różnorodności danych. Wykorzystywana jest głównie w *uczeniu maszynowym*, pozwala na poprawę generalizacji modeli oraz redukcję ryzyka *przeuczenia sieci neuronowej*, szczególnie w sytuacjach, gdzie dostępność danych jest ograniczona *(Dempster, Laird i Rubin, 1977)*.

[75] **Przeuczenie sieci neuronowej** - zjawisko, w którym *sieć neuronowa* uczy się zbyt dokładnie na danych treningowych, tracąc zdolność do generalizacji i skutecznego działania na nowych, nieznanych danych, co prowadzi do spadku jej efektywności *(Burnham i Anderson, 2004)*.



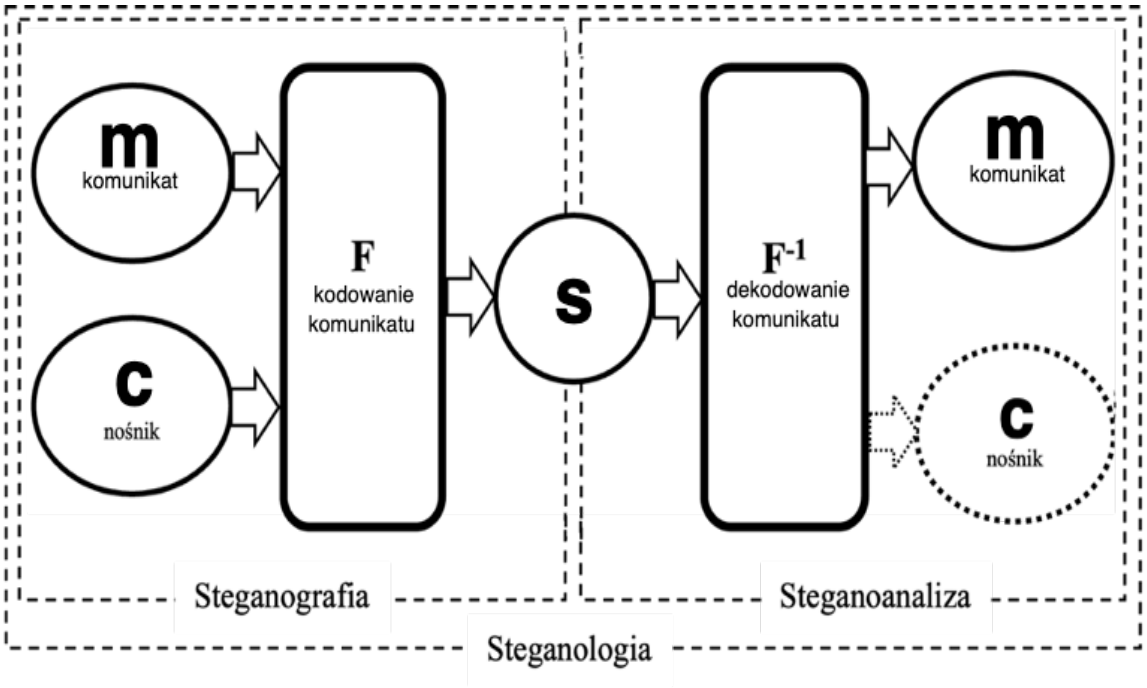
Rysunek 31 - Przykład: model sieci *steganoanalitycznej CNN* składający się z warstw: 1 przetwarzania obrazu, 5 splotowych i 3 w pełni połączonych. Źródło: *Qian, Y.; Dong, J.; Wang, W.; Tan, T., 2018, Feature learning for steganalysis using convolutional neural networks. Multimedia Tools, DOI 77. 10.1007/s11042-017-5326-1*

Ciekawe podejście do wykorzystania *uczenia maszynowego* w steganoanalizie zaprezentowała J. Fridrich i J. Kodovsky w artykule *"Rich Models for Steganalysis of Digital Images"* z 2012 roku *(Fridrich i Kodovsky, Rich models for steganalysis of digital images, 2012)*. Opisują oni nową propozycję metody *steganoanalitycznej* wykorzystującej budowanie detektorów *steganoanalitycznych* dla obrazów cyfrowych wprowadzając pojęcie bogatego modelu, który składa się z wielu zróżnicowanych podmodeli, bazujących na rozkładach sąsiednich próbek zredukowanych *szumów* obrazu uzyskanych za pomocą *liniowych i nieliniowych filtrów wysokoprzepustowych*[76]. Podejście to pozwala na efektywniejsze wykrywanie zakodowanego *komunikatu* w obrazach poprzez uwzględnienie statystycznych zależności zarówno w *domenie przestrzennej*, jak też w *domenie częstotliwościowej*.

[76] **Liniowe i nieliniowe filtry wysokoprzepustowe** - filtry stosowane do wykrywania ukrytych *informacji* przez analizę wysokoczęstotliwościowych komponentów obrazu. Przykładem filtra liniowego jest filtr Butterwortha, a nieliniowego filtr medianowy *(Antoniou, 1993)*.

2.4 Podstawy steganologii

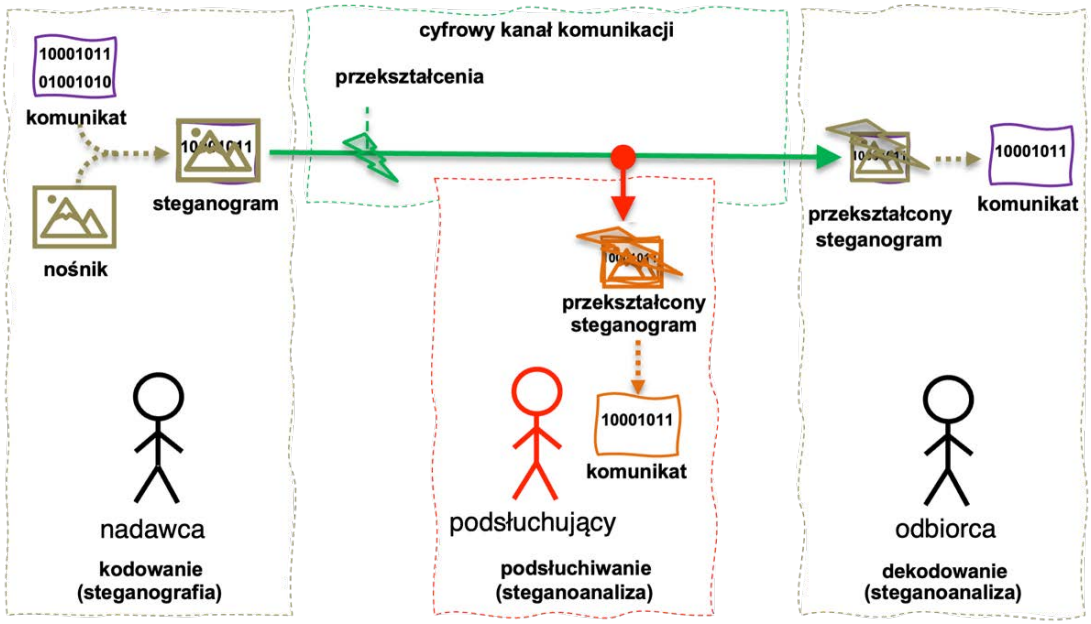
Steganologia łączy *steganografię* i *steganoanalizę* w jedną gałąź nauki związaną z rozwojem metod umożliwiających z jednej strony skuteczne ukrywanie *komunikatów*, a z drugiej doskonalenie technik ich wykrywania. Punktem łączącym oba podejścia jest *steganogram*, który dla *steganografii* jest produktem końcowym, a dla *steganoanalizy* punktem początkowym procesu. Schemat zależności pomiędzy *steganografią*, *steganoanalizą* i *steganologią* przedstawiono na rysunku 32.



Rysunek 32 - *Steganologia = Steganografia + Steganoanaliza*
Źródło: *(Pery, Zastosowanie steganologii w cyberbezpieczeństwie, 2024)*

W kontekście *teorii informacji* Shannona *steganografia* zwiększa *entropię* danych w celu ukrycia tajnego *komunikatu* w *szumie* informacyjnym, natomiast *steganoanaliza* zmniejsza *entropię* danych wydobywając ten *komunikat* z zakodowanych nadmiarowych *informacji*.

Podstawowym scenariuszem steganologicznym[77], przedstawionym na rysunku 33, jest utworzenie przez *nadawcę steganogramu* w postaci połączenia *nośnika* z tajnym *komunikatem* i wysłanie go *kanałem cyfrowym*[78] do *odbiorcy*. *Odbiorca* po otrzymaniu *steganogramu* od *nadawcy* powinien móc zdekodować ze *steganogramu* zakodowany *komunikat*.



Rysunek 33 - Schemat *podstawowego scenariusza steganologicznego*
Źródło: *Opracowanie własne*

W przypadku trywialnym *nadawca* poprzez *kanał cyfrowy* przekaże *odbiorcy* niepodsłuchany przez nikogo i w żaden sposób nie przekształcony *steganogram*, a *odbiorca* za pomocą funkcji odwrotnej do funkcji *steganograficznej* zastosowanej przez *nadawcę* odczyta przeznaczony dla niego poufny *komunikat*. W tym przypadku trzeba jednoznacznie zdefiniować dla *nadawcy* funkcję *steganograficzną* F oraz należy zadbać o to, aby *odbiorca* znał odwrotne jednoznaczne przekształcenie F^{-1}.

---

[77] **Podstawowy scenariusz steganologiczny** - scenariusz przesyłania pomiędzy *nadawcą* a *odbiorcą steganogramu* w postaci pliku cyfrowego poprzez nieprzerwany, ciągły *kanał cyfrowy*.

[78] **Kanał cyfrowy** - medium do przesyłania pomiędzy *nadawcą* a *odbiorcą steganogramu* w postaci pliku cyfrowego, którego zadaniem jest efektywny, niezawodny i bezpieczny transfer cyfrowych danych, minimalizując potencjalne zakłócenia i straty *informacji*.

Problem staje się nietrywialny wtedy, gdy *nadawca* chce skutecznie i w sposób tajny przekazać *komunikat odbiorcy*, ale jednocześnie zakłada się, że w trakcie przesyłania *steganogramu* mogą się zdarzyć dwie rzeczy:

1. *steganogram* może ulec przekształceniu lub zakłóceniu,
2. *podsłuchujący*[79] może podsłuchać komunikację pomiędzy *nadawcą* i *odbiorcą* oraz może chcieć dowiedzieć się, czy oprócz oficjalnej komunikacji *nadawca* przekazuje *odbiorcy* jeszcze jakiś dodatkowy *komunikat*, a jeżeli tak, to jaka jest jego treść.

Zaadresowanie kwestii przekształcenia lub zakłócenia *steganogramu* oznacza konieczność zapewnienia po stronie *nadawcy* takich technik *steganograficznych*, które w sposób *odporny* na przewidywane sposoby zniekształceń lub wykorzystując mechanizmy *redundancji* będą zapewniały, że *odbiorca* będzie w stanie odczytać *komunikat* z *przekształconego steganogramu*[80].

Natomiast kwestia ewentualnego podsłuchu jest sprawą fundamentalną w kontekście głównego założenia *steganografii*, jakim jest zapewnienie, że obecność ukrytego *komunikatu* pozostaje niewidoczna dla jakichkolwiek osób nieupoważnionych. W *steganologii* zakłada się apriori, że *podsłuchujący* może podsłuchiwać *steganogram* wysyłany od *nadawcy* do *odbiorcy*. Zadaniem *nadawcy* jest więc takie ukrycie *komunikatu* w *steganogramie*, aby *podsłuchujący* nie był w stanie odkryć faktu, iż oprócz oczywistej i oficjalnej *informacji* zawartej w nośniku, w *steganogramie* jest ukryty jeszcze inny tajny *komunikat* przeznaczony do *odbiorcy*. Użyte przez *nadawcę* techniki *steganograficzne* muszą więc być nieoczywiste i na swój sposób wyrafinowane tak, aby zmylić *podsłuchującego*. Z kolei *podsłuchujący* starając się przechytrzyć *nadawcę*, musi starać się odkrywać coraz to nowe techniki *steganoanalizy*, aby nie zostać zaskoczony przez *nadawcę* i móc odkryć zakodowany *komunikat*.

Ciągła rywalizacja pomiędzy *podsłuchującym* i *nadawcą* może być uogólniona do problemu *"miecza i tarczy"*, czyli wynajdowania z jednej strony coraz to skuteczniejszych

---

[79] **Podsłuchujący** - osoba postronna, która podsłuchuje transmisję pomiędzy *nadawcą* i *odbiorcą*, a która nie powinna dowiedzieć się, że oprócz oficjalnych *informacji nadawca* przekazuje *odbiorcy komunikat* zakodowany w *steganogramie*.

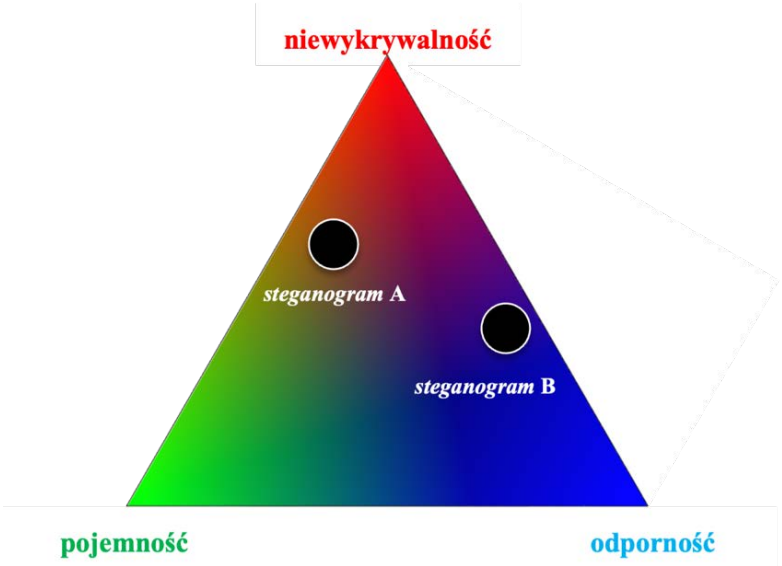
[80] **Przekształcony steganogram** - otrzymany w *podstawowym scenariuszu steganologicznym* przez *odbiorcę* przekształcony lub zniekształcony plik cyfrowy wysłany pierwotnie przez *nadawcę* jako *oryginalny steganogram*.

form ataku, a z drugiej odkrywania coraz to lepszych form obrony przed tymi atakami. Ten stały proces obecny jest w życiu człowieka od zarania dziejów *(Freedman, 2013)*, a w tym konkretnym przypadku, którego dotyczy niniejsza praca, stanowi jedną z przyczyn ciągłego rozwoju dziedziny nauki jaką jest *steganologia*.

2.5 Miary używane w steganologii

W *steganologii* stosuje się różne miary do oceny jakości ukrywania *komunikatu* i wynikowego *steganogramu* oraz możliwości wykrycia faktu zakodowania *komunikatu* w *nośniku*. Najważniejsze z tych miar to: *niewykrywalność*, *pojemność* oraz *odporność*.

W przypadkach praktycznych nie da się zoptymalizować wszystkich miar naraz, zwykle powiększanie wartości jednej miary wiąże się ze zmniejszaniem innej. Na schemacie przedstawionym na rysunku 34 pokazany jest model współzależności pomiędzy optymalizacją poszczególnych miar *steganografii* z przykładami dwóch hipotetycznych *steganogramów* A i B. *Steganogram* A jest zoptymalizowany głównie pod kątem *niewykrywalności* oraz trochę pod kątem *pojemności* kosztem *odporności*, natomiast *steganogram* B jest zoptymalizowany głównie pod kątem *odporności* oraz *niewykrywalności* kosztem *pojemności*.



Rysunek 34 - Schemat zależności pomiędzy miarami *steganogramów*
Źródło: *Opracowanie własne* na podstawie *(Katzenbeisser i Petitcolas, 2000; Huynh-Thu i Ghanbari, 2008; Wang, Bovik, Sheikh i Simoncelli, 2004; Shannon, 1948)*

Dobór odpowiedniej techniki *steganograficznej* oraz jej poszczególnych parametrów wpływający na ostateczną wartość metryk poszczególnych miar powinien zależeć od przyjętych założeń i przypisanych tym miarom priorytetów.

2.5.1 Miara niewykrywalności metod steganograficznych

Niewykrywalność[81] to miara jakości techniki *steganograficznej* oraz samego *steganogramu* po zakodowaniu w nim ukrytego *komunikatu*. *Niewykrywalność* jest tym większa, im *steganogram* jest bliższy oryginalnemu *nośnikowi*, a zakodowany *komunikat* jest trudniej wykrywalny przez analizy *steganoanalityczne*.

Najprostszym wskaźnikiem używanym do określenia *niewykrywalności* poprzez ocenę podobieństwa *steganogramu* do *nośnika* jest *MSE*[82] *(*błąd średniokwadratowy), zdefiniowany wzorem (4).

$$MSE = \frac{1}{N * M} \sum_{i=1}^{N} \sum_{j=1}^{M} ([f(i,j) - f'(i,j)]^2) \tag{4}$$

gdzie: N, M- wymiary obrazu,
$f(i,j)$ - wartość piksela (i,j) w *nośniku*,
$f'(i,j)$ - wartość piksela (i,j) w *steganogramie*.

Kod 9 zawiera prosty algorytm obliczający błąd średniokwadratowy pomiędzy dwoma obrazami.

```
def MSE(image1, image2):
    """
    Oblicza błąd średniokwadratowy (MSE) między dwoma obrazami.

    Argumenty:
```

[81] **Niewykrywalność (Undetectability)** - miara zdolności techniki *steganograficznej* do ukrycia *komunikatu* w *steganogramie* w sposób, który uniemożliwia wykrycie jego obecności zarówno subiektywnymi testami percepcji ludzkich zmysłów, jak też obiektywnymi metodami matematycznymi. Duża *niewykrywalność* oznacza, że zmiany w *nośniku* wprowadzone przez techniki *steganograficzne* są mniej zauważalne oraz trudniejsze do wykrycia *(Katzenbeisser i Petitcolas, 2000; Huynh-Thu i Ghanbari, 2008; Wang, Bovik, Sheikh i Simoncelli, 2004; Shannon, 1948)*.

[82] **MSE (Mean Squared Error)** - metryka oceniająca podobieństwo *steganogramu* do *nośnika* zdefiniowana jako średnia kwadratów różnic między nimi. Niższe wartości *MSE* wskazują na większe podobieństwo *steganogramu* do *nośnika*, co jest skorelowane z większą *niewykrywalnością*. *(Cheddad, Condell, Curran i Kevitt, 2010)*.

```
    image1 (np.array): pierwszy obraz, np. nośnik
    image2 (np.array): drugi obraz, np. steganogram

    Wyniki:
    float: wartość błędu średniokwadratowego (MSE)
    """

    # Konwersja obrazów do typu float64
    image1 = image1.astype(np.float64)
    image2 = image2.astype(np.float64)

    # Obliczenie różnicy między obrazami i podniesienie do kwadratu
    diff = (image1 - image2) ** 2

    # Obliczenie średniej wartości różnicy
    mse = np.mean(diff)

    return mse
```

Kod 9 - Algorytm obliczania wskaźnika *MSE* - błędu średniokwadratowego
Źródło: *Opracowanie własne*

Przykład 10 - wartości wskaźnika MSE dla steganogramów

Przykładowe wartości wskaźnika *MSE* dla par oryginalnego *nośnika* oraz *steganogramów* zakodowanych różnymi technikami *steganograficznymi* zaprezentowane są w tabeli 2.

Tabela 2 - Przykład: wartości *MSE* dla różnych technik *steganograficznych*
Źródło: *Opracowanie własne*

Użyta technika steganograficzna	***Wartość MSE***
LSB	0,50
Patchwork	0,08
PVD	0,01
DCT	265,12

W powyższym przykładzie najmniejsze wartości wskaźnika *MSE* zostały uzyskane dla *steganogramów* uzyskanych przy pomocy technik *PVD* oraz *Patchwork*. Natomiast *steganogram* powstały w wyniku zastosowania techniki *DCT* wykazuje znacząco większą wartość wskaźnika *MSE*, co wskazuje w tym przypadku na istotnie większą różnicę pomiędzy *nośnikiem* a *steganogramem*.

Koniec przykładu 10

Jednym z najczęściej stosowanych wskaźników do oceny *niewykrywalności* jest *PSNR*[83], czyli szczytowy stosunek sygnału do *szumu*. Oryginalnie wskaźnik ten był używany do oceny jakości kompresji plików cyfrowych, ale z powodzeniem został zaadoptowany do oceny podobieństwa *steganogramu* do *nośnika* w przypadku technik *steganograficznych*. Wskaźnik *PSNR* wyrażany jest w decybelach i zdefiniowany wzorem (5).

$$PSNR = 10 * log_{10} \frac{\left[max\left(f(i,j)\right)\right]^2}{MSE} \tag{5}$$

gdzie: $max\left(f(i,j)\right)$- wartość maksymalna sygnału, np. 255.

Im większa wartość *PSNR*, tym mniejsze zniekształcenia *steganogramu* i większe jego podobieństwo do oryginalnego *nośnika*, a tym samym większa jego *niewykrywalność*.

Kod 10 zawiera prosty algorytm obliczający wskaźnik *PSNR* dla dwóch obrazów.

```
def PSNR(image1, image2):
    """
    Oblicza stosunek sygnału do szumu (PSNR) między dwoma obrazami.

    Argumenty:
    image1 (np.array): pierwszy obraz, np. nośnik
    image2 (np.array): drugi obraz, np. steganogram
    Wyniki:
    float: wartość PSNR
    """

    mse = MSE(image1, image2)
    if mse == 0:
        return 100  # Gdy obrazy są identyczne, PSNR jest maksymalne
    max_pixel = 255.0
    return 20 * log10(max_pixel / sqrt(mse))
```

Kod 10 - Algorytm obliczania wskaźnika *PSNR* (szczytowego stosunku sygnału do *szumu)*
Źródło: *Opracowanie własne*

[83] **PSNR (Peak Signal-to-Noise Ratio)** - metryka oceniająca podobieństwo *steganogramu* do *nośnika* (szczytowy stosunek sygnału do szumu). Im większe wartości *PSNR,* tym mniejsze są zniekształcenia *nośnika* i większe podobieństwo *steganogramu* do *nośnika*, co oznacza większą *niewykrywalność*, czyli skuteczniejsze ukrycie *komunikatu (Katzenbeisser i Petitcolas, 2000; Huynh-Thu i Ghanbari, 2008)*.

Przykład 11 - wartości wskaźnika PSNR dla steganogramów

Przykładowe wartości wskaźnika *PSNR* dla par oryginalnego *nośnika* oraz *steganogramów* zakodowanych różnymi technikami *steganograficznymi* zaprezentowane są w tabeli 3.

Tabela 3 - Przykład: wartości *PSNR* dla różnych technik *steganograficznych*
Źródło: *Opracowanie własne*

Użyta technika steganograficzna	***Wartość PSNR***
LSB	51,16
Patchwork	59,12
PVD	69,24
DCT	23,90

W powyższym przykładzie wskaźnik *PSNR* jest największy dla *steganogramu* uzyskanego techniką *PVD*. Wskazuje to, iż według tego wskaźnika to *steganogram PVD* jest najbliższy oryginalnemu *nośnikowi*. Najniższa wartość *PSNR* w przypadku zastosowania techniki *DCT* wskazuje, iż *steganogram DCT* najbardziej odbiega od oryginalnego nośnika.

Koniec przykładu 11

W artykule *"Scope of validity of PSNR in image/video quality assessment"* autorstwa Q. Huynh-Thu'a i M. Ghanbari'ego opublikowanym w 2008 roku *(Huynh-Thu i Ghanbari, 2008)* przedstawiona jest analiza przydatności i ograniczeń metryki *PSNR* jako miary *niewykrywalności* obrazu i *wideo*. Autorzy podkreślają, że mimo popularności *PSNR* ze względu na jej prostotę, nie zawsze dokładnie koreluje ona z subiektywnymi ocenami ludzkiej percepcji podobieństwa. Metryka *PSNR* jest bardziej przydatna do wstępnej analizy i w zaawansowanych ocenach nie powinna być traktowana jako jedyna miara *niewykrywalności steganogramu*. W artykule zasugerowano również, że alternatywne metryki, takie jak *SSIM*[84], mogą lepiej odzwierciedlać percepcyjne podobieństwo obrazów

[84] **SSIM (Structural Similarity Index)** - metryka używana do oceny *niewykrywalności* poprzez określanie podobieństwa strukturalnego *steganogramu* do oryginalnego *nośnika* uwzględniającego jasność, kontrast i strukturę w sposób bardziej zbliżony do percepcji ludzkich zmysłów *(Wang, Bovik, Sheikh i Simoncelli, 2004)*.

odbierane przez ludzi, co czyni ją w pewnych zastosowaniach bardziej odpowiednią do całościowej oceny *niewykrywalności steganogramów obrazowych* i *steganogramów wideo*.

Alternatywnym wskaźnikiem do oceny *niewykrywalności* jest *SSIM,* czyli podobieństwo strukturalne pomiędzy oryginalnym *nośnikiem* a *steganogramem*. Został on zaproponowany przez Z. Wanga, A. Bovika, H. Sheikha i E. Simoncellego w 2004 roku i opublikowany w artykule *"Image Quality Assessment: From Error Visibility to Structural Similarity" (Wang, Bovik, Sheikh i Simoncelli, 2004)*. Wskaźnik ten ocenia podobieństwo *steganogramu* do *nośnika* uwzględniając percepcyjne aspekty wizualne, takie jak jasność, kontrast i struktura, co stanowi bardziej dokładne odwzorowanie ludzkiego postrzegania wizualnego w porównaniu np. do metryki *PSNR*. Artykuł zawiera zarówno teoretyczne podstawy, jak też wyniki przeprowadzonych eksperymentów potwierdzających skuteczność *SSIM*.

Wskaźnik *SSIM* pełni analogiczną rolę jak *PSNR*, ale jest bardziej zbliżony do ludzkiego postrzegania różnic pomiędzy obrazami. *SSIM* uwzględnia trzy komponenty porównywanych obrazów: jasność, kontrast i strukturę i jest zdefiniowany wzorem (6).

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x{}^2 + \mu_y{}^2 + C_1)(\sigma_x{}^2 + \sigma_y{}^2 + C_2)} \tag{6}$$

gdzie: μ_x *i* μ_y - średnie wartości jasności pikseli,
$\sigma_x{}^2$ i $\sigma_y{}^2$ - wariancje poszczególnych zmiennych x i y,
σ_{xy} - kowariancja zmiennych x i y,
C_1 *i* C_2 - stałe normalizujące zależne od nośnika.

Im większa wartość *SSIM*, tym *steganogram* jest bliższy oryginalnemu *nośnikowi* i większa jest wartość jego *niewykrywalności*.

Kod 11 zawiera prosty algorytm obliczający wskaźnik *SSIM* dla dwóch obrazów.

```
def SSIM(image1, image2):
    """
    Oblicza wskaźnik podobieństwa strukturalnego (SSIM) między dwoma obrazami.
    Argumenty:
    image1 (np.array): pierwszy obraz, np. nośnik
    image2 (np.array): drugi obraz, np. steganogram
    Wyniki:
    float: wartość wskaźnika podobieństwa strukturalnego (SSIM)
    """
    # Stałe używane w obliczeniach SSIM
    C1 = (0.01 * 255) ** 2
    C2 = (0.03 * 255) ** 2

    image1 = image1.astype(np.float64)
    image2 = image2.astype(np.float64)
```

```
    # Tworzenie okna Gaussowskiego do wygładzania obrazów
    window = np.outer(
        cv2.getGaussianKernel(11, 1.5), cv2.getGaussianKernel(11, 1.5).T )

    # Obliczenie średnich kroczących obrazów
    mu1 = cv2.filter2D(image1, -1, window)[5:-5, 5:-5]
    mu2 = cv2.filter2D(image2, -1, window)[5:-5, 5:-5]

    # Obliczenie kwadratów średnich kroczących
    mu1_sq = mu1 ** 2
    mu2_sq = mu2 ** 2
    mu1_mu2 = mu1 * mu2

    # Obliczenie wariancji i kowariancji obrazów
    sigma1_sq = cv2.filter2D(image1 ** 2, -1, window)[5:-5, 5:-5] - mu1_sq
    sigma2_sq = cv2.filter2D(image2 ** 2, -1, window)[5:-5, 5:-5] - mu2_sq
    sigma12 = cv2.filter2D(image1 * image2, -1, window)[5:-5, 5:-5] - mu1_mu2

    # Obliczenie mapy SSIM
    ssim_map = (
        (2 * mu1_mu2 + C1) * (2 * sigma12 + C2) /
        ((mu1_sq + mu2_sq + C1) * (sigma1_sq + sigma2_sq + C2)) )

    # Zwrócenie średniej wartości mapy SSIM
    return ssim_map.mean()
```

Kod 11 - Algorytm obliczania *SSIM* - podobieństwa strukturalnego
Źródło: *Opracowanie własne*

Przykład 12 - wartości wskaźnika SSIM dla steganogramów

Przykładowe wartości *SSIM* dla par oryginalnego *nośnika* oraz *steganogramów* zakodowanych różnymi technikami *steganograficznymi* zaprezentowane są w tabeli 4.

Tabela 4 - Przykład: wartości *SSIM* dla różnych technik *steganograficznych*
Źródło: *Opracowanie własne*

Użyta technika steganograficzna	***Wartość SSIM***
LSB	0,99
Patchwork	0,99
PVD	0,99
DCT	0,86

Dla technik *steganograficznych LSB*, *Patchwork* oraz *PVD* uzyskane *steganogramy* są z punktu widzenia percepcji ludzkiego oka bardzo bliskie oryginalnemu *nośnikowi*, co ma odzwierciedlenie w wartościach *SSIM* bliskich wartości 1. Natomiast *steganogram* uzyskany techniką *DCT* ma trochę niższą wartość, co wskazuje na większą widoczność zmian wprowadzonych przez zastosowanie tej techniki *steganograficznej*.

Koniec przykładu 12

W 2010 roku A. Horé i D. Ziou opublikowali artykuł *"Image Quality Metrics: PSNR vs. SSIM" (Horé i Ziou, 2010)*, który porównuje metryki *PSNR* oraz *SSIM*. Autorzy analizują to, w jaki sposób każda z tych metryk ocenia podobieństwo obrazów, zwracając uwagę na ich zalety i wady. Opublikowane przez nich wnioski pokazują, że *PSNR* jest metryką, która mierzy proste różnice intensywności między pikselami, co z jednej strony jest łatwe do obliczenia i ma jasne znaczenie fizyczne, ale z drugiej strony nie zawsze dobrze koreluje z percepcją wizualną człowieka, ponieważ nie uwzględnia strukturalnych właściwości obrazu. W przeciwieństwie do *PSNR*, *SSIM* został zaprojektowany z myślą o lepszym odwzorowaniu percepcji wizualnej ludzkiego oka biorąc pod uwagę różne czynniki. *SSIM* ocenia podobieństwo między dwoma obrazami modelując zniekształcenia jako kombinację utraty korelacji, zniekształceń jasności i kontrastu. Wyniki eksperymentalne przedstawione w artykule pokazują, że *SSIM* lepiej odzwierciedla subiektywną ocenę podobieństwa obrazów w porównaniu do *PSNR*, szczególnie w przypadku zniekształceń strukturalnych, takich jak rozmycie lub *kompresja stratna*[85]. Autorzy podkreślają, że chociaż *PSNR* jest szeroko stosowany ze względu na swoją prostotę, *SSIM* oferuje bardziej wiarygodną ocenę podobieństwa obrazów w kontekście percepcji wizualnej, co czyni go odpowiednim narzędziem w wielu zastosowaniach związanych ze *steganografią obrazową*.

Jednym z najprostszych wskaźników używanych do oceny *niewykrywalności steganogramu* jest poziom *entropii* Shannona *(Shannon, 1948)* zdefiniowany wzorem (7).

$$H(X) = -\sum_{i=1}^{n} P(x_i) log_2\big(P(x_i)\big) \tag{7}$$

gdzie: $P(x_i)$ - prawdopodobieństwo wystąpienia wartości x_i.

Duża wartość *entropii* w *nośniku* może sugerować obecność ukrytego *komunikatu*, natomiast mała wartość *entropii* jest skorelowana z większą wartością *niewykrywalności*.

Prosty algorytm obliczający poziom *entropii Shannona* dla obrazu przedstawiony jest w kodzie 12.

[85] **Kompresja stratna** - nieodwracalna metoda redukcji danych, polegająca na eliminacji informacji mniej istotnych z punktu widzenia ludzkiej percepcji. Pozwala ona uzyskać znacznie większy stopień kompresji niż metody bezstratne, wprowadzając zniekształcenia dobrane tak, aby percepcja wzrokowa lub słuchowa pozostała zbliżona do oryginału.

```
def H(image):
    """
    Oblicza entropię Shannona dla obrazu.

    Argumenty:
    image (np.array): obraz wejściowy

    Wyniki:
    float: wartość entropii Shannona dla obrazu
    """

    # Obliczenie histogramu obrazu z 256 binami w zakresie 0-255
    hist, _ = np.histogram(image, bins=256, range=(0, 256), density=True)

    # Usunięcie zerowych wartości z histogramu
    hist = hist[hist > 0]

    # Obliczenie entropii
    entropy = -np.sum(hist * np.log2(hist))

    return entropy
```

Kod 12 - Algorytm obliczania *H* - poziomu *entropii* Shannona
Źródło: *Opracowanie własne*

Przykład 13 - wartości entropii H dla nośnika i steganogramów

Przykładowe wartości *entropii* Shannona *H* dla oryginalnego *nośnika* i *steganogramów* zakodowanych różnymi technikami *steganograficznymi* zaprezentowane są w tabeli 5.

Tabela 5 - Przykład: wartości *entropii H* dla *nośnika* i różnych *technik steganograficznych*
Źródło: *Opracowanie własne*

Użyta technika steganograficzna	***Wartość entropii Shannona H***
Oryginalny *nośnik*	5,11
LSB	7,16
Patchwork	6,00
PVD	6,28
DCT	6,01

Jak można zauważyć, wartość *entropii* Shannona *H* dla wszystkich *steganogramów* jest większa niż dla oryginalnego *nośnika*, co potwierdza, iż zastosowanie dowolnej techniki *steganograficznej* zwiększa *entropię steganogramu*.

Koniec przykładu 13

Do oceny *niewykrywalności* można użyć również analizy statystycznej właściwości *nośnika* za pomocą *procesów Markowa*[86]*,* w tym *łańcuchów Markowa*[87] *(Shi, Chen i Chen, 2006)*. Proces ten obejmuje skonstruowanie modelu zależności między sąsiednimi elementami danych, np. pikselami obrazu, poprzez analizę wartości pikseli w określonym kierunku (poziomo, pionowo, po przekątnej) i stworzenie macierzy zawierającej prawdopodobieństwa przejść między różnymi stanami pikseli, co pozwala opisać regularność i przewidywalność oryginalnych danych *nośnika*. Porównując macierze przejść oryginalnego *nośnika* z macierzami dla *steganogramu*, można wykryć znaczące różnice wskazujące na obecność ukrytego *komunikatu*. Do oceny istotności tych różnic używane są testy statystyczne. Ponieważ zastosowanie *steganografii* zakłóca regularność wartości pikseli, co objawia się w zmianach macierzy przejść, wykrycie takich nieregularności identyfikuje potencjalne ukrycie *komunikatu*. Im więcej zmian może zostać w ten sposób wykrytych, tym mniejsza jest wartość *niewykrywalności*.

2.5.2 Miara pojemności metod steganograficznych

Pojemność[88] to miara maksymalnej ilości *informacji*, którą można zakodować w *nośniku* bez powodowania zauważalnych zmian, które mogłyby zwrócić uwagę na obecność ukrytego *komunikatu (Fridrich, Steganography in Digital Media: Principles, Algorithms, and Applications, 2009; Cox, Miller, Bloom, Fridrich i Kalker, 2007).*

Pojemność może być definiowana na dwa sposoby. Bezwzględnie - jako dostępny całkowity rozmiar pamięci dla dodatkowego *komunikatu* w ramach konkretnego *nośnika*

[86] **Proces Markowa** - proces stochastyczny, w którym przy decyzji o przejściu do kolejnego stanu systemu uwzględnia się wyłącznie obecny stan, co jest określane jako własność braku pamięci *(Meyn i Tweedie, 2009)*.

[87] **Łańcuch Markowa** - *proces Markowa* z dyskretną przestrzenią stanów *(Meyn i Tweedie, 2009)*.

[88] **Pojemność (Capacity)** - miara ilości *informacji*, którą można ukryć w *nośniku* cyfrowym bez zauważalnych zmian jego jakości. Zależy m.in. od właściwości *nośnika* i użytej techniki *steganografii*. Duża *pojemność* umożliwia ukrycie większej ilości *informacji*, ale może jednocześnie zwiększać ryzyko wykrycia zmniejszając *niewykrywalność (Fridrich, Steganography in Digital Media: Principles, Algorithms, and Applications, 2009; Cox, Miller, Bloom, Fridrich i Kalker, 2007)*.

albo względnie - w *bitach* na piksel *bpp*[89] dla obrazów, *bitach* na próbkę dla *audio*, *bitach* na *klatkę* dla *wideo* lub w procentach dostępnego rozmiaru *nośnika*, co pozwala na bardziej uniwersalne porównanie różnych metod *steganograficznych* i ich zdolności do kodowania *komunikatów*. *Pojemność* w *steganografii* jest determinowana w dużym stopniu przez typ, jakość i rozdzielczość *nośnika*. W *nośnikach* o większej jakości i rozdzielczości, np. obrazach, plikach *audio* i *wideo*, można zwykle zakodować więcej *informacji* bez zauważalnego pogorszenia jakości *steganogramu* niż np. w *nośnikach* tekstowych.

Różne metody *steganograficzne* poszukują kompromisu pomiędzy *pojemnością* a *niewykrywalnością* zakodowania informacji, ponieważ większe *pojemności* mogą prowadzić do bardziej wykrywalnych zmian w *steganogramie*. Wybór odpowiedniej metody *steganograficznej* musi uwzględniać balans pomiędzy ilością zakodowanych danych a ryzykiem ich wykrycia.

Bazując na *teorii informacji* Shannona *(Shannon, 1948) pojemność* można zdefiniować w najogólniejszy sposób wzorem (8).

$$C = k * log_2\left(1 + \frac{i}{N}\right) \tag{8}$$

gdzie: k - liczba dostępnych miejsc w *nośniku* do ukrycia danych,
i - *informacja* do ukrycia,
N - poziom *szumu* w *nośniku*.

Liczba dostępnych miejsc k w powyższym wzorze zależy od rodzaju wybranego *nośnika* cyfrowego oraz konkretnej techniki *steganograficznej*.

W przypadku techniki *LSB* w najprostszej wersji polegającej na podmianie jednego najmniej znaczącego *bitu* każdego koloru każdego piksela, wzór na *pojemność* zdefiniowaną bezwzględnie upraszcza się do formuły (9).

$$C = N * M * K \tag{9}$$

gdzie: $N \times M$ - wymiary obrazu,
K - liczba kanałów koloru.

[89] **BPP (bits per pixel)** - miara określająca ilość *bitów* używanych do reprezentacji koloru pojedynczego piksela w obrazie cyfrowym. Większe *bpp* oznacza większą głębię koloru i lepsze odwzorowanie kolorów, np. 1 *bpp* oznacza dwa kolory, 8 *bpp* oznacza 256 kolorów, a 24 *bpp* oznacza ponad 16 milionów kolorów.

Przykład 14 - wartości pojemności informacyjnej dla steganogramu LSB

Pojemność informacyjna dla najprostszej wersji techniki *LSB*, zobrazowanej *steganogramem* z rysunku 35, przy następujących założeniach: (1) wymiary obrazu: N x M = 640 pikseli × 320 pikseli, (2) liczba kanałów koloru: K = 3; wynosi:

$$C = (640 * 320 * 3)\ b \cong 75\ KB.$$

Rysunek 35 - Przykład: *pojemność* dla *nośnika* w przypadku najprostszej techniki *LSB*
Źródło: *Opracowanie własne*

Koniec przykładu 14

2.5.3 Miara odporności metod steganograficznych

Odporność[90] to miara określająca to, w jakim stopniu *komunikat* zakodowany w *steganogramie* jest *odporny* na modyfikacje pliku cyfrowego, np. kompresję, przekształcenia geometryczne oraz zakłócenia *(Provos i Honeyman, 2003; Cox, Miller, Bloom, Fridrich i Kalker, 2007; Fridrich, Steganography in Digital Media: Principles, Algorithms, and Applications, 2009; Katzenbeisser i Petitcolas, 2000)*. Większa wartość *odporności* oznacza większą niezawodność odczytu ukrytego *komunikatu* i większą wytrzymałość *steganogramu* na zakłócenia oraz przekształcenia.

Jednym z podstawowych wskaźników definiujących *odporność* jest *BER*[91] *- bitowa* stopa błędów określająca stosunek liczby błędnie zakodowanych *bitów* do całkowitej liczby

---

[90] **Odporność (Robustness)** - miara zdolności techniki *steganograficznej* do utrzymania *komunikatu* w *steganogramie* pomimo różnych przekształceń, takich jak np. kompresja, zmiany formatu, filtrowanie, obracanie lub skalowanie. Większa *odporność* oznacza, że *komunikat* po takich modyfikacjach z większym prawdopodobieństwem pozostanie nienaruszony i nieusuwalny bądź ilość pozostałej *informacji* w *komunikacie* będzie większa. *(Provos i Honeyman, 2003; Cox, Miller, Bloom, Fridrich i Kalker, 2007; Fridrich, Steganography in Digital Media: Principles, Algorithms, and Applications, 2009; Katzenbeisser i Petitcolas, 2000)*.

[91] **BER (Bit Error Rate) -** metryka oceniająca dokładność kodowania jako udział błędnie zakodowanych bitów *komunikatu* w stosunku do wielkości *komunikatu (Katzenbeisser i Petitcolas, 2000)*.

zakodowanych *bitów komunikatu*, zdefiniowany wzorem (10). Im niższa wartość *BER*, tym mniejsza utrata *informacji* zakodowanej w *komunikacie* oraz większa wartość *odporności*.

$$BER = \frac{bits_{error}}{bits_{all}} \tag{10}$$

gdzie: $bits_{error}$ - liczba błędnie zakodowanych *bitów informacji,*
$bits_{all}$ - liczba wszystkich zakodowanych *bitów informacji*.

Kod 13 zawiera prosty algorytm obliczający wskaźnik *BER* dla *komunikatu* oryginalnego i zdekodowanego ze *steganogramu*.

```
def BER(original_message, decoded_message):
    """
    Oblicza wskaźnik błędów bitowych (BER) między dwoma komunikatami.

    Argumenty:
    original_message (np.array): oryginalny komunikat
    decoded_message (np.array): zdekodowany komunikat

    Wyniki:
    float: wskaźnik błędów bitowych (BER)
    """

    count = 0
    # Iteracja przez oryginalny i zdekodowany komunikat
    for original, decoded in zip(original_message, decoded_message):
        # Obliczenie różnicy bitowej (xor) i zliczenie bitów równych 1
        count += bin(original ^ decoded).count('1')

    # Obliczenie i zwrócenie wskaźnika błędów bitowych (BER)
    return count / (len(original_message) * 8)
```

Kod 13 - Algorytm obliczania wskaźnika *BER* - bitowej stopy błędów
Źródło: *Opracowanie własne*

Im *BER* jest bliższy wartości 0,5, tym bardziej utworzony *steganogram* zawiera losowy *szum* zamiast oryginalnego *komunikatu*, natomiast wartości istotnie powyżej 0,5 oznaczają, iż *steganogram* zawiera odwrócenie oryginalnego *komunikatu*.

Przykład 15 - wartości wskaźnika BER dla steganogramów

Przykładowe wartości *BER* dla par oryginalnego *komunikatu* oraz *komunikatów* zdekodowanych ze *stenogramów* zakodowanych różnymi technikami *steganograficznymi* zaprezentowano w tabeli 6.

Tabela 6 - Przykład: wartości *BER* dla różnych technik *steganograficznych*
Źródło: *Opracowanie własne*

Użyta technika steganograficzna	***Wartość BER***
LSB	0,00
Patchwork	0,20
PVD	0,10
DCT	0,31

W tym przykładzie zastosowanie techniki *LSB* nie zmienia zakodowanego komunikatu, natomiast pozostałe techniki *steganograficzne* 'gubią' niektóre *bity informacji* z oryginalnego *komunikatu*. Uzyskane tutaj wartości wskaźnika *BER* są oczywiście przykładowe i zależne od specyfiki *nośnika* oraz dobranych parametrów w konkretnych algorytmach *steganograficznych*.

Koniec przykładu 15

W przypadku *steganografii transformatowej odporność* można zdefiniować na podstawie analizy zmian współczynników *transformat* w *steganogramie* po wprowadzeniu modyfikacji w oryginalnym *nośniku* zgodnie ze wzorem (11).

$$R = 1 - \frac{\sum_{i=1}^{k} |S_i - S_i'|}{\sum_{i=1}^{k} S_i} \tag{11}$$

gdzie: k - liczba współczynników,
S_i - współczynniki dla nośnika,
S_i' - współczynniki dla steganogramu.

2.6 Dodatkowa literatura przeglądowa z zakresu steganologii

Na przestrzeni ostatnich trzydziestu lat powstało wiele prac naukowych, w tym zarówno monografii, jak i artykułów naukowych, które stanowią kompleksowy przegląd i charakterystykę technik *steganografii*, *steganoanalizy* i cyfrowego *znakowania wodnego*. Autorzy tych publikacji omawiają teoretyczne podstawy ukrywania danych, różnorodne metody *steganografii* i *steganoanalizy* w mediach cyfrowych, takich jak obrazy, tekst, *audio* i *wideo* oraz ich zastosowania w dziedzinach takich jak ochrona praw autorskich, bezpieczeństwo cyfrowe, komunikacja wojskowa i prywatność *informacji*. Prace te poruszają kwestie *pojemności*, *niewykrywalności* i *odporności* systemów *steganograficznych* na ataki *steganoanalityczne*, prezentując zarówno teoretyczne koncepcje, jak i praktyczne implikacje w nowoczesnych *systemach komunikacyjnych*.

Dodatkowo, opracowania te identyfikują kluczowe wyzwania i kierunki przyszłych badań w kontekście rosnących wyzwań związanych z *cyberbezpieczeństwem* oraz rozwojem nowych technologii, takich jak *sieci neuronowe*, *uczenie maszynowe*, chmury obliczeniowe, internet rzeczy IoT i platformy mobilne.

W ramach niniejszej pracy nie sposób ani omówić, ani chociażby wymienić wszystkich tych pozycji naukowych. Oprócz tych, o których już wcześniej w pracy wspomniano, warto zwrócić uwagę na następujące opracowania:

- artykuł *"Information hiding - a survey"* autorstwa F. Petitcolas'a, R. Andersona i M. Kuhn'a z 1999 roku *(Petitcolas, Anderson i Kuhn, 1999)*,
- książka *"Information Hiding Techniques for Steganography and Digital Watermarking"* S. Katzenbeissera i F. Petitcolasa opublikowana w 2000 roku *(Katzenbeisser i Petitcolas, 2000)*,
- książka *"Information Hiding: Steganography and Watermarking - Attacks and Countermeasures"* autorstwa N. Johnsona, Z. Durica i S. Jajodii opublikowana w 2001 roku *(Johnson, Duric i Jajodia, Information Hiding: Steganography and Watermarking - Attacks and Countermeasures, 2001)*,
- artykuł *"Digital Steganography: Hiding Data within Data"* z 2001 roku autorstwa D. Artza *(Artz, 2001)*,
- książka *"Hiding in Plain Sight: Steganography and the Art of Covert Communication"* opublikowana w 2003 roku autorstwa E. Cole'a *(Cole, 2003)*,
- artykuł *"Hide and Seek: An Introduction to Steganography"* z 2003 roku autorstwa N. Provosa i P. Honeymana *(Provos i Honeyman, 2003)*,
- książka *"Data Hiding Fundamentals and Applications"* autorstwa H. Sencara, M. Ramkumara oraz A. Akansu'a opublikowana w 2004 roku *(Sencar, Ramkumar i Akansu, 2004)*,
- artykuł *"An Overview of Image Steganography"* z 2005 roku autorstwa T. Morkela, J. Eloffa oraz M. Oliviera *(Morkel i Eloff, 2005)*,
- książka *"Digital Watermarking and Steganography"* autorstwa I. Coxa, M. Millera, J. Blooma, J. Fridrich oraz T. Kalkera z 2007 roku *(Cox, Miller, Bloom, Fridrich i Kalker, 2007)*,
- książka *"Steganography in Digital Media: Principles, Algorithms, and Applications"* autorstwa J. Fridrich z 2009 roku *(Fridrich, Steganography in Digital Media: Principles, Algorithms, and Applications, 2009)*,

- artykuł *“Digital image steganography: Survey and analysis of current methods”* autorstwa A. Cheddada, J. Condella, K. Currana i P.Kevitta opublikowany w 2010 roku *(Cheddad, Condell, Curran i Kevitt, 2010)*,
- artykuł *“A Survey on Image Steganography and Steganalysis”* autorstwa B. Liego, J. Huanga i Y. Shiego z 2011 roku *(Li, He, Huang i Shi, 2011)*,
- artykuł *“Current Status and Key Issues in Image Steganography: A Survey”* autorstwa M. Subhedara i V. Mankara z 2014 roku *(Subhedar i Mankar, 2014)*,
- artykuł *" Comprehensive survey of image steganography: Techniques, Evaluations, and trends in future research"* z 2019 roku *(Kadhim, Premaratne, Vial i Halloran, 2019)*,
- książka *"Multidisciplinary Approach to Modern Digital Steganography"* z 2021 roku *(Pramanik, Ghonge, Ravi i Cengiz, 2021)*.

3 Zdefiniowanie problemu badawczego

W dotychczasowych badaniach dotyczących *steganologii wideo* zakładano istnienie ciągłego *kanału cyfrowego* pomiędzy *nadawcą* a *odbiorcą*, przez który przekazywany jest *steganogram*. Chociaż wielu badaczy intensywnie analizowało problem *odporności steganografii wideo* na potencjalne zniekształcenia, takie jak przekształcenia lub zakłócenia *steganogramu*, założenia te opierały się na istnieniu ciągłego *kanału cyfrowego* pomiędzy *nadawcą* a *odbiorcą*. Niniejsza praca została zainspirowana pytaniem, czy możliwe jest skonstruowanie techniki *steganografii wideo*, która umożliwiałaby przekazanie *komunikatu* od *nadawcy* do *odbiorcy* nawet w przypadku, gdy *kanał cyfrowy* nie będzie ciągły i część transmisji odbywać się będzie *kanałem analogowym*. W takim scenariuszu *odbiorca* byłby zmuszony do odbioru *komunikatu* np. poprzez bezpośrednią obserwację lub rejestrację ekranu komputera, telewizora lub innego urządzenia wyświetlającego *steganogram wideo* z zakodowanym tajnym *komunikatem* i nie miałby dostępu do fizycznego oryginalnego pliku cyfrowego *steganogramu*, lecz musiałby samodzielnie starać się odtworzyć najlepszą możliwą postać takiego pliku na swoim urządzeniu. W tym celu *odbiorca* musiałby na przykład nagrać obserwowany ekran, a następnie przeprowadzić proces dekodowania *komunikatu* z tak utworzonej nowej wersji *steganogramu wideo*.

To pytanie prowadzi do dalszych rozważań: czy możliwe jest opracowanie techniki *steganografii wideo* o tak dużej *odporności*, że nawet w przypadku wystąpienia istotnie dużych *zniekształceń analogowych* przekazanie *komunikatu* nie zostanie uniemożliwione, a jednocześnie o wystarczającej *niewykrywalności*, aby pozostać niezauważalne dla postronnych osób. W takim kontekście należy założyć, że potencjalny *podsłuchujący* również nie ma dostępu do oryginału pliku cyfrowego *steganogramu* i może przechwycić jedynie analogowy fragment komunikacji, w konsekwencji posiadając co najwyżej tę samą *informację*, co *odbiorca*.

Jak to zostanie pokazane w pracy m.in. poprzez przegląd literatury przedmiotu, powyżej postawione pytanie stanowi w badaniach nad *steganologią wideo* lukę badawczą, którą niniejsza rozprawa próbuje wypełnić.

3.1 Specyfika steganologii wideo

Steganologia wideo[92] polega na ukrywaniu i odkrywaniu *komunikatów* w *nośnikach*, którymi są pliki *wideo*. Ze względu na to, że *nośniki wideo* są bardziej złożone niż w przypadku *steganografii obrazowej, steganografii audio* lub *steganografii lingwistycznej*, *steganologia wideo* jest bardziej skomplikowana i wymaga bardziej złożonych algorytmów kodowania i dekodowania *komunikatów*. Złożoność nośników stanowi jednak również ich zaletę - *steganografia wideo* charakteryzuje się potencjalnie dużo większą *pojemnością* oraz *odpornością* niż w przypadku innych rodzajów *steganografii*.

Na początku XXI wieku wraz z dynamicznym rozwojem technik przetwarzania, kompresji i przesyłania *wideo* przez internet zaobserwowano znaczący wzrost zainteresowania *steganologią wideo*. Pierwsze specjalistyczne publikacje w tej dziedzinie zaczęły pojawiać się już na początku lat 2000. W miarę postępu technologicznego oraz rosnących wymagań dotyczących bezpieczeństwa danych, rozwój technik *steganografii wideo* nabierał tempa, a dzięki zaawansowanym metodom kodowania i kompresji możliwe stało się *steganograficzne* ukrywanie *komunikatów* w *wideo* w sposób niewidoczny dla ludzkiego oka oraz trudny do wykrycia przez automatyczne metody *steganoanalityczne*.

W ostatnich latach opublikowano szereg prac naukowych stanowiących przegląd aktualnego stanu wiedzy i literatury oraz analizujących istniejące metody *steganografii wideo* i odpowiadające im techniki *steganoanalizy.* Opracowania te identyfikują również kluczowe wyzwania i kierunki potencjalnych przyszłych badań w tych obszarach.

Jedną z pierwszych przekrojowych prac stanowiących kompleksowy przegląd metod *steganografii wideo* był opublikowany w 2014 roku przez M. Sadeka i innych artykuł *"Video steganography: a comprehensive review" (Sadek, Khalifa i Mostafa, 2014)*. Autorzy przedstawiają w nim szczegółową analizę różnorodnych technik *steganografii wideo* omawiając ich mocne i słabe strony. W artykule opisuje się zarówno metody działające w *domenie przestrzennej*, jak i *domenie częstotliwościowej*, a także mechanizmy *predykcji międzyklatkowej*. Szczególną uwagę poświęcono aspektom bezpieczeństwa, w tym *niewykrywalności* algorytmów *steganograficznych* prezentując przegląd potencjalnych technik *steganoanalitycznych*. Autorzy dokonują krytycznej analizy istniejących metod

[92] **Steganologia wideo** - *steganologia*, w której *nośnikiem* i *steganogramem* jest plik *wideo*.

wskazując na potencjalne kierunki rozwoju *steganografii wideo* oraz prezentując rekomendacje i wskazówki dotyczące dobrych praktyk w projektowaniu technik *steganografii wideo*.

Kolejny kompleksowy przegląd technik *steganografii wideo* został opracowany w 2018 roku przez L. Yunxia'ego i innych w postaci artykułu *"Video Steganography: A Review" (Yunxia, Shuyang, Yonghao, Hongguo i Si, 2018)*. Autorzy systematyzują w nim i analizują najnowsze osiągnięcia w dziedzinie *steganografii wideo*, przedstawiając m.in. szczegółową taksonomię metod kategoryzując je według różnych kryteriów, takich jak domena osadzania lub typ wykorzystywanych danych *wideo*. W artykule omówiono również różnorodne techniki *steganografii wideo*, w tym metody oparte np. na modyfikacji *wektorów ruchu*, współczynnikach *transformat* częstotliwościowych lub mechanizmach *predykcji wewnątrzklatkowej*. Analiza zalet i wad poszczególnych metod koncentruje się na takich aspektach jak *pojemność* i *niewykrywalność*. Autorzy podkreślają również najnowsze trendy, w tym wykorzystanie technik *uczenia maszynowego* i *sieci neuronowych* do poprawy efektywności kodowania *komunikatów* w *steganogramach wideo*. Na końcu praca identyfikuje ówczesne kluczowe wyzwania oraz potencjalne kierunki rozwoju badań nad *steganografią wideo*.

Jednym z najnowszych interesujących przeglądów osiągnięć w dziedzinie *steganografii wideo* jest praca *"Video Steganography: Recent Advances and Challenges"* autorstwa J. Kunhotha i innych opublikowana w 2023 roku *(Kunhoth, Subramanian, Al-Maadeed i Bouridane, 2023)*. Autorzy przedstawiają w niej szczegółową analizę różnorodnych metod kodowania *komunikatów* w *wideo* obejmujących zarówno techniki działające w domenie przestrzennej, jak i w *domenie częstotliwościowej*. W pracy omówiono m.in. metody *LSB*, *DWT*, *DCT*. Autorzy dokonują krytycznej analizy porównawczej różnych metod oceniając je pod kątem *pojemności* i *niewykrywalności* oraz identyfikują kluczowe wyzwania i potencjalne kierunki przyszłych badań w obszarze *steganografii wideo*.

Zgodnie z definicją *steganografii iteracyjnej* przedstawioną w opracowaniu *(Pery i Waszkowski, A Model and Quantitative Framework for Evaluating Iterative Steganography, 2024)*, *steganografia wideo* jest jednocześnie *steganografią iteracyjną*. Będzie to wykorzystane w dalszej części pracy poprzez odwołanie się do modelu formalnego techniki *steganografii iteracyjnej* oraz poprzez wykorzystanie zastosowania funkcji *IIF* do badania właściwości proponowanej metody *steganografii wideo*.

3.1.1 Metody steganografii specyficzne dla steganogramów wideo

Techniki *steganografii wideo* można podzielić na dwie grupy.

Pierwszą grupę stanowią techniki używane w *steganografii obrazowej* oraz *steganografii audio*, które są adoptowane do wykorzystania w przypadku *wideo*. Przykładem jest wykorzystanie *steganografii obrazowej* do kodowania *komunikatów* bezpośrednio w poszczególnych *klatkach*, które mogą być traktowane jako pojedyncze *nośniki* będące cyfrowymi obrazami. W tym przypadku możliwe jest wykorzystanie wszystkich znanych technik *steganografii obrazowej*, w tym wielokrotnie wskazywanych w przykładach przytaczanych w niniejszej pracy, czyli techniki *steganografii przestrzennej LSB*, *Patchwork* i *PVD* lub techniki *steganografii transformatowej DCT*.

Drugą grupę stanowią techniki specyficzne dla *nośników*, jakimi są pliki *wideo*, które wykorzystują unikalne właściwości związane z faktem, iż poszczególne *klatki* połączone są ze sobą w jedną spójną całość, a przejścia pomiędzy kolejnymi *klatkami* zawierają dodatkowe *informacje*, które mogą być wykorzystywane do zakodowania *komunikatu*. Przykładami takich technik są metody wykorzystujące modyfikacje *wektorów ruchu* oraz mechanizmy *predykcji międzyklatkowej* polegające na ukrywaniu danych poprzez manipulację różnicami pomiędzy kolejnymi *klatkami*.

3.1.1.1 Techniki steganografii wideo wykorzystujące modyfikacje wektorów ruchu

W opublikowanym w 2001 roku artykule *"Video Watermark Technique in Motion Vector"* J. Zhang, J. Li i L. Zhang *(Zhang, Li i Zhang, 2001)* zaprezentowali innowacyjne podejście do cyfrowego *znakowania wodnego* w materiałach *wideo* poprzez osadzanie *znaku wodnego* w *wektorach ruchu*, które są kluczowymi elementami w algorytmach kompresji *wideo*. Technika ta wykorzystuje fakt, że drobne modyfikacje *wektorów ruchu* są trudne do wykrycia zarówno przez ludzkie oko, jak i automatyczne systemy *steganoanalityczne*, co czyni ją szczególnie atrakcyjną w kontekście ochrony praw autorskich. Praca ta była jedną z pionierskich w dziedzinie *steganografii wideo*, skupiając się na wykorzystaniu *wektorów ruchu* do kodowania *komunikatów w steganogramach wideo*.

Steganografia wideo oparta na modyfikacji *wektorów ruchu* polega na subtelnej ingerencji w parametry tych wektorów w materiale *wideo*. Proces rozpoczyna się od wyboru bloków pikseli, w których zostanie zakodowany *komunikat*. Bloki te mogą charakteryzować się niewielkimi różnicami w ruchu między kolejnymi *klatkami*, co utrudnia wykrycie

wprowadzonych modyfikacji. W sytuacjach wymagających większej *pojemności steganogramu* możliwe jest również wykorzystanie bloków o większych różnicach w ruchu, jednak kosztem zmniejszenia *niewykrywalności*. Następnym etapem jest modyfikacja wartości *wektorów ruchu* tak, aby zakodować w nich kolejne *bity komunikatu*. Można wykorzystać do tego metody *LSB*, czyli zamieniać najmniej znaczące *bity* oryginalnego *wektora ruchu* na *bity komunikatu*, co minimalnie wpływa na jakość obrazu, utrzymując dużą *niewykrywalność*, alternatywnie można lekko zmienić kierunek lub długość *wektora ruchu*, co również umożliwia zakodowanie informacji, przy czym zmiany te muszą być na tyle subtelne, aby nie były zauważalne, lecz jednocześnie wystarczające, aby przenosić *bity informacji komunikatu*.

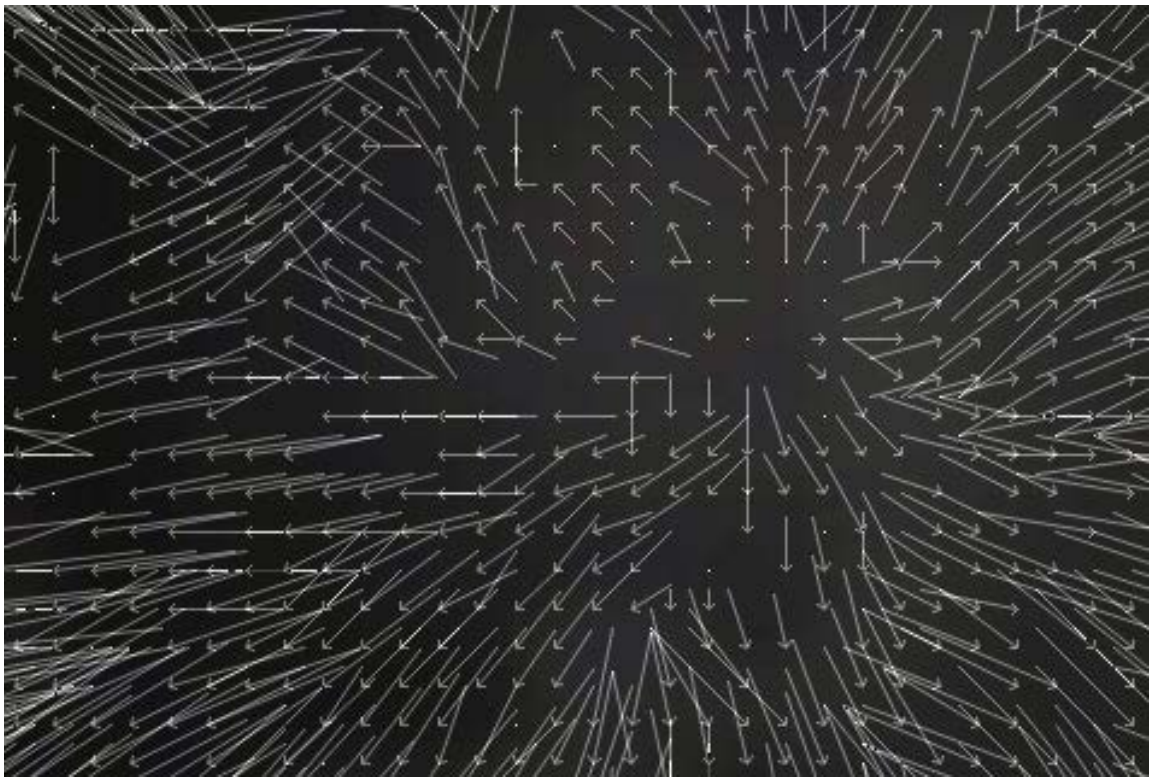
W przeciwieństwie do wcześniejszych metod, które często ingerowały bezpośrednio w piksele poszczególnych *klatek*, podejście zaproponowane przez Zhanga i współautorów pozwala na *znakowanie wodne* bez znaczącego wpływu na *niewykrywalność* lub rozmiar *steganogramu*. Autorzy wykazali, że ich metoda jest odporna na typowe operacje przetwarzania wideo, takie jak kompresja, zachowując integralność osadzonego *komunikatu*. Ta innowacyjna technika otworzyła nowe możliwości w dziedzinie zabezpieczania treści multimedialnych, inspirując dalsze badania nad wykorzystaniem *wektorów ruchu* w *steganografii wideo*.

Kontynuacją i rozwinięciem pracy Zhanga z 2001 roku *(Zhang, Li i Zhang, 2001)* była praca F. Pana, L. Xianga, X. Yanga i Y. Guo'ego zatytułowana *"Video steganography using motion vector and linear block codes"* z 2010 roku *(Pan, Xiang, Yang i Guo, 2010)*. Artykuł przedstawia autorskie podejście do *steganografii wideo*, które łączy wykorzystanie *wektorów ruchu* z *kodami liniowymi*[93]. Technika ta polega na modyfikacji *wektorów ruchu* w sposób, który maksymalizuje *niewykrywalność*, jednocześnie zapewniając dużą *pojemność*. W artykule omówiono również zastosowanie *kodów liniowych* do poprawy *odporności steganogramów*. Dzięki temu podejściu możliwe jest skuteczne ukrywanie dużych ilości *informacji* bez zauważalnej degradacji jakości *steganogramów wideo*. Praca ta, jako jedna z pierwszych, łączy techniki *steganografii wideo* z *kodami liniowymi*, co

[93] **Kody liniowe** - kody korekcyjne, które zabezpieczają dane przed błędami transmisji. Przekształcają bloki danych wejściowych na bloki kodowe za pomocą operacji liniowych, umożliwiając wykrywanie i korekcję błędów poprzez dodanie *bitów* kontrolnych, np. kody Hamminga i Reeda-Solomona *(Ryan i Lin, 2009)*.

znacząco zwiększa *niewykrywalność steganogramów wideo*. Autorzy wykazują, że ich metoda jest skuteczna przy zastosowaniu różnych *kodeków*, np. *MPEG-2 lub H.264*, a proponowana technika przewyższa inne metody *steganografii wideo* pod względem *pojemności* i *niewykrywalności*.

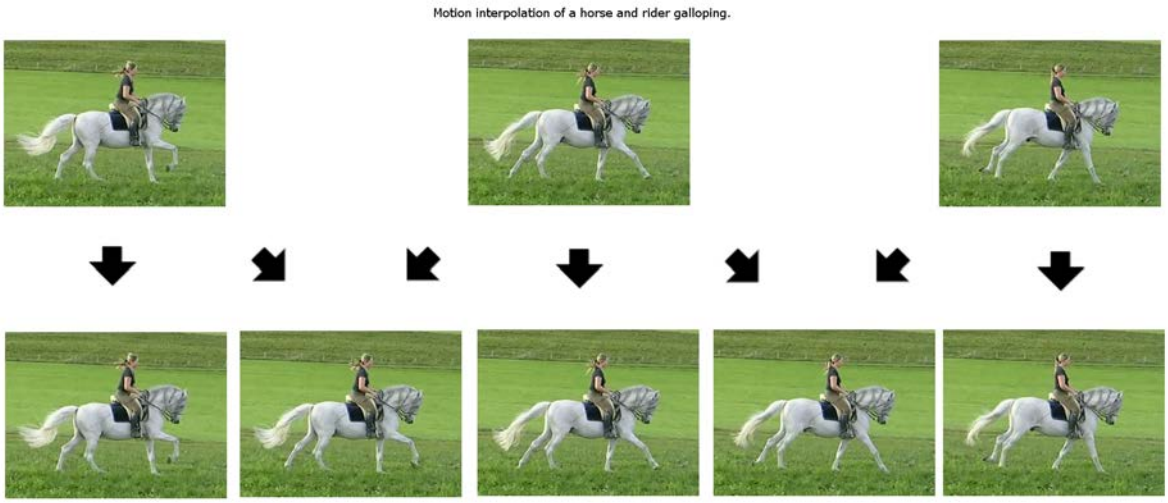
Na rysunku 36 przedstawiono przykładową wizualizację *wektorów ruchu* w *wideo* w formacie *MPEG* powstałych w wyniku ruchu kamery w dwóch kierunkach.



Rysunek 36 - Przykład: wizualizacja *wektorów ruchu* w pliku *wideo* w formacie *MPEG* powstałych w wyniku ruchu kamery w płaszczyźnie obrazu i bocznego przesunięcia.
Źródło: *(c) copyright 2006, Blender Foundation / Netherlands Media Art Institute / www.elephantsdream.org, CC BY 3.0, https://commons.wikimedia.org/w/index.php?curid=8694038*

W celu zapewnienia tego, że modyfikacje *wektorów ruchu* nie wpłyną znacząco na jakość *steganogramu* i *niewykrywalność* pozostanie na wysokim poziomie, można zastosować podejście adaptacyjne. Algorytmy adaptacyjne dostosowują modyfikacje *wektorów ruchu* do lokalnych cech obrazu, takich jak tekstura lub kontrast. Dzięki temu zmiany są bardziej zharmonizowane z naturalnymi właściwościami obrazu, co zwiększa *niewykrywalność* i tym samym utrudnia ich detekcję przez osoby postronne lub specjalistyczne oprogramowanie analizujące *wideo*.

Na rysunku 37 pokazano przykładowe działanie zasady mechanizmu interpolacji ruchu, polegającego na przewidywaniu zawartości *klatek* pośrednich na podstawie *klatek* sąsiadujących.



Rysunek 37 - Przykład: zastosowanie w *klatkach* pliku *wideo* mechanizmu interpolacji ruchu
Źródło: Peregrine Fisher, nr1jack, Waugsberg, CC BY 2.5,
https://commons.wikimedia.org/w/index.php?curid=3508252

Dekodowanie ukrytego *komunikatu* przez *odbiorcę* wymaga znajomości lokalizacji zmodyfikowanych bloków oraz metody użytej do modyfikacji *wektorów ruchu*. Poprzez analizę zmienionych *wektorów ruchu* i np. ich porównanie z wektorami oryginalnymi *odbiorca* może zdekodować poszczególne *bity komunikatu*. Skuteczność tej techniki zależy od precyzyjnego odwzorowania zmian oraz odpowiedniego zabezpieczenia ukrytej komunikacji przed wykryciem, co jest kluczowe dla zachowania integralności przekazu od *nadawcy* do *odbiorcy*.

Manipulowanie *wektorami ruchu* w kontekście *steganografii* wiąże się z istotnymi wyzwaniami związanymi z ich złożonością. Proces ten wymaga zastosowania zaawansowanych technik i algorytmów oraz głębokiej znajomości specyfiki mechanizmów kompresji *wideo*, co czyni go bardziej skomplikowanym w implementacji w porównaniu do prostszych technik *steganograficznych*. Wyzwaniem jest zachowanie odpowiedniej jakości obrazu podczas kodowania *komunikatu*. Znalezienie optymalnej równowagi między skutecznym zakodowaniem *komunikatu* i zapewnieniem dużej *niewykrywalności* jest szczególnie trudne w scenach o złożonym ruchu. W takich przypadkach, nawet niewielkie modyfikacje *wektorów ruchu* mogą prowadzić do zauważalnych artefaktów, co może zdradzić obecność *komunikatu* lub znacząco obniżyć jakość *steganogramu*.

3.1.1.2 Techniki steganografii wideo wykorzystujące mechanizmy predykcji międzyklatkowej

Steganografia wideo oparta na predykcji różnicowej między *klatkami* polega na ukrywaniu *komunikatów* poprzez manipulację różnicami pojawiającymi się między kolejnymi *klatkami*. Techniki te zyskały na znaczeniu szczególnie w kontekście nowoczesnych *kodeków* takich jak *MPEG*, *H.264* lub *H.265*, które wykorzystują mechanizmy *predykcji międzyklatkowej* w celu zmniejszenia ilości danych niezbędnych do zapisu *wideo (Beach i Owen, 2018)*. Dzięki takiemu podejściu możliwe jest osiągnięcie dużej efektywności kompresji, co jednocześnie stwarza nowe możliwości dla technik *steganografii wideo* wykorzystujących mechanizmy *predykcji międzyklatkowej*.

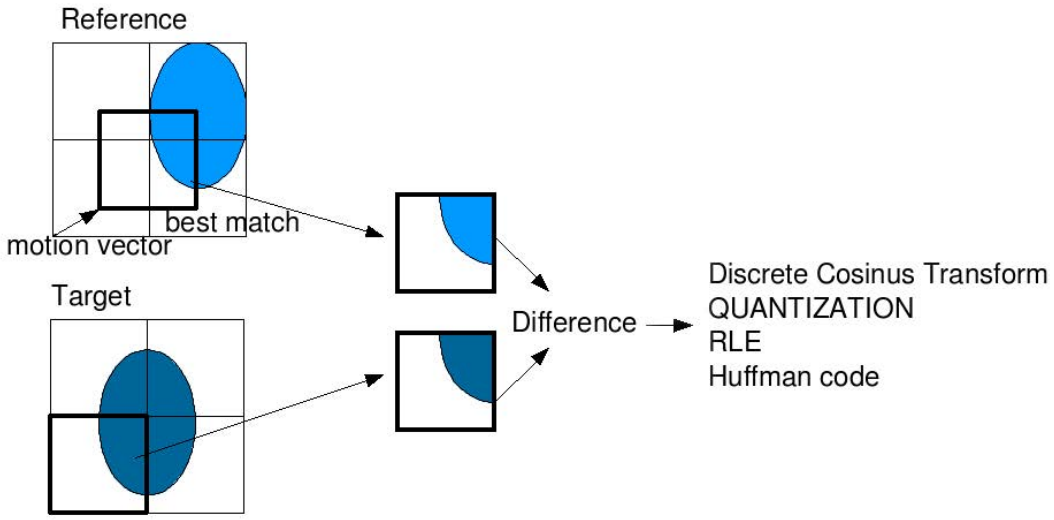
Wideo składa się z wielu *klatek*, z których znaczna część jest do siebie bardzo podobna. Większość *kodeków* wykorzystuje tę cechę zapisując jedynie różnice między *klatkami* zamiast pełnych danych dla każdej z nich, co znacząco redukuje rozmiar pliku. W ramach tego procesu stosuje się trzy główne typy klatek: *klatki* I (Intra-coded frames), które są pełnymi *klatkami* zakodowanymi niezależnie od innych *klatek*; *klatki* P (Predictive-coded frames), które są zakodowane na podstawie różnic w stosunku do poprzednich *klatek*, zazwyczaj *klatek* I lub wcześniejszych klatek P; oraz *klatki* B (Bidirectionally-predictive coded frames), zakodowane na podstawie różnic w stosunku do poprzednich i/lub następnych *klatek wideo*, zarówno I, jak i P.

W kontekście ukrywania *komunikatu* w różnicach między *klatkami*, istnieje kilka podejść pozwalających na efektywne kodowanie dodatkowych *informacji*. Jednym z nich jest modyfikacja *reszt kompresji*, które reprezentują różnice pomiędzy przewidywanymi a rzeczywistymi wartościami pikseli po zastosowaniu *predykcji różnicowej*. Ukrywanie danych może polegać na subtelnej manipulacji tymi resztami, na przykład poprzez zmianę najmniej znaczących bitów *LSB*, co pozwala na zakodowanie informacji bez zauważalnego wpływu na końcowy obraz.

Kolejną metodą jest manipulacja współczynnikami *predykcji różnicowej*, które determinują to, w jaki sposób obliczane są różnice między *klatkami*. Zmiana tych współczynników w odpowiedni sposób umożliwia przenoszenie *bitów komunikatu* w taki sposób, że różnice w *steganogramie* są niezauważalne lub trudne do wykrycia. Technika ta pozwala na bardziej zaawansowane ukrywanie danych, co może być szczególnie użyteczne w przypadkach, gdzie wymagane jest zachowanie dużej *niewykrywalności*.

Innym możliwym podejściem jest wprowadzanie subtelnych zmian w kodowaniu B-frames i P-frames, które bazują na różnicach w stosunku do innych *klatek*. W tym przypadku można manipulować wartościami różnic w sposób, który dodaje *bity komunikatu* do *steganogramu*, pozostając przy tym niezauważalnym dla ludzkiego oka.

Na rysunku 38 zaprezentowano przykład procesu *predykcji międzyklatkowej* polegającej na zmianie jasności bloku w klatce referencyjnej i bloku kodowanego.



Rysunek 38 - Przykład: proces *predykcji międzyklatkowej* polegający na zmianie jasności bloku w klatce referencyjnej i bloku, który jest kodowany. Źródło: *Public Domain, https://commons.wikimedia.org/w/index.php?curid=5396150*

Podobnie, jak w przypadku innych technik, kluczowym wyzwaniem przy stosowaniu tych metod jest zapewnienie, że wprowadzone zmiany są na tyle subtelne, aby nie zostały wykryte przez algorytmy *steganoanalizy*, a jednocześnie zapewniały dużą *odporność* i *pojemność informacyjną*.

3.1.1.3 Techniki steganografii wideo wykorzystujące mechanizmy kodowania wewnątrzklatkowego

Techniki *steganografii wideo* wykorzystujące mechanizm *kodowania wewnątrzklatkowego* są rozwinięciem technik *steganografii obrazowej* bazujące na specyficznych właściwościach *kodeków wideo*.

Przykładem techniki *steganografii wideo* wykorzystującym mechanizm *kodowania wewnątrzklatkowego* w standardzie kodowania *H.264* jest metoda zaproponowana przez M. Cao'ego, L. Tiana i C. Li'ego w artykule *"A Secure Video Steganography Based on the*

Intra-Prediction Mode" opublikowanym w 2020 roku *(Cao, Tian i Li, 2020)*. Autorzy proponują adaptacyjne podejście do selekcji bloków wybranych do kodowania kolejnych *bitów komunikatu* oraz wykorzystanie do tego również *reszt kompresji* w zmodyfikowanych blokach, co ma na celu zachowanie dużej *niewykrywalności*. Autorzy przeprowadzili również szczegółową analizę wpływu różnych strategii osadzania *informacji* na uzyskiwane wartości *niewykrywalności* tworzonych *steganogramów*.

3.1.1.4 Techniki steganografii wideo wykorzystujące uczenie maszynowe

Podobnie jak w przypadku technik *steganografii obrazowej* wykorzystujących *uczenie maszynowe* i głębokie *sieci neuronowe*, w przypadku *steganografii wideo steganografia maszynowa* stanowi jedną z najbardziej perspektywicznych i dynamicznie rozwijających się dziedzin *steganologii*. Poniżej przedstawione zostaną przykłady różnych podejść do *steganografii wideo* wykorzystującej *uczenie maszynowe* bazujące na różnych modelach *sieci neuronowych*. Warto podkreślić, że wszystkie przykłady pochodzą z kilku ostatnich lat, co wskazuje na to, jak świeże jest to zagadnienie.

Przykładem wykorzystania w *steganografii wideo* generatywnej sieci przeciwstawnej *GAN* w modelu *end-to-end*[94] jest technika zaproponowana w artykule *"An End-to-End Video Steganography Network Based on a Coding Unit Mask"* z 2022 roku przez H. Chai'na, Z. Li'ego, F. Li'ego i Z. Zhanga *(Chai, Li, Li i Zhang, 2022)*. Technika nazwana PyraGAN łączy generatywną sieć przeciwstawną *GAN* w modelu *end-to-end* z wykorzystaniem adaptacyjnego mechanizmu identyfikacji istotnych kanałów cech i regionów obrazu. W kolejnym artykule *"An End-to-End Robust Video Steganography Model Based on a Multi-Scale Neural Network"* opublikowanym w 2022 roku *(Xu, Li, Zhang i Liu, 2022)* autorzy przedstawili zaawansowany model *steganografii wideo* wykorzystujący uczenie *sieci neuronowej* w architekturze *end-to-end* MSNN (Multi-Scale Neural Network), która przetwarza informacje na różnych skalach przestrzennych lub czasowych wykorzystując hierarchiczne warstwy do analizy cech na różnych poziomach szczegółowości. MSNN jest szczególnie skuteczna w zadaniach wymagających uchwycenia

[94] **End-to-end** - model *sieci neuronowej* przetwarzającej surowe dane wejściowe i generującej pożądane wyjścia w sposób zintegrowany, bez ręcznego projektowania cech pośrednich. *Sieć neuronowa end-to-end* uczy się istotnych reprezentacji i przekształceń podczas treningu, co upraszcza modelowanie i zwiększa jej elastyczność *(Goodfellow, Bengio i Courville, Deep Learning, 2016)*.

złożonych, wielopoziomowych zależności, takich jak rozpoznawanie obrazów lub analiza sygnałów. Badania autorów pracy koncentrowały się na stworzeniu modelu *steganografii wideo* opartego na generatywnych wieloskalowych sieciach przeciwstawnych. Model składa się z trzech głównych komponentów: enkodera, dekodera i dyskryminatora. Kluczowym elementem jest wprowadzenie warstwy *szumu* pomiędzy enkoderem a dekoderem, co pozwala modelowi na zwiększenie *odporności* wobec popularnych *kodeków* takich jak np. *H.264* i *H.265*. Wyniki eksperymentów pokazane przez autorów wskazują, że proponowany model osiąga dużą *niewykrywalność*, *pojemność* oraz *odporność* na przekształcenia i zniekształcenia.

Artykuł *"Large-capacity and Flexible Video Steganography via Invertible Neural Networks"* autorstwa C. Mou'ego i innych, opublikowany w 2023 roku *(Mou i inni, 2023)* przedstawia kolejne autorskie podejście do *steganografii wideo*, które wykorzystuje odwracalne *sieci neuronowe* INN (Invertible Neural Network). Są one zaprojektowane z myślą o odwracalności co oznacza, że wejście sieci może być dokładnie odtworzone na podstawie wyjścia. Takie sieci są szczególnie przydatne w zadaniach związanych z modelowaniem probabilistycznym, generatywnym oraz w problemach odwrotnych, gdzie tradycyjne *sieci neuronowe* mogą mieć trudności z dokładnym odtworzeniem danych wejściowych. Dzięki odwracalności INN minimalizują straty *informacji*, co czyni je bardziej efektywnymi i precyzyjnymi w zastosowaniach wymagających dokładnego odwzorowania danych. Autorzy proponują wykorzystanie INN do ukrywania i odzyskiwania wielu *komunikatów wideo* w jednym *steganogramie wideo*. Autorzy proponują architekturę *sieci neuronowej* o dużej pojemności i elastyczności LF-VSN (Large-capacity and Flexible Video Steganography Network), która pozwala na zakodowanie i zdekodowanie jednocześnie do siedmiu *komunikatów* w jednym *steganogramie* przy zachowaniu dużej *odporności*. Autorzy proponują również użycie *kluczy kryptograficznych*, które umożliwiają różnym *odbiorcom* dekodowanie dedykowanych dla nich *komunikatów* z tego samego *steganogramu*. Wyniki eksperymentalne przedstawione przez autorów pokazują, że proponowana metoda osiąga lepsze wyniki w zakresie *pojemności* i *odporności* w porównaniu do innych porównywanych technik *steganografii wideo*.

Wykorzystanie technik głębokiego uczenia konwolucyjnych sieci neuronowych *CNN* w *steganografii wideo zaproponował* R. Sushmy i G. Manjuli w artykule *"StegVRN: Enhancing Quality of Video Steganography Using CNN-Based Techniques"* opublikowanym w 2024 roku *(Sushma i Manjula, 2024)*. Praca przedstawia autorski pomysł nowej metody *steganografii wideo* z wykorzystaniem technik głębokiego uczenia

konwolucyjnych sieci neuronowych *CNN*. Głównym celem badania było poprawienie *niewykrywalności steganografii wideo* poprzez zastosowanie *CNN* do adaptacyjnej selekcji obiektów wykorzystywanych do kodowania *komunikatów*, co umożliwia precyzyjne ukrywanie *informacji* w wybranych częściach *wideo*. Autorski algorytm StegVRN łączy w sobie zaawansowane techniki wykrywania punktów narożnych w *klatkach* oraz metodę ukrywania danych za pomocą zastępowania czterech najmniej znaczących *bitów* przy pomocy *LSB*. W artykule szczegółowo opisano proces implementacji systemu, w tym zastosowanie *szyfrowania* danych, co dodatkowo zwiększa bezpieczeństwo zakodowanego *komunikatu*. Wyniki eksperymentów pokazane przez autorów wskazują, że proponowana metoda nie tylko utrzymuje dużą *niewykrywalność steganogramu wideo*, ale także jest *odporna* na różne rodzaje przekształceń i zniekształceń. Według autorów proponowana przez nich metoda przewyższa inne techniki *steganograficzne* pod względem *odporności* oraz *pojemności* ukrywania danych.

Interesujące podejście do problemu zmniejszania rozdzielczości obrazu - downsamplingu, powszechnie stosowanego przy tworzeniu miniatur wideo w *serwisach społecznościowych*[95], przedstawił Y. Wang w artykule „Hiding Data Within Thumbnail Videos: An Adaptive Downsampling-Resilient Video Steganography Method" *(Wang Y., 2024)*. Autor zaproponował autorską technikę *steganograficzną* skupiającą się na ukrywaniu danych w miniaturach *wideo*. Proponowana adaptacyjna metoda *steganografii wideo* jest odporna na zmniejszanie *rozdzielczości* obrazów, odpowiadając na wyzwania stawiane przez współczesne platformy mediów społecznościowych. Kluczowym aspektem tej techniki jest zdolność do zachowania ukrytych *informacji* nawet po procesie downsamplingu. Metoda wykorzystuje adaptacyjne techniki osadzania danych, uwzględniając specyfikę procesu zmniejszania rozdzielczości *wideo*. Autor skupił się na opracowaniu metody pozwalającej na efektywne kodowanie *komunikatu* w obszarach *wideo* najmniej podatnych na zniekształcenia podczas zmiany *rozdzielczości*, zachowując przy tym dużą *niewykrywalność steganogramu*.

[95] **Serwis społecznościowy** - internetowa platforma umożliwiająca użytkownikom tworzenie profili, nawiązywanie kontaktów oraz dzielenie się treściami, takimi jak posty, zdjęcia i filmy. Przykładami serwisów społecznościowych są YouTube, Facebook i Twitter (X).

3.1.2 Metody steganoanalizy specyficzne dla steganogramów wideo

Ze względu na fakt, iż w *steganografii wideo* możliwe jest zastosowanie technik typowych dla *steganografii obrazowej* lub *steganografii audio*, ale jednocześnie możliwe jest również zastosowanie technik specyficznych dla właściwości *nośnika wideo*, jakim jest m.in. wykorzystanie zależności pomiędzy kolejnymi *klatkami*, *steganoanaliza wideo* musi uwzględniać obie te możliwości. Techniki *steganoanalizy wideo* stosowane w innych, prostszych rodzajach *steganografii* były już omawiane wcześniej w tej pracy. W tej części uwaga skupiona jest na technikach *steganoanalitycznych* specyficznych ze względu na właściwości *nośnika* jakim jest *wideo*.

Metody specyficzne dla *steganoanalizy wideo* to techniki bazujące na analizach ruchu i różnic międzyklatkowych. Do tej grupy należą techniki wykorzystujące analizę anomalii w predykcji ruchu, porównanie *histogramów* różnic między *klatkami*, analizę zmienności jasności i kolorów, a także detekcję nienaturalnych zmian ruchu i ich rytmiczności. Wszystkie te techniki koncentrują się na wykrywaniu subtelnych zmian w *wektorach ruchu* i innych cechach dynamicznych *wideo*, w tym analizy *reszt kompresji*[96] i analizy struktury *makrobloków*[97], które mogą wskazywać na obecność zakodowanego *komunikatu*.

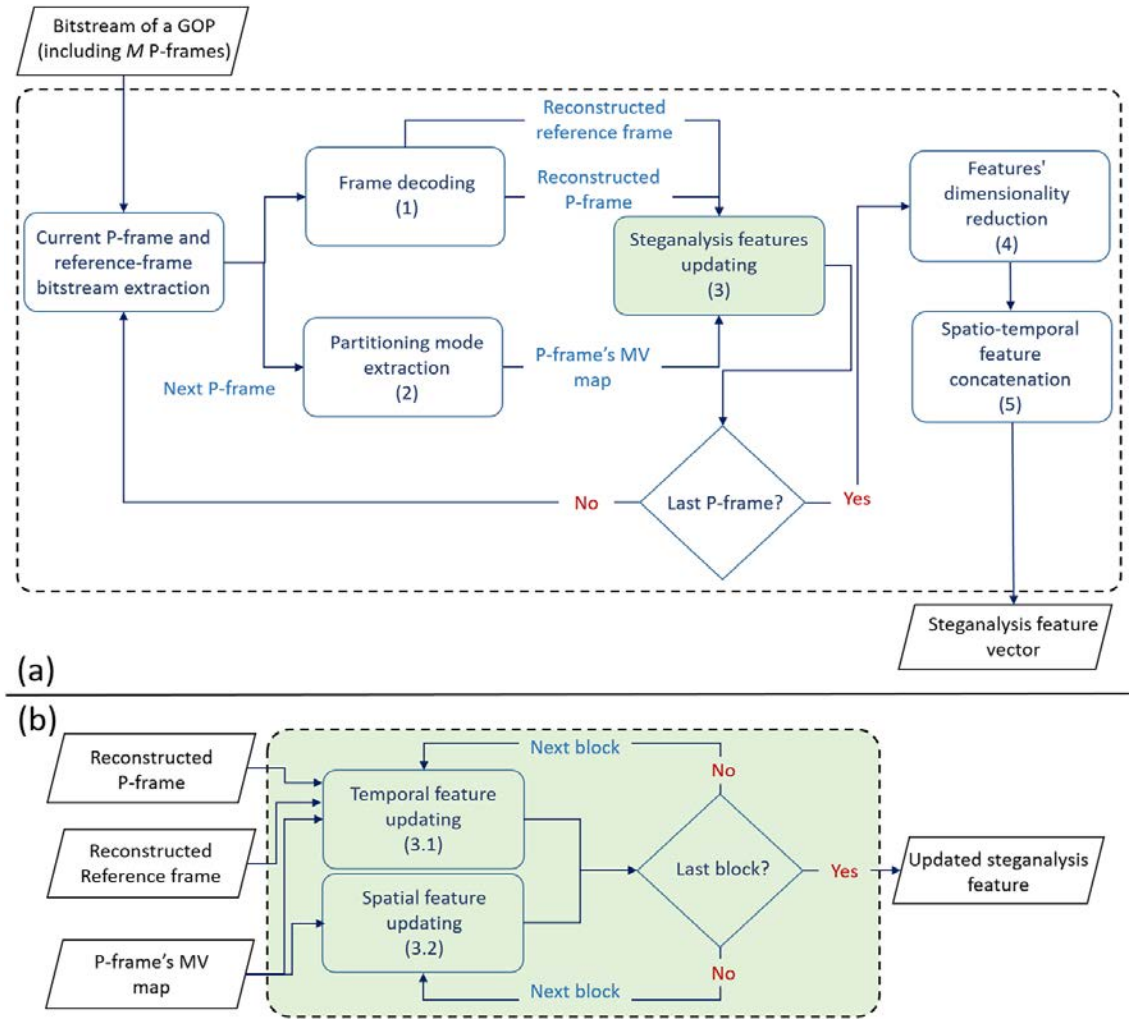
Przykładem metody *steganoanalizy ślepej*, która wykorzystuje statystyki przestrzenno-czasowe *wektorów ruchu* do wykrywania ukrytych *komunikatów,* jest zaproponowana przez N. Ghamsariana i K. Schoeffmanna technika opisana w artykule *"Blind MV-based video steganalysis based on joint inter-frame and intra-frame statistics"* z 2021 roku *(Ghamsarian, Schoeffmann i Khademi, 2021)*. Unikalność zaproponowanego przez autorów podejścia polega na zastosowaniu wielowymiarowego zestawu cech, które uwzględniają zależności między sąsiednimi *wektorami ruchu*, co z kolei pozwala na identyfikację subtelnych zmian wprowadzonych podczas stosowania technik *steganografii*

---

[96] **Reszta kompresji (residual)** - różnica między rzeczywistymi a przewidywanymi wartościami pikseli w *klatkach* kodowana i przesyłana w *wideo*, umożliwiająca rekonstrukcję obrazu podczas procesu dekompresji.

[97] **Makroblok** - podstawowa jednostka przetwarzania obrazu stosowana w standardach *MPEG*, *H.264* i *H.265*. Składa się zazwyczaj z 16x16 pikseli i odgrywa kluczową rolę w *predykcji ruchu*, umożliwiając efektywne kodowanie różnic między *klatkami* za pomocą *wektorów ruchu*. Przy wykorzystaniu *DCT*, *kwantyzacji* i *kodowania entropijnego makrobloki* pozwalają na znaczną redukcję danych przy zachowaniu jakości obrazu.

wideo. Autorzy podkreślają, że ich metoda przewyższa podobne techniki *steganoanalizy* dzięki redukcji wymiarowości cech, co zapobiega *przeuczeniu modelu* oraz możliwości dostosowania do różnych standardów *kodeków*, np. poprzez zmienną wielkość *makrobloków*. Schemat blokowy techniki zaproponowanej przez Ghamsariana i Schoeffmanna zaprezentowany został na rysunku 39.



Rysunek 39 - Przykład: schemat blokowy metody *steganoanalitycznej* badającej *wektory ruchu*
Źródło: *(Ghamsarian, Schoeffmann i Khademi, 2021)*

3.1.3 Problem odporności steganogramów wideo

Ewentualne zakłócenia i przekształcenia *steganogramów wideo* mogą znacząco wpływać na skuteczność ukrywania *komunikatów* oraz na ostateczną jakość *steganogramu wideo* docierającego do *odbiorcy*, co z kolei wpływa na możliwość odczytania przez niego zakodowanego *komunikatu*. Rodzaje możliwych zniekształceń istotnych z punktu widzenia

skutecznego przekazu ukrytego *komunikatu* zależą w dużym stopniu od zastosowanej techniki *steganograficznej*.

Oczywistym jest, że techniki *steganograficzne,* używane w przypadku *steganografii wideo,* powinny być od początku tak projektowane, aby charakteryzowały się dużą *odpornością* na potencjalne zniekształcenia. Najlepszymi strategiami mogącymi znacząco zwiększyć *odporność steganogramów wideo* jest zastosowanie w kodowanych *komunikatach redundancji*[98] danych oraz zaawansowanych algorytmów *kodowania korekcyjnego*.

Poniżej przedstawiono najczęściej występujące zniekształcenia *steganogramów wideo* wraz z ich konsekwencjami dla procesu ukrywania *komunikatów* oraz możliwości ich wykrycia w procesach *steganoanalizy*.

3.1.3.1 Odporność steganogramów wideo na kompresję

Jednymi z najczęściej występujących zniekształceń *steganogramów wideo* są efekty kompresji *wideo*. Używane w tym celu *kodeki*[99], będące naturalnym elementem *kanału cyfrowego* używanego do transmisji *steganogramu* między *nadawcą* a *odbiorcą*, modyfikują zawartość oryginalnego cyfrowego pliku *steganogramu wideo* zmniejszając jego *redundancję* i tym samym *entropię*. Głównym celem działania *kodeków wideo* jest zmniejszenie *bitrate*[100] pliku lub strumienia *wideo*. W przypadku, gdy *kodeki wideo* zostaną zastosowane do *steganogramów wideo,* może to doprowadzić do częściowego lub nawet całkowitego usunięcia ukrytego w *steganogramie* tajnego *komunikatu.* Skutki działania *kodeków wideo* nakładają się na skutki kodowania *steganograficznego*, w efekcie przeciwdziałając technikom *steganograficznym*.

---

[98] **Redundancja** - nadmiarowość *informacji* w systemie, która zwiększa niezawodność przekazu, umożliwiając jego odtworzenie w przypadku błędów lub utraty danych. W *steganografii* wykorzystuje się ją do ukrywania *komunikatu*, np. zmieniając nadmiarowe *bity* bez zauważalnej utraty jakości *steganogramu*.

[99] **Kodek** - oprogramowanie do kodowania i dekodowania, w tym do kompresji i dekompresji obrazów i *wideo*, dzięki któremu można efektywnie przechowywać i przesyłać wysokiej jakości *wideo* przy minimalnym zużyciu zasobów.

[100] **Bitrate** - ilość danych przesyłanych lub przetwarzanych na jednostkę czasu. W kompresji *wideo* określa ilość *bitów* danych na sekundę materiału. Wartość *bitrate* wpływa na jakość i rozmiar pliku. Większy *bitrate* oznacza lepszą jakość i większy rozmiar, a mniejszy *bitrate* odwrotnie. Nowoczesne kodeki dynamicznie dostosowują *bitrate* do złożoności treści, optymalizując kompresję i jakość *wideo*.

W przypadku zastosowania kompresji najczęściej zniekształceniom ulegają następujące parametry *wideo*: ostrość i szczegółowość obrazu, wierność kolorów, płynność ruchu, kontrast oraz spójność czasowa między kolejnymi *klatkami wideo*. Stopień wprowadzonych przez *kodek* zniekształceń zależy od konkretnej techniki kompresji, parametrów *kodeka* oraz szczegółowych charakterystyk oryginalnego *nośnika*. Do najważniejszych i najczęściej używanych przez *kodeki* mechanizmów wykorzystywanych w algorytmach kompresujących należą: *kodowanie międzyklatkowe*[101] wykorzystujące mechanizmy *predykcji międzyklatkowej*[102], *kodowanie wewnątrzklatkowe*[103] wykorzystujące mechanizmy *predykcji wewnątrzklatkowej*[104], *kodowanie entropijne*[105] oraz *kodowanie transformatowe*[106] wykorzystujące mechanizmy *kwantyzacji*[107]. Techniki te powodują następujące zniekształcenia *wideo*:

---

[101] **Kodowanie międzyklatkowe** (**Inter-frame coding)** - technika kompresji *wideo* wykorzystująca redundancję czasową między *klatkami* i mechanizm *predykcji międzyklatkowej*. Kodowane są tylko różnice między *klatkami*, co pozwala na znaczne zmniejszenie ilości danych. Kluczowe *klatki* (I-frames) są w pełni kodowane, a pozostałe (P-frames, B-frames) bazują na różnicach względem innych. Dzięki temu uzyskuje się większą efektywność kompresji *(Tekalp, 1995; Gao i Ma, 2015)*.

[102] **Predykcja międzyklatkowa** - technika wykorzystywana przez *kodeki* w procesie *kodowania międzyklatkowego*, która umożliwia przewidywanie zawartości kolejnych *klatek* na podstawie różnic względem poprzednich *klatek*, redukując ilość danych potrzebnych do przechowywania i transmisji sekwencji *(Tekalp, 1995; Gao i Ma, 2015; Richardson, 2003)*.

[103] **Kodowanie wewnątrzklatkowe (Intra-frame coding)** - technika kompresji *wideo*, w której każda *klatka* jest kodowana niezależnie, podobnie jak obrazy statyczne, ale wykorzystująca mechanizm *predykcji wewnątrzklatkowej*, czyli przewidywanie wartości pikseli w danej *klatce* na podstawie wartości sąsiednich pikseli. Choć jest mniej efektywne pod względem kompresji w porównaniu do *kodowania międzyklatkowego*, jego zaletą jest większa odporność na błędy i łatwy dostęp do poszczególnych *klatek*, co jest istotne np. w edycji i szybkim przeszukiwaniu *wideo*.

[104] **Predykcja wewnątrzklatkowa** - metoda przewidywania wartości pikseli w bieżącej *klatce* na podstawie sąsiednich pikseli stosowana w procesie *kodowania wewnątrzklatkowego*.

[105] **Kodowanie entropijne -** technika *kompresji bezstratnej* na podstawie częstotliwości występowania symboli, np. kodowanie Huffmana - przypisuje krótsze kody binarne częściej występującym symbolom i dłuższe rzadziej występującym lub kodowanie arytmetyczne - koduje całe ciągi symboli jako pojedyncze liczby zmiennoprzecinkowe z przedziału [0,1) dzieląc ten przedział na podprzedziały proporcjonalne do prawdopodobieństwa wystąpienia symboli *(Nelson i Gailly, 1995; Duda, Tahboub, Gadgil i Delp, 2015)*.

[106] **Kodowanie transformatowe** - technika stosowana w kompresji *wideo* zmniejszająca *redundancję danych* poprzez przekształcenie sygnału z *domeny przestrzennej* do *domeny częstotliwościowej*. Proces obejmuje podział obrazu na bloki, zastosowanie *transformaty*, np. *DCT* i *kwantyzację* jej współczynników poprzez ich zaokrąglenie do określonych poziomów. *(Muchahary, Mondal, Parmar, Borah i Majumder, 2015)*.

[107] **Kwantyzacja** - proces przekształcania sygnałów ciągłych w dyskretne poprzez zaokrąglanie wartości do najbliższych poziomów w określonym zbiorze. Stosowana w przetwarzaniu sygnałów, kompresji danych i grafice komputerowej, umożliwia redukcję ilości danych kosztem wprowadzenia niewielkich zniekształceń *(Bennett W., 1948)*.

a) powstawanie artefaktów blokowych objawiających się jako widoczne krawędzie między blokami pikseli wynikające np. z niezależnego kodowania bloków obrazu w *kodowaniu transformatowym,*
b) występowanie efektu migotania lub rozmycia wokół krawędzi obiektów, spowodowane np. kumulacją efektów kompresji w kolejnych *klatkach* w *kodowaniu międzyklatkowym*,
c) występowanie rozmycia powodującego utratę szczegółów i ostrości obrazu, w szczególności w ciemnych scenach i obszarach *wideo*, spowodowane np. usuwaniem wysokich częstotliwości podczas procesu *kwantyzacji*,
d) przenikanie i zniekształcenia kolorów wynikające z niezależnej kompresji kanałów kolorów, spowodowane np. *kompresją stratną* poszczególnych *klatek* w *kodowaniu wewnątrzklatkowym*,
e) występowanie niespójności w ruchu obiektów pomiędzy *klatkami*, spowodowane np. zniekształceniami *wektorów ruchu*[108] przy zastosowaniu *kodowania międzyklatkowego*.

W zależności od użytej przez *nadawcę* techniki *steganograficznej* wymienione powyżej potencjalne zniekształcenia *wideo* spowodowane zastosowaniem *kodeków* mogą zdegradować lub nawet całkowicie usunąć zakodowany w procesie *steganograficznym* tajny *komunikat* przeznaczony dla *odbiorcy*. Dlatego pożądane jest, aby *nadawca* kontrolował zarówno użytą technikę *steganograficzną*, jak też wybór *kodeka* i algorytmu kompresji użytego w ramach *kanału cyfrowego* z *odbiorcą*. W niektórych przypadkach jest to możliwe, ale w wielu praktycznych zastosowaniach *nadawca* nie ma na to wpływu, ponieważ z konkretnymi *kanałami cyfrowymi* skojarzone są z góry określone *kodeki*. Wiele serwisów *VOD*[109] oraz *serwisów społecznościowych* stosuje własne zasady kompresji *wideo* i używa wybranych przez siebie konkretnych *kodeków* posiadających specyficzne właściwości. W takim przypadku *nadawcy* pozostaje jedynie wybór odpowiednio *odpornej* techniki *steganograficznej*.

[108] **Wektor ruchu** - podstawowy element wykorzystywany przez *kodeki* w *predykcji międzyklatkowej*. *Wektor ruchu* opisuje przesunięcie bloku pikseli pomiędzy kolejnymi *klatkami*, umożliwiając efektywne kodowanie różnic między *klatkami* i redukcję ilości danych potrzebnych do reprezentacji ruchu *(Gao i Ma, 2015)*.

[109] **VOD (Video on demand)** - usługa umożliwiająca użytkownikom wybieranie i oglądanie *wideo* na żądanie niezależnie od harmonogramu nadawania i rodzaju urządzenia. Przykładami serwisów VOD są YouTube i Netflix.

Do najczęściej wykorzystywanych obecnie algorytmów kompresji *wideo* wykorzystujących opisane wcześniej techniki należą: *H.264*[110], *H.265*[111], *MPEG-4 Part 2*[112], *MJPEG*[113], *VP9*[114], *H.266*[115] oraz *AV1*[116].

W artykule *"Performance study of common image steganography and steganalysis techniques"* z 2006 roku *(Kharrazi, Sencar i Memon, 2006)* autorzy analizują skuteczność różnych technik *steganografii obrazowej* oraz odpowiadającym im technik *steganoanalizy*. Autorzy badają metody kodowania *komunikatów* zarówno w *domenie przestrzennej*, jak i *domenie częstotliwościowej,* wykorzystując duży zestaw danych składający się z obrazów *JPEG*. Analizują wpływ rozdzielczości *obrazu* i tekstury na skuteczność wykrywania zakodowanego *komunikatu*. Przeprowadzają porównania wydajności *steganoanalizy* przy różnych wielkościach kodowanych *informacji*. Otrzymane wyniki wskazują, że poziom kompresji *steganogramów* istotnie wpływa na efektywność metod *steganoanalitycznych*.

---

[110] **H.264** - znany również jako *MPEG-4 Part 10* lub *AVC* (*Advanced Video Coding*). Jest jednym z najczęściej stosowanych standardów kompresji *wideo*. *H.264* oferuje wysoką jakość obrazu przy stosunkowo małym *bitrate*. Technika ta wykorzystuje metody *kodowania międzyklatkowego* oraz *kodowania wewnątrzklatkowego*, które pozwalają na redukcję *redundancji* przestrzennej i czasowej. Dzięki temu może osiągnąć dużą kompresję, przy zachowaniu wysokiej jakości *(ITU-T, H.264, 2011)*.

[111] **H.265** - znany również jako *HEVC (High Efficiency Video Coding)*. Jest następcą *H.264* oferując większą efektywność kompresji i mniejszy o około 50% *bitrate,* przy zachowaniu podobnej jakości obrazu. Technika ta wykorzystuje zaawansowane metody *predykcji międzyklatkowej* oraz *predykcji wewnątrzklatkowej* stosując mi.in mechanizmy dynamicznego podziału obrazu na bloki o różnych rozmiarach, co pozwala na lepsze dopasowanie do złożonych scen w *wideo (ITU-T, H.265, 2019)*.

[112] **MPEG-4 Part 2** - standard kompresji *wideo* znany również jako *MPEG-4 Visual*. Wykorzystuje m.in. techniki *kodowania międzyklatkowego* oraz *kodowanie transformatowe (ITU-T, H.263, 1996)*.

[113] **MJPEG (Motion JPEG)** - technika kompresji *wideo* wykorzystująca mechanizmy *kodowania wewnątrzklatkowego*, w której każda *klatka* jest kompresowana jako oddzielny obraz *JPEG*. *MJPEG* jest prosty w implementacji i oferuje wysoką jakość obrazu, ale jego efektywność kompresji jest niższa w porównaniu do standardów takich jak *H.264* lub *H.265* (ITU-T, MJPEG (T.802), 2006).

[114] **VP9** - otwarty standard kompresji *wideo*, prekursor *kodeka AV1*. Wykorzystywany i wspierany m.in. przez liczne przeglądarki internetowe i platformy *VOD (VP9, 2024)*.

[115] **H.266** - znany również jako *VVC (Versatile Video Coding)*. Jest następcą *H.265* i oferuje nawet o 50% lepszą efektywność kompresji niż *H.265*, przy zachowaniu tej samej jakości obrazu. Wykorzystuje zaawansowane metody *predykcji międzyklatkowej*, adaptacyjne podziały bloków obrazu i adaptacyjną optymalizację *wektorów ruchu (ITU-T, H.266, 2024)*.

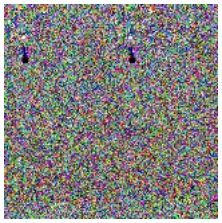
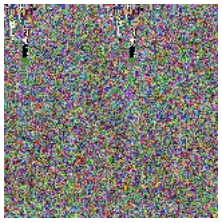
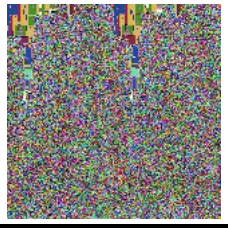
[116] **AV1 (AOMedia Video 1)** - nowoczesny otwarty standard kompresji *wideo* oferujący większą efektywność kompresji niż *H.264* wykorzystujący zaawansowane adaptacyjne metody *predykcji międzyklatkowej* oraz *kodowania transformatowego (Rivaz i Haughton, 2018)*.

Przykład 16 - zniekształcenia steganogramów spowodowane kompresją stratną

W niniejszym przykładzie pokazano wpływ zniekształceń *steganogramów* spowodowanych zastosowaniem *kompresji stratnej JPEG* (przy różnych stopniach kompresji oraz różnych technikach *steganograficznych)* na proces dekodowania *komunikatów* z tych przekształconych *steganogramów*.

Jak pokazano w tabeli 7 w przypadku najprostszej techniki *LSB* zastosowanej w *domenie przestrzennej* użycie *kodeka JPEG* całkowicie usuwa informację zawartą w *komunikacie*, niezależnie od stopnia kompresji.

Tabela 7 - Przykład: zniekształcenia *steganogramu LSB* spowodowane kompresją
Źródło: *Opracowanie własne*

LSB					
Stopień kompresji steganogramu	***Wielkość pliku steganogramu***	***Steganogram***	***SSIM***	***Zdekodowany komunikat***	***BER***
oryginał	553 KB		0.99		0.00
10	127 KB		0.97		0.50
25	84 KB		0.93		0.50
55	54 KB		0.83		0.50

Komunikatem w tym przykładzie jest zdjęcie psa Groma, kodowane metodą *LSB* w *domenie przestrzennej* w *nośniku* będącym zdjęciem El Capitane. Po zakodowaniu *komunikatu steganogram* jest poddawany *kompresji stratnej* z różnymi poziomami

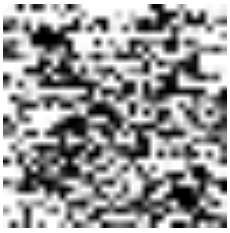
kompresji (od oryginału - czyli poziomu 0, do poziomu 55 odpowiadającym wysokiemu poziomowi kompresji). W kolumnie *"Zdekodowany komunikat"* można zobaczyć efekt działania funkcji dekodującej próbującej odczytać zakodowany *komunikat* w *steganogramie* w zależności od stopnia kompresji. Wartości wskaźnika *BER* w kolejnej kolumnie pokazują stopień błędów dekodowania.

Widać wyraźnie, że dla stopnia kompresji = 0, zdjęcie psa jest wyraźne, a wartość wskaźnika *BER* wskazuje, że nie ma żadnych błędów dekodowania komunikatu (wszystkie *bity informacji komunikatu* zostały odczytane bezbłędnie). Natomiast dla każdego wyższego stopnia kompresji (10, 25, 55), funkcja dekodująca zwraca tylko szum, a zdjęcie wynikowe nie jest w jakimkolwiek stopniu podobne do zdjęcia psa co oznacza, że zdekodowany *komunikat*, który dociera do *odbiorcy steganogramu* jest całkowicie losowy.

W kolejnych przykładach, ze względu na mniejszą *pojemność* informacyjną analizowanych technik *steganograficznych*, użyto *komunikatu* w postaci mniejszego czarno-białego zdjęcia głowy Groma zamieszczonego na rysunku 8.

W przypadku techniki *Patchwork*, co pokazano w tabeli 8, kompresja *JPEG* powoduje szybką utratę *informacji* zawartą w *komunikacie*, tym większą, im większy jest stopień kompresji. W odróżnieniu od wcześniejszego przykładu dla techniki *LSB*, widać na tym przykładzie, że degradacja *informacji* w zdekodowanym *komunikacie* ze skompresowanego *steganogramu* następuje już od zerowego poziomu kompresji, ale za to przy małych poziomach kompresji (np. 10), da się ciągle zauważyć zarys oryginalnego zdjęcia psa, a wartości wskaźnika *BER* (0.41) pokazują, że zachowane zostało około 10% informacji. Przy wyższych poziomach kodowania (25, 55) zachowane jest już tylko 5% informacji, co skutkuje tym, że na uzyskanym zdjęciu bardzo trudno dopatrzyć się podobieństwa do oryginalnego *komunikatu*.

Tabela 8 - Przykład: zniekształcenia *steganogramu Patchwork* spowodowane kompresją
Źródło: *Opracowanie własne*

Patchwork					
Stopień kompresji steganogramu	***Wielkość pliku steganogramu***	***Steganogram***	***SSIM***	***Zdekodowany komunikat***	***BER***
oryginał	542 KB		0.99		0.20
10	127 KB		0.97		0.41
25	84 KB		0.93		0.45
55	54 KB		0.83		0.45

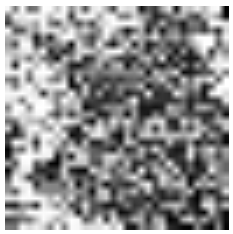
Z kolei w przypadku techniki *PVD*, jak wynika z tabeli 9, podobnie jak w przypadku techniki *LSB,* zastosowanie kompresji *JPEG* niezależnie od jej stopnia powoduje całkowitą utratę *informacji* w *komunikacie*. Jedynie dla zerowego stopnia kompresji funkcja dekodująca jest w stanie odzyskać *komunikat* bliski oryginałowi. Dla kolejnych stopni kompresji (10, 25, 55) wartości wskaźnika *BER* oraz to, co widać na odzyskanych zdjęciach, pokazują wyraźnie, że w przypadku zastoso*wania kompresji stratnej* dla tej techniki *steganograficznej* tracona jest praktycznie cała *informacja*.

Tabela 9 - Przykład: zniekształcenia *steganogramu PVD* spowodowane kompresją
Źródło: *Opracowanie własne*

PVD					
Stopień kompresji steganogramu	***Wielkość pliku steganogramu***	***Steganogram***	***SSIM***	***Zdekodowany komunikat***	***BER***
oryginał	540 KB		0.99		0.10
10	127 KB		0.97		0.54
25	84 KB		0.93		0.55
55	54 KB		0.83		0.55

Natomiast z tabeli 10 możemy wyczytać, że w przypadku zastosowania *kompresji stratnej* dla *steganogramu* utworzonego techniką *DCT informacja* w *komunikacie* utrzymuje się nawet przy dużych stopniach kompresji. Głowa psa jest widoczna zarówno dla niskich, jak i wysokich poziomów kompresji. Na tym przykładzie widać kluczową zaletę technik *steganologii transformatowej*.

Tabela 10 - Przykład: zniekształcenia *steganogramu DCT* spowodowane kompresją
Źródło: *Opracowanie własne*

DCT					
Stopień kompresji steganogramu	***Wielkość pliku steganogramu***	***Steganogram***	***SSIM***	***Zdekodowany komunikat***	***BER***
oryginał	538 KB		0.86		0.31
10	117 KB		0.85		0.37
25	82 KB		0.85		0.40
55	57 KB		0.83		0.44

Koniec przykładu 16

3.1.3.2 Odporność steganogramów wideo na transkodowanie

Transkodowanie, czyli proces konwersji jednego formatu *wideo* do innego, jest kluczowym czynnikiem wpływającym na zniekształcenie *steganogramów wideo*. Transkodowanie obejmuje dekodowanie oryginalnego *wideo*, a następnie jego ponowne kodowanie przy użyciu innego *kodeka* lub innych parametrów kompresji. Każdy etap może wprowadzać różnorodne zniekształcenia, które zmieniają strukturę danych, a tym samym wpływają na integralność ukrytego *komunikatu*. W przypadku transkodowania zniekształceniom ulegają analogiczne parametry, jak w przypadku procesu kompresji, czyli ostrość i szczegółowość obrazu, wierność kolorów, płynność ruchu, kontrast i spójność czasowa między *klatkami*.

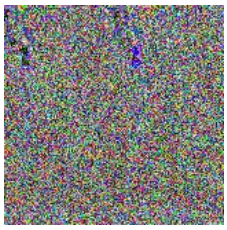
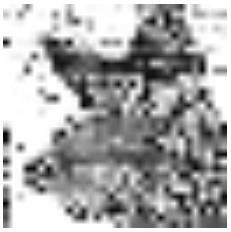
Pierwszym etapem transkodowania jest dekodowanie oryginalnego *wideo*, co samo w sobie może wprowadzać błędy i zniekształcenia. Dekodowanie polega na odtworzeniu obrazu z zakodowanych danych, co może być obciążone stratami wynikającymi z wcześniejszej kompresji. Te straty mogą już na tym etapie zniszczyć ukryty *komunikat*, szczególnie jeśli był zakodowany w bardziej wrażliwych na przekształcenia elementach *wideo*, które są bardziej podatne na *kompresję stratną*, np. jeśli *komunikat* był ukryty w wysokoczęstotliwościowych współczynnikach *DCT*, dekodowanie może spowodować jego degradację lub całkowite usunięcie.

Ponowne kodowanie *wideo* w nowym formacie może dodatkowo zniekształcać *steganogram wideo*. Nowy *kodek* lub inne parametry kompresji mogą różnić się pod względem algorytmów predykcji, kwantyzacji i filtrowania. Te różnice mogą prowadzić do modyfikacji ukrytego *komunikatu*, zwłaszcza jeśli nowy *kodek* stosuje bardziej agresywne techniki kompresji, np. jeżeli oryginalny strumień był zakodowany przy użyciu *kodeka H.264*, a nowy format używa *kodeka H.265*, różnice w algorytmach kompresji mogą spowodować znaczne zmiany w strukturze danych usuwając lub zniekształcając zakodowany *komunikat*. Transkodowanie może wprowadzać także nowe artefakty do *przekształconego steganogramu wideo*, które mogą zasłaniać lub niszczyć szczegóły przechowujące zakodowany *komunikat*. Ewentualna zmiana rozdzielczości *wideo* do większej lub mniejszej zmienia układ pikseli i niszczy strukturę *steganogramu*. Z kolei zmiana przestrzeni kolorów, np. z *RGB* do *YUV,* wpływa na sposób reprezentacji kolorów, co w przypadku technik bazujących na *LSB* może istotnie wpływać na degradację zakodowanego *komunikatu*.

Przykład 17 - przekształcenia steganogramów spowodowane transkodowaniem

W tabeli 11 pokazano wpływ przykładowego przekształcenia *steganogramu* polegającego na transkodowaniu poprzez zamianę formatu obrazu z *PNG* do *JPEG*.

Tabela 11 - Przykład: zniekształcenia *steganogramu* spowodowane transkodowaniem
Źródło: *Opracowanie własne*

Technika steganograficzna	*Przekształcony steganogram na format JPEG*	*Zdekodowany komunikat z formatu JPEG*	*PSNR*		*SSIM*		*BER*	
			PNG	*JPEG*	*PNG*	*JPEG*	*PNG*	*JPEG*
LSB			51.16	35.49	0.99	0.99	0.00	0.49
Patchwork			59.12	35.56	0.99	0.99	0.2	0.35
PVD			69.24	35.57	0.99	0.99	0.10	0.42
DCT			23.90	23.81	0.86	0.86	0.31	0.36

Na to, ile *informacji* uda się zachować w *transkodowanym steganogramie*, ma wpływ typ użytej techniki *steganograficznej*. Dla technik *LSB* oraz *PVD* zastosowanie kompresji *JPEG* oznacza usunięcie praktycznie całej *informacji*. W przypadku *Patchwork* zniekształceniu ulega bardzo duża część *informacji* przechowywanej w *komunikacie*, ale część z tej *informacji* da się jeszcze odczytać. Natomiast w przypadku *steganografii DCT* przekształcenie obrazu do *domeny częstotliwościowej* poprzez przekonwertowanie z formatu *PNG* do *JPEG* nie powoduje istotnego zniekształcenia przekazywanego *komunikatu*.

Koniec przykładu 17

3.1.3.3 Odporność steganogramów wideo na edytowanie i przekształcenia geometryczne

Operacje związane z manualną lub automatyczną edycją *wideo*, wykorzystujące m.in. przekształcenia geometryczne *klatek,* stanowią duże wyzwanie w kontekście *odporności* i integralności zakodowanego w nich *komunikatu*.

Cięcie *wideo* oraz usuwanie i dodawanie do niego elementów może skutkować utratą lub nadpisywaniem części albo całości *komunikatu*.

Usuwanie i dodawanie efektów wizualnych lub zmiana prędkości odtwarzania *wideo* może prowadzić do zniekształcenia oryginalnych pikseli *klatek*, w których zakodowany jest *komunikat*, co znacząco utrudnia jego późniejsze dekodowanie.

Transformacje geometryczne, takie jak rotacje, skalowanie i translacje, stanowią dodatkowe wyzwanie dla technik *steganografii wideo*. W przypadku *steganografii przestrzennej* wpływają one bezpośrednio na zniekształcenia oryginalnych pikseli *klatek*, w których zakodowany jest *komunikat*, co znacząco utrudnia jego późniejsze dekodowanie. Z kolei w przypadku *steganografii transformatowej* operacje te mogą wpływać na pozycję współczynników *transformat* użytych do kodowania *komunikatu* prowadząc do przemieszczenia tych współczynników, co w konsekwencji komplikuje lub uniemożliwia poprawne zdekodowanie *komunikatu* z tak *przekształconego steganogramu*.

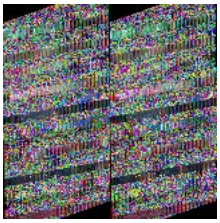
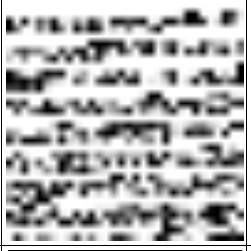
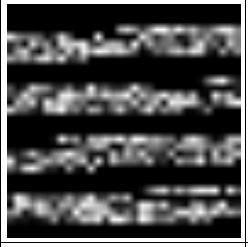
Przycinanie *wideo,* polegające na usunięciu jego części, może prowadzić do bezpośrednich i znaczących skutków dla integralności *komunikatu*. W przypadkach, gdy technika polega na precyzyjnej lokalizacji danych w ramach *steganogramu*, takie przycinanie może całkowicie uniemożliwić poprawne zdekodowanie *komunikatu*. Zaawansowane techniki *steganograficzne* powinny rozpraszać dane w całym obrazie minimalizując ryzyko utraty *informacji* zakodowanej w *komunikacie*.

Zmiana szybkości lub tempa odtwarzania *wideo* obejmująca np. przyspieszenie lub spowolnienie również stanowi istotne zagrożenie dla ukrytej *informacji*. Przekształcenia te mogą zmieniać synchronizację czasową między *steganogramem* a oryginalnym *nośnikiem*, co może powodować trudności w poprawnym dekodowaniu *komunikatu*.

Przykład 18 - przekształcenia geometryczne steganogramów

W tabeli 12 pokazano wpływ obrotu *steganogramu* o 10 stopni na możliwość dekodowania *komunikatu* zakodowanego różnymi technikami *steganograficznymi*.

Tabela 12 - Przykład: zniekształcenia *steganogramu* spowodowane obróceniem i przycięciem
Źródło: *Opracowanie własne*

Technika steganogra-ficzna	*Przekształcony steganogram (obrócony)*	*Zdekodowany komunikat (obrócony)*	*PSNR*		*SSIM*		*BER*	
			oryginał	*obrócony*	*oryginał*	*obrócony*	*oryginał*	*obrócony*
LSB			51.16	10.84	0.99	0.09	0.00	0.48
Patchwork			59.12	10.84	0.99	0.09	0.2	0.45
PVD			69.24	10.84	0.99	0.09	0.10	0.57
DCT			23.90	10.77	0.86	0.09	0.31	0.54

Dla każdej z powyższych przykładowych technik *steganograficznych* zastosowanie przekształcenia geometrycznego polegającego na jednoczesnym obrocie *steganogramu* o 10 stopni i przycięciu obrazu do pierwotnych rozmiarów powoduje całkowite usunięcie *informacji* z *komunikatu*. Należy podkreślić, iż użyte w przykładach algorytmy kodowania i dekodowania są bardzo proste i nie są zoptymalizowane pod kątem *odporności* na tego typu przekształcenia, ale przykład ten jasno pokazuje, że przekształcenia geometryczne stanowią jedno z największych wyzwań związanych z projektowaniem *odpornych* technik *steganograficznych*.

Koniec przykładu 18

3.1.3.4 Odporność steganogramów wideo na błędy transmisji i szum

Steganografia wideo ze względu na specyfikę osadzania *informacji* w *wideo* wykazuje znaczną podatność na degradację *komunikatów* wynikającą z błędów transmisji, takich jak utrata pakietów oraz ewentualne zakłócenia i *szumy*. Na przykład w sieciach charakteryzujących się niską jakością transmisji lub wysokim poziomem ryzyka utraty pakietów, np. w przypadku transmisji wykorzystujących protokół UDP (User Datagram Protocol), który przesyła dane bez gwarancji dostarczenia, może dochodzić do degradacji zakodowanego *komunikatu*.

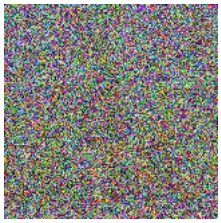
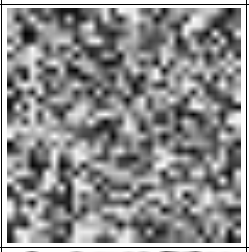
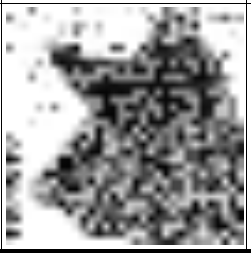
Szumy, które są wprowadzane podczas transmisji, stanowią poważne zagrożenie dla integralności *steganogramów*. Losowe zmiany wartości pikseli wynikające z obecności *szumu* mogą prowadzić do znacznej degradacji, a w skrajnych przypadkach nawet do całkowitej utraty *komunikatu*. Modelowanie zakłóceń odpowiadających procesowi zaszumienia *steganogramu* najczęściej wykonuje się poprzez dodawanie do obiektu *szumu Gaussa*[117] (białego szumu).

[117] **Szum Gaussa** - model zakłóceń oparty na *rozkładzie normalnym (Patel i Read, 1996)*, gdzie wartości losowo rozkładają się wokół średniej. Charakteryzuje się symetrycznym rozkładem i jest powszechnie stosowany w analizie sygnałów i przetwarzaniu obrazów. Dzięki swoim właściwościom dobrze opisuje naturalne zjawiska losowe, co czyni go kluczowym w modelowaniu zakłóceń w systemach *(Jain, 1989)*.

Przykład 19 - zniekształcenia steganogramów spowodowane szumem

W tabeli 13 pokazano wpływ wprowadzenia do *steganogramu szumu Gaussa* na możliwość zdekodowania *komunikatu*.

Tabela 13 - Przykład: zniekształcenia *steganogramu* spowodowane *szumem Gaussa*
Źródło: *Opracowanie własne*

Technika steganogra-ficzna	*Przekształcony steganogram (z szumem Gaussa)*	*Zdekodowany komunikat (z szumem Gaussa)*	*PSNR*		*SSIM*		*BER*	
			oryginał	szum	oryginał	szum	oryginał	szum
LSB			51.16	47.97	0.99	0.99	0.00	0.48
Patchwork			59.12	50.69	0.99	0.99	0.2	0.27
PVD			69.24	51.22	0.99	0.99	0.10	0.49
DCT			23.90	23.89	0.86	0.86	0.31	0.35

Zgodnie z oczekiwaniami, w przypadku dodania *szumu Gaussa steganogramy* utworzone techniką *LSB* oraz *PVD* utraciły całą *informację* zakodowaną w *komunikacie*. W przypadku technik *Patchwork* i *DCT komunikat* został w dużym stopniu zachowany.

Koniec przykładu 19

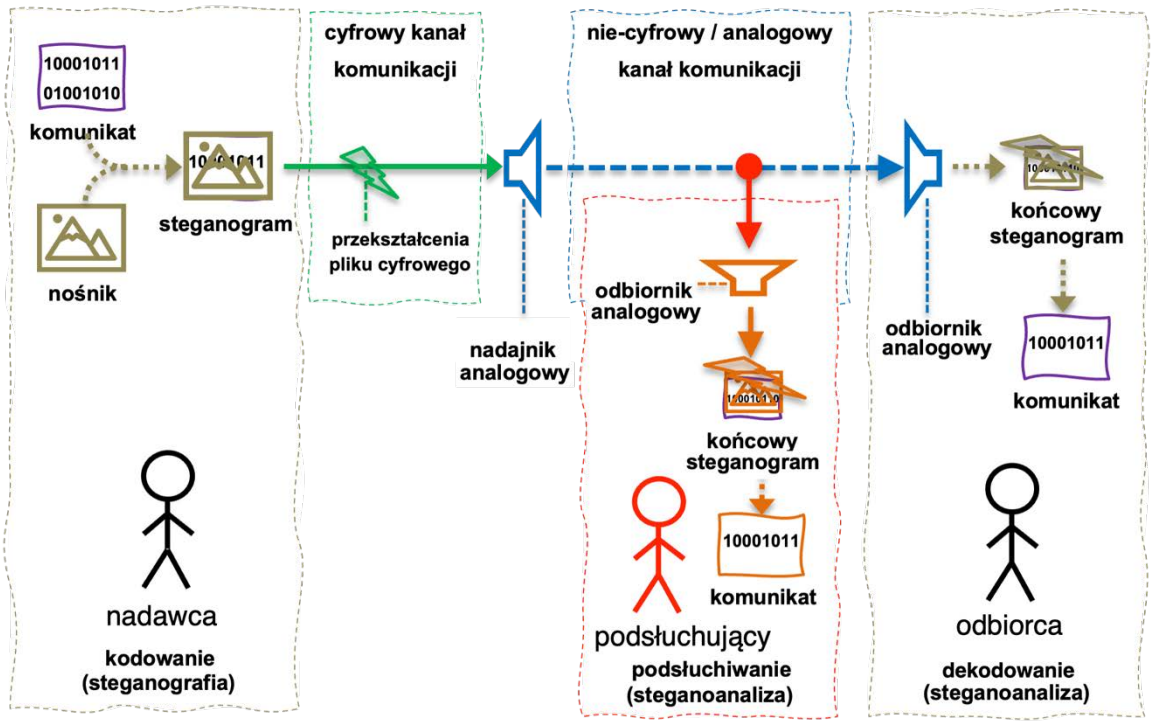
3.2 Zdefiniowanie problemu zniekształceń analogowych

W *podstawowym scenariuszu steganologicznym nadawca* komunikuje się z *odbiorcą* za pośrednictwem *kanału cyfrowego*. Oznacza to, że *odbiorca* otrzymuje od *nadawcy* plik cyfrowy będący *steganogramem*. *Steganogram* ten może ulec pewnym przekształceniom, ale można założyć, że *odbiorca* ma dostęp do *steganogramu* bardzo zbliżonego do oryginalnego pliku cyfrowego. W przypadku *steganologii wideo* oznacza to dostęp do treści poszczególnych *klatek*, struktur danych definiujących przejścia pomiędzy tymi *klatkami* oraz informacji opisujących parametry całego *wideo*, który przesłał *nadawca*.

Jeśli jednak *kanał cyfrowy* nie będzie ciągły i pomiędzy *nadawcą* a *odbiorcą* część transmisji będzie odbywać się drogą analogową, to *odbiorca* nie będzie miał dostępu do wszystkich oryginalnych *informacji* zakodowanych przez *nadawcę* w pierwotnym cyfrowym *steganogramie wideo*. Nowy plik *wideo*, który *odbiorca* będzie musiał sam stworzyć, zachowa jedynie najbardziej istotne cechy z *domeny przestrzennej*, czyli te właściwości, które są widoczne *"*gołym okiem*"*. Scenariusz ten można opisać w następujący sposób:

1. *Odbiorca* tworzy *"*nowy*"* *steganogram*, który będzie się nazywać *końcowym steganogramem*, na podstawie najlepszego dostępnego źródła *wideo*, które jednak nie pochodzi bezpośrednio z *kanału cyfrowego*, lecz z *kanału analogowego* stanowiącego część *kanału* pomiędzy *nadawcą* a *odbiorcą.* Na przykład *odbiorca* może nagrać swoim smartfonem *wideo*, na którym wyświetlany jest ekran z zarejestrowanym *oryginalnym steganogramem* stworzonym przez *nadawcę*.
2. *Podsłuchujący* ma dostęp do tego samego analogowego źródła co *odbiorca*. *Podsłuchujący* nie ma dostępu do żadnej lepszej wersji pliku *wideo* i może jedynie zastosować te same środki techniczne co *odbiorca*, aby utrwalić i poddać *steganoanalizie* utworzony przez siebie *końcowy steganogram*.
3. Wszystkie analizy i próby dekodowania *informacji* prowadzone przez *odbiorcę* (i potencjalnie przez *podsłuchującego*) są wykonywane na *końcowym steganogramie*.

Tak zdefiniowany scenariusz będzie nazywany *scenariuszem zniekształceń analogowych*[118] i będzie on przedmiotem dalszych rozważań w niniejszej pracy. Schemat *scenariusza zniekształceń analogowych* został przedstawiony na rysunku 40. Jest to zmodyfikowany schemat względem *podstawowego scenariusza steganologicznego* poprzez dodanie nowego elementu, jakim jest analogowa część *kanału* komunikacyjnego.



Rysunek 40 - Schemat s*cenariusza zniekształceń analogowych*
Źródło: *Opracowanie własne*

Istotną różnicą w *scenariuszu zniekształceń analogowych,* w porównaniu do *podstawowego scenariusza steganologicznego,* jest zastąpienie w pewnym momencie *kanału cyfrowego* kanałem nie-cyfrowym, analogowym, zwanym dalej w pracy *kanałem analogowym*[119]. Przyjęto, że na początku *kanału analogowego* znajduje się *nadajnik*

[118] **Scenariusz zniekształceń analogowych** - scenariusz przesyłania pomiędzy *nadawcą* a *odbiorcą steganogramu* początkowo przez *kanał cyfrowy*, a od pewnego momentu poprzez *kanał analogowy*. W tym scenariuszu *końcowy steganogram* jest nowo utworzonym przez odbiorcę plikiem cyfrowym różniącym się istotnie od *oryginalnego steganogramu* utworzonego pierwotnie przez *nadawcę*.

[119] **Kanał analogowy** - występująca w *scenariuszu zniekształceń analogowych* część *kanału* pomiędzy *nadawcą* a *odbiorcą*, w ramach którego *steganogram* przekazywany jest pomiędzy *nadajnikiem*

analogowy[120], który przetwarza *oryginalny steganogram*[121] na postać analogową w celu dalszej transmisji w sposób interpretowalny dla zmysłów człowieka - wzroku lub słuchu. Może być to np. emisja obrazu za pomocą ekranu telewizora lub emisja dźwięku za pomocą głośników. Z punktu widzenia rozpatrywanego scenariusza ten właśnie moment jest najbardziej kluczowy z punktu widzenia *odporności* i *niewykrywalności steganogramu*. Z jednej strony *odbiorca* musi umieć odebrać i zinterpretować transmisję analogową, z drugiej strony *podsłuchujący* może podsłuchać tę transmisję i próbować stwierdzić, czy w transmisji zakodowany jest dodatkowy tajny *komunikat*, a jeżeli tak, to jaka jest jego treść. Załóżmy, że zarówno *odbiorca*, jak i *podsłuchujący* dysponują takimi samymi *odbiornikami analogowymi*[122], które wykonują odwrotne przekształcenie do *nadajnika analogowego*, czyli przetwarzają transmisję analogową do cyfrowej tworząc nowy cyfrowy plik, który w tym scenariuszu będzie nazywać się *końcowym steganogramem*[123]. *Odbiornikiem analogowym* może być np. kamera w smartfonie lub mikrofon. W rozpatrywanym scenariuszu zarówno *odbiorca*, jak i *podsłuchujący* będą poddawać analizie *końcowe steganogramy*, ale będą to już inne cyfrowe pliki, niż ten, który pierwotnie wysłał *nadawca* jako pierwotny *oryginalny steganogram*.

Zniekształceniami analogowymi[124] w *scenariuszu zniekształceń analogowych* będą nazywać się wszystkie przekształcenia i zakłócenia, jakim poddany jest *oryginalny steganogram* w procesie transmisji przez *kanał* od *nadawcy* do *odbiorcy*, w tym zwłaszcza poprzez *kanał analogowy*.

---

analogowym a *odbiornikiem analogowym* w sposób analogowy, np. poprzez wyświetlenie *steganogramu wideo* na ekranie odbiornika telewizyjnego.

[120] **Nadajnik analogowy** - urządzenie przetwarzające cyfrowy *steganogram* na postać analogową, np. ekran odbiornika telewizyjnego.

[121] **Oryginalny steganogram** - w *scenariuszu zniekształceń analogowych* utworzony przez *nadawcę* plik cyfrowy będący pierwotnym *steganogramem*, który *nadawca* wysyła do *odbiorcy*.

[122] **Odbiornik analogowy** - urządzenie przetwarzające analogową postać *steganogramu* ponownie do postaci pliku cyfrowego, np. kamera smartfonu.

[123] **Końcowy steganogram** - w *scenariuszu zniekształceń analogowych* utworzony przy pomocy *odbiornika analogowego* nowy plik cyfrowy, którym dysponuje *odbiorca* oraz potencjalnie *podsłuchujący kanał analogowy*.

[124] **Zniekształcenia analogowe** - w *scenariuszu zniekształceń analogowych* suma przekształceń i zakłóceń, jakim poddany jest *oryginalny steganogram* w procesie transmisji przez *kanał* od *nadawcy* do *odbiorcy*.

Założenie, że *odbiorca* i *podsłuchujący* dysponują takimi samymi *odbiornikami analogowymi* sprowadza się do tego, że *podsłuchujący* nie jest w stanie odtworzyć z podsłuchiwania *kanału analogowego* więcej *informacji* niż *odbiorca*. W innym przypadku *podsłuchujący* byłby w uprzywilejowanej pozycji wobec docelowego *odbiorcy*. Dla uproszczenia dalszych rozważań można założyć, że *podsłuchujący* może stworzyć zawierający dokładnie te same *informacje* plik cyfrowy *końcowego steganogramu*, co *odbiorca*.

Ze względu na fakt, że niniejsza praca skupia się na *steganologii wideo*, przyjęto założenie, że *końcowy steganogram*, którego właściwości będzie dalej badać, to *wideo* utworzone poprzez nagranie filmu przez *odbiornik analogowy*, np. kamerę smartfona, na którym widoczne jest odtworzenie na *nadajniku analogowym*, np. ekranie telewizora lub monitora, oryginalnego *wideo* będącego *oryginalnym steganogramem* wysłanym przez *nadawcę*.

W *scenariuszu zniekształceń analogowych nadawca* może od razu wykluczyć pewne rodzaje technik *steganograficznych*, za pomocą których na pewno nie uda się przekazać odbiorcy żadnego *komunikatu*.

Wykluczone są wszystkie techniki *steganografii nadmiarowej*, gdyż *odbiorca* nie będzie miał dostępu do żadnych *metadanych*[125] *oryginalnego steganogramu*, w tym np. *nagłówków*[126] oryginalnego pliku *wideo*.

Zastosowanie przez *nadawcę* technik *steganograficznych* wykorzystujących mechanizmy *kodowania międzyklatkowego* może okazać się bardzo utrudnione lub wręcz nieskuteczne, gdyż w *końcowym steganogramie* utworzonym przez *odbiorcę* parametry wykorzystywane do tego typu kodowań będą stworzone i wyliczone na nowo. Nawet podstawowe techniczne i wizualne parametry *wideo* zarejestrowanego przez *odbiorcę* mogą znacznie różnić się od *oryginalnego steganogramu*. Mogą to być różnice w *rozdzielczości*[127]

---

[125] **Metadane** - dane opisujące właściwości i strukturę innych danych, takie jak kontekst, pochodzenie lub format, umożliwiające organizację, wyszukiwanie i zarządzanie *informacjami*.

[126] **Nagłówki** - *metadane* opisujące strukturę i format pliku cyfrowego lub pakietu danych, np. typ danych i rozmiar.

[127] **Rozdzielczość wideo** - liczba pikseli w poziomie i pionie wpływająca na jakość i ostrość obrazu, wyrażana jako szerokość x wysokość, np. 1920x1080 - Full HD lub 3840x2160 - 4K. Im większa *rozdzielczość wideo* tym bardziej szczegółowy jest obraz *wideo*, ale tym większe jest równocześnie zapotrzebowanie na przepustowość *kanału* oraz moc obliczeniową do przetwarzania takiego *wideo*.

wideo, liczbie klatek na sekundę *fps*[128], umiejscowieniu *końcowego steganogramu* względem *rozdzielczości* całego ekranu oraz ogólnych parametrach poszczególnych pikseli lub obszarów obrazu, takich jak kolor i jasność. Istotna część tych różnic pomiędzy *oryginalnym steganogramem* i *końcowym steganogramem* wynika z samego faktu zastosowania w komunikacji pomiędzy *nadawcą* i *odbiorcą* dodatkowego *nadajnika analogowego* i *odbiornika analogowego*, a wielkość tych różnic zależy bezpośrednio od jakości przetwarzania przez nie sygnałów - im gorsza jakość przetwarzania, tym bardziej *końcowy steganogram* będzie różnił się od *oryginalnego steganogramu*.

W *scenariuszu zniekształceń analogowych* trzeba przedefiniować podstawowy paradygmat *steganologii*, który polega na ukryciu samego faktu przesyłania tajnego *komunikatu* między *nadawcą* a *odbiorcą*. W *podstawowym scenariuszu steganologicznym* założenie to oznaczało, iż *steganogram* w postaci cyfrowego pliku będzie *odporny* na techniki *steganoanalizy* ukierunkowanej na analizy cech cyfrowych *nośników*. Jednak w omawianym scenariuszu, gdzie kluczowym elementem jest *kanał analogowy,* głównym celem staje się ukrycie zakodowanego *komunikatu* w taki sposób, aby podczas jego transmisji w *kanale analogowym* nie można było stwierdzić faktu jego istnienia. Innymi słowy, transmisja powinna być prowadzona w taki sposób, aby nie była możliwa detekcja komunikatu przez ludzkie oko w pełnym zakresie *kanału analogowego*, od *nadajnika analogowego* do *odbiornika analogowego*. Jest to szczególnie istotne w przypadku transmisji przez analogowe media, gdzie ukrycie *komunikatu* może być znacznie trudniejsze niż w przypadku transmisji w pełni cyfrowych. W analizowanym scenariuszu priorytetem jest dostosowanie metod *steganograficznych* do warunków analogowych, z naciskiem na ukrycie *komunikatu* w taki sposób, aby nie był on wykrywalny dla ludzkich zmysłów. Wiązać się to może z koniecznością opracowania technik *steganograficznych*, które będą tworzyły *steganogramy* o dużo większej *odporności*, ale równocześnie z akceptacją mniejszej *pojemności* oraz mniejszej *niewykrywalności* przez automatyczne techniki *steganoanalizy*, niż w przypadku *podstawowego scenariusza steganologicznego*.

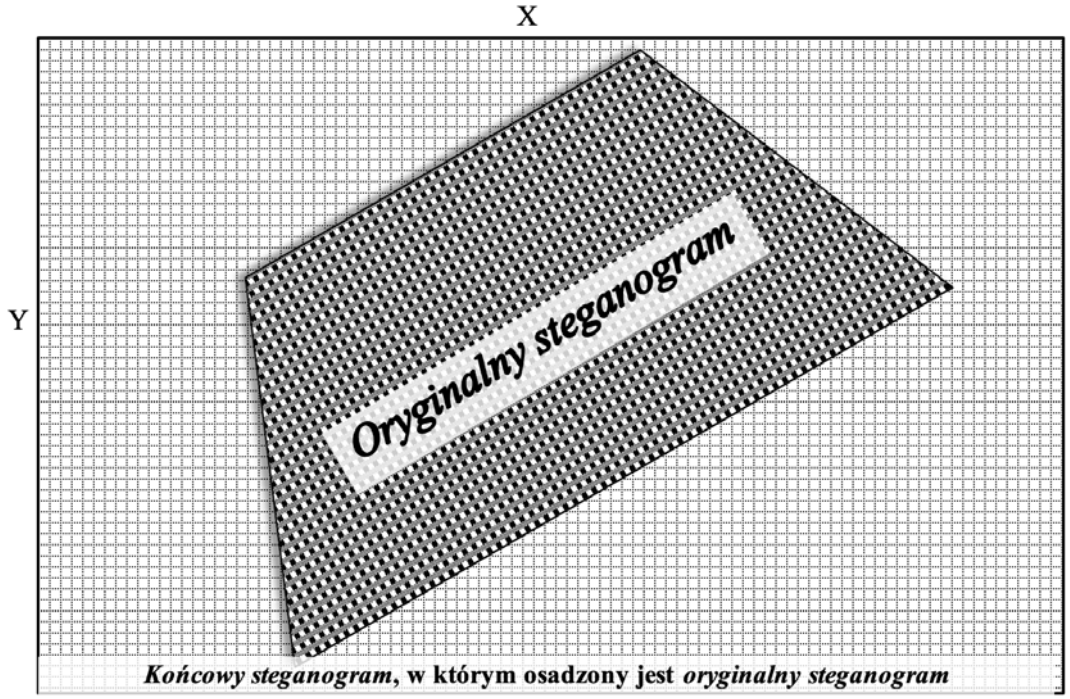
---

[128] **FPS (frames per second)** - liczba *klatek* wyświetlanych na sekundę, wpływająca na płynność, jakość i percepcję wizualną *wideo*. Mieści się w zakresie od 24 fps do 120 fps i jest dobierana tak, aby widz miał wrażenie oglądania ciągłego, płynnego *wideo*, np. 24 fps używany jest w filmach kinowych, 30 fps w internecie, a 60 fps w transmisjach sportowych.

Problemy i ograniczenia związane ze *zniekształceniami analogowymi* transmisji między *nadawcą* a *odbiorcą* można modelować poprzez przyjęcie następujących założeń dotyczących *końcowego steganogramu,* do którego mają dostęp zarówno *odbiorca*, jak i potencjalnie *podsłuchujący*:

1. *Końcowy steganogram* ma formę *wideo* o rozdzielczości X×Y pikseli w przestrzeni kolorów *RGB*.
2. Zakłada się, że *odbiorca* nagrywając *wideo* mógł skierować aparat na źródło wyświetlające *oryginalny steganogram* z pewnej odległości, pod jakimś kątem i w określonych warunkach otoczenia i oświetlenia. W związku z tym zakłada się, że *oryginalny steganogram* znajduje się w niedookreślonym miejscu wewnątrz *końcowego steganogramu*, a dodatkowo może być on zakłócony i przekształcony, np. zaszumiony, przesunięty, obrócony, poddany rzutowi perspektywicznemu.
3. Informacje o dokładnym sposobie umieszczenia *oryginalnego steganogramu* wewnątrz *końcowego steganogramu* oraz o zakłóceniach i przekształceniach, jakim został on poddany, nie są apriorycznie znane ani *odbiorcy* ani *podsłuchującemu*.

Na rysunku 41 schematycznie przedstawiono model *końcowego steganogramu*, który będzie przedmiotem dalszych badań w niniejszej pracy.



Rysunek 41 - Schemat osadzania *oryginalnego steganogramu* w *końcowym steganogram*ie
Źródło: *Opracowanie własne*

Przykład 20 - steganogram poddany wielokrotnym zniekształceniom

Na rysunku 42 przedstawiono przykład osadzenia zniekształconego przekształceniami geometrycznymi oraz *szumem Gaussa oryginalnego steganogramu* na tle zdjęcia Horseshoe Bend[129] tworzącego w ten sposób *końcowy steganogram*.

Rysunek 42 - Przykład: *oryginalny steganogram* osadzony w *końcowym steganogramie*
Źródło: *Opracowanie własne*

W tak zniekształconym *oryginalnym steganogramie,* dodatkowo osadzonym w innym obiekcie, żadna prosta technika *steganograficzna* nie jest w stanie zakodować *komunikatu* tak, aby *odbiorca* był ją w stanie skutecznie zdekodować.

Wartości poszczególnych wskaźników dla tego przykładowego *końcowego steganogramu* wynoszą: MSE = 6459,45; PSNR = 10,03; SSIM = 0,06; BER = 0,52.

Koniec przykładu 20

[129] **Horseshoe Bend** - malownicze zakole rzeki Kolorado w Arizonie w USA, przypominające kształtem podkowę. Powstało na skutek erozji rzeki, która wyrzeźbiła skały przez tysiące lat w imponujący klif wznoszącego się 300 metrów nad wodą. Źródło: https://www.nps.gov/glca/planyourvisit/horseshoe-bend.htm.

3.3 Postawienie tezy badawczej

W niniejszej pracy postawiona i poddana weryfikacji jest następująca teza badawcza: Możliwe jest opracowanie nowej metody *steganografii wideo* spełniającej jednocześnie dwa warunki:

1) Metoda będzie charakteryzowała się tak dużą *odpornością*, że nawet w przypadku *zniekształceń* analogowych oryginalnego *steganogramu wideo* polegających na jednoczesnych przekształceniach geometrycznych, zaszumieniu, transkodowaniu i na końcu utworzeniu przez *odbiorcę* nowego pliku *steganogramu wideo* - możliwe będzie przekazanie od *nadawcy* do *odbiorcy* tajnego *komunikatu*.
2) Metoda będzie cechowała się na tyle dużą *niewykrywalnością* podczas przechodzenia przez *kanał analogowy*, że fakt ukrycia w *steganogramie komunikatu* nie będzie możliwy do odkrycia przy pomocy ludzkich zmysłów, w tym przypadku wzroku.

4 RAI - metoda steganografii wideo odporna na zniekształcenia analogowe

4.1 Proponowana metoda steganografii RAI

W celu weryfikacji postawionej tezy badawczej zaproponowano nową autorską metodę *steganografii wideo*, którą nazwano *RAI (Robust Adaptive Incremental)*[130]. Jak zostało dalej wykazane podczas przeprowadzania badań nad właściwościami zaproponowanej metody, jest to technika *steganografii wideo* charakteryzująca się dużą *odpornością* i *niewykrywalnością* przez ludzkie oko.

Metoda *RAI* czerpie wiele inspiracji z istniejących technik *steganograficznych*, starając się połączyć najkorzystniejsze elementy tych technik pod kątem wykorzystania w rozwiązywanym problemie, w tym w szczególności wykorzystuje elementy:

1. technik *LSB* działających w *domenie przestrzennej* - np. w zakresie, w którym algorytm *RAI* dokonuje modyfikacji wartości kolorów wybranych pikseli *klatek* o jak najmniejszą wartość, co oznacza dokonywanie zmian w jak najmniejszej liczbie najmniej znaczących *bitów* poszczególnych pikseli,
2. techniki *Patchwork* - *np.* w zakresie, w którym algorytm *RAI* dokonuje kwantyzacji blokowej *klatek* i zmienia parametry pikseli blokami, kodując poszczególne *bity komunikatu* w wybranych blokach,
3. *techniki PVD* - np. w zakresie, w jakim algorytm *RAI* adaptacyjnie wyszukuje takie pary pikseli, których zmiana jest możliwa ze względu na niezbyt dużą odległość kolorystyczną pomiędzy nimi.

Metodę *RAI* można sklasyfikować jednocześnie jako technikę *steganografii wideo* (co jednocześnie oznacza, że jest to technika *steganografii iteracyjnej*), steganografii *przestrzennej* oraz *steganografii adaptacyjnej.*

[130] **RAI (Robust Adaptive Incremental)** - zaproponowana w niniejszej pracy nowa technika *steganografii wideo*.

4.1.1 Podstawy modelu formalnego metody RAI

Jedno z pierwszych podejść do stworzenia matematycznego modelu teoretycznego opisującego pewną klasę systemów *steganograficznych* to artykuł opublikowany w 1998 roku przez C. Cachina *"An Information-Theoretic Model for Steganography" (Cachin, 1998)*. W artykule tym zadanie polegające na rozróżnieniu między *nośnikiem* a *steganogramem*, jest interpretowane jako problem testowania pewnych statystycznych hipotez. Cachin kwantyfikuje bezpieczeństwo systemu *steganograficznego* za pomocą względnej *entropii* między rozkładami prawdopodobieństwa *nośnika* i *steganogramu*. Model ten stanowi istotny wkład w ocenę i projektowanie bezpiecznych systemów *steganograficznych* oferując matematyczne podejście do analizy ich *niewykrywalności*.

W dalszej części pracy przedstawiony zostanie autorski model formalny *scenariusza zniekształceń analogowych* oraz proponowanej metody *steganografii wideo RAI*.

W uproszczeniu każde *wideo* można przedstawić jako sekwencję następujących po sobie obrazów, zwanych *klatkami*. W praktyce, w zależności od zastosowanego *kodeka*, *wideo* nie jest zapisywane jako prosta sekwencja obrazów. Stosowane są bardziej zaawansowane algorytmy, które np. uwzględniają zmiany pomiędzy kolejnymi *klatkami*. Zamiast zapisywania wszystkich *klatek* po kolei uwzględniają one np. jedynie różnice między kolejnymi obrazami, co znacząco redukuje ilość danych potrzebnych do zapisu *wideo*.

Na potrzeby niniejszej pracy przyjęto najprostszy model *wideo* - bez mechanizmów zaawansowanej kompresji pomiędzy poszczególnymi *klatkami*. Odpowiada on rzeczywistemu formatowi *wideo MJPEG*. Uwzględnienie ewentualnego wpływu zastosowania bardziej złożonych modeli *wideo* na przyjęte założenia oraz uzyskiwane wyniki zaproponowano jako jeden z możliwych kierunków dalszych badań.

Ponieważ *nośnik* i *steganogram* są plikami *wideo*, można wprost zaadoptować przedstawiony w pracy *(Pery i Waszkowski, Analyzing natural pixel color variations in consecutive video frames to enhance spatial domain video steganography, 2024)* formalny model pliku *wideo.*

Plikiem *wideo* v nazywa się Z-elementowy wektor *klatek* zdefiniowany wzorem (12).

$$v = (v_1, v_2, \dots, v_i, \dots, v_Z) \tag{12}$$

$$Z \in \mathbb{N}, v \in \mathcal{V}$$

gdzie: v_i – i-ta *klatka wideo* v,
v – Z-elementowy wektor *klatek,*
$\mathcal{V}$ – zbiór wszystkich możliwych plików *wideo.*

Przy czym wektor v^i definiuje się jako wektor zawierający i pierwszych elementów wektora v, zgodnie ze wzorem (13).

$$\bigwedge_{i \in \{1 \dots Z\}} v^i = \begin{cases} (v_1) & , i = 1 \\ v^{i-1} \oplus v_i & , i \in \{2 \dots Z\} \end{cases} \tag{13}$$

gdzie: v_i – i-ta *klatka wideo* v,
v^i – i-elementowy wektor *klatek.*

Klatką v_i nazywa się macierz 3-elementowych wektorów reprezentujących piksele w przestrzeni *RGB*, zgodnie ze wzorami (14) i (15).

$$v_i = \begin{bmatrix} p^i_{0,0} & \dots & p^i_{X-1,0} \\ \dots & p^i_{x,y} & \dots \\ p^i_{0,Y-1} & \dots & p^i_{X-1,Y-1} \end{bmatrix} \tag{14}$$

$$X, Y \in N$$

$$\bigwedge_{i \in \{1 \dots Z\}} v_i \in \{0 \dots 255\}^{3^{X \times Y}} \tag{15}$$

gdzie: $p^i_{x,y}$ – piksel o współrzędnych (x, y) i-tej *klatki* v_i,
$X \times Y$ – wymiary *klatek*,
$\{0 \dots 255\}^{3^{X \times Y}}$ – zbiór wszystkich możliwych klatek, które są macierzami o wymiarach $X \times Y$ 3-elementowych wektorów liczb z zakresu $\{0 \dots 255\}$.

Pikselem $p_{x,y}^{i}$ nazywa się 3-elementowy wektor reprezentujący w przestrzeni *RGB* trzy kolory podstawowe (czerwony, zielony i niebieski), zgodnie ze wzorami (16) i (17).

$$p_{x,y}^{i} = \left(red_{x,y}^{i}, green_{x,y}^{i}, blue_{x,y}^{i}\right) \tag{16}$$

$$\bigwedge_{i\in\{1\ldots Z\}} \bigwedge_{\substack{x\in\{0\ldots X-1\}\\ y\in\{0\ldots Y-1\}}} \begin{cases} red_{x,y}^{i} & \in \{0\ldots 255\} \\ green_{x,y}^{i} & \in \{0\ldots 255\} \\ blue_{x,y}^{i} & \in \{0\ldots 255\} \end{cases} \tag{17}$$

gdzie: $red_{x,y}^{i}, green_{x,y}^{i}, blue_{x,y}^{i}$ – kolory piksela $p_{x,y}^{i}$.

W pracy *(Pery i Waszkowski, A Model and Quantitative Framework for Evaluating Iterative Steganography, 2024)* przedstawiono formalny model *steganografii iteracyjnej*, w tym *nośnika* i *steganogramu* iteracyjnego. Ze względu na fakt, że obiektem zainteresowań w ramach niniejszej pracy jest *steganografia wideo*, a *wideo* jest *nośnikiem* iteracyjnym, zaproponowane w tamtym opracowaniu modele odpowiednio wykorzystano i zaadoptowano na potrzeby niniejszej pracy. W przypadku plików *wideo* kolejnymi iteracjami są poszczególne *klatki wideo*.

Nośnikiem iteracyjnym c będzie nazywać się Z-elementowy wektor iteracji, zgodnie ze wzorami (18) i (19).

$$c = (c_1, c_2, \ldots, c_i, \ldots, c_Z) \tag{18}$$

$$Z \in \mathbb{N}, c \in \mathcal{C}$$

$$\bigwedge_{i\in\{1\ldots Z\}} c_i \in \mathcal{I}, \qquad \mathcal{I} = \{0,1\}^{*} \tag{19}$$

gdzie: c_i – i-ta iteracja *nośnika* c,
$\mathcal{I}$ – zbiór wszystkich możliwych iteracji *nośnika*,
Z – liczba iteracji *nośnika* c,
$\mathcal{C}$ – zbiór wszystkich możliwych *nośników*.

Przy czym nośnik iteracyjny c^i definiuje się jako wektor zawierający i pierwszych elementów wektora c, zgodnie ze wzorami (20) i (21).

$$\bigwedge_{i\in\{1\ldots Z\}} c^i = \begin{cases} (c_1) & ,i = 1 \\ c^{i-1} \oplus c_i & ,i \in \{2\ldots Z\} \end{cases} \tag{20}$$

$$\bigwedge_{i\in\{1\ldots Z\}} c^i \in \mathcal{C}\,, \qquad \mathcal{C} = \mathcal{I}^* \tag{21}$$

gdzie: c_i – *i*-ta iteracja *nośnika* c,
c^i – *i*-elementowy wektor iteracji.

Komunikatem m będzie nazywać się L-elementowy wektor *bitów*, zgodnie ze wzorami (22) i (23).

$$m = (m_1, m_2, \ldots, m_k, \ldots, m_L) \tag{22}$$

$$L \in \mathbb{N}, m \in \mathcal{M},\ \mathcal{M} = \{0,1\}^*$$

$$\bigwedge_{k\in\{1\ldots Z\}} m_k \in \{0,1\} \tag{23}$$

gdzie: L – liczba bitów *komunikatu* m (długość *komunikatu*),
m_k – *k*-ty bit *komunikatu* m,
$\mathcal{M}$ – zbiór wszystkich możliwych komunikatów.

Wskaźniki $m(1)$ *i* $m(0)$ definiuje się odpowiednio jako liczby bitów jedynkowych i zerowych *komunikatu* m, zgodnie ze wzorami (24), (25) i (26).

$$m(1) = \sum_{k=1}^{L} 1_{\{m_k=1\}} \tag{24}$$

$$m(0) = \sum_{k=1}^{L} 1_{\{m_k=0\}} \tag{25}$$

$$m(0) + m(1) = L \tag{26}$$

gdzie: L – liczba bitów *komunikatu* m,
m_k – *k*-ty bit *komunikatu* m.

Steganogramem iteracyjnym s, analogicznie do *nośnika* iteracyjnego c, będzie nazywać się Z-elementowy wektor iteracji, zgodnie ze wzorami (27) i (28).

$$s = (s_1, s_2, \ldots, s_i, \ldots, s_Z) \tag{27}$$

$$Z \in \mathbb{N}, s \in \mathcal{S}$$

$$\bigwedge_{i \in \{1 \ldots Z\}} s_i \in \mathcal{I}, \qquad \mathcal{I} = \{0,1\}^* \tag{28}$$

gdzie: s_i – *i*-ta iteracja *steganogramu* s,
$\mathcal{I}$ – zbiór wszystkich możliwych iteracji *steganogramu*,
Z – liczba iteracji *steganogramu* s,
$\mathcal{S}$ – zbiór wszystkich możliwych *steganogramów*.

Ze zbioru wszystkich możliwych indeksów dla *nośnika* oraz jednocześnie dla *steganogramu* wybierzmy podzbiór, który nazwiemy indeksami kodującymi, a który będzie wyznaczał zbiór tych iteracji, które zostaną poddane kodowaniu *steganograficznemu (Pery i Waszkowski, A Model and Quantitative Framework for Evaluating Iterative Steganography, 2024)*, zgodnie ze wzorem (29).

$$\mathrm{I} \subset \{2 \ldots Z\}, \qquad \mathrm{I} \in 2^{\{2 \ldots Z\}} \tag{29}$$

gdzie: I – zbiór indeksów kodujących.

Zgodnie z założeniem dotyczącym kodowania *komunikatów* w ste*ganografii iteracyjnej* w każdej iteracji *algorytmu kodującego* kodowany jest cały *komunikat* m.

Pomocniczą funkcję kodującą f', która przyporządkowuje *nośnikowi* oraz *komunikatowi* pojedynczą zakodowaną *klatkę steganogramu*, definiuje się zgodnie ze wzorami (30) i (31).

$$f' : \mathcal{I} \times \mathcal{M} \to \mathcal{I} \tag{30}$$

$$\bigwedge_{i \in \{1 \ldots Z\}} s_i := f'^{(c_i, m)} \tag{31}$$

gdzie: c_i – *i*-ta iteracja *nośnika*,
s_i – *i*-ta iteracja *steganogramu*,
m – kodowany *komunikat*,
Z – liczba kroków kodujących.

Iteracyjną funkcję f kodującą *komunikat* w *nośniku* poprzez utworzenie *steganogramu* definiuje się zgodnie ze wzorami (32), (33) i (34).

$$f: \mathcal{C} \times \mathcal{M} \to \mathcal{S} \tag{32}$$

$$\bigwedge_{i \in \{1 \ldots Z\}} s^i := f(c^i, m) \tag{33}$$

$$\bigwedge_{i \in \{1 \ldots Z\}} f(c^i, m) = \begin{cases} (c_1) & , i = 1 \\ f(c^{i-1}, m) \oplus c_i, & , i \notin I \\ f(c^{i-1}, m) \oplus f'^{(c_i, m)} & , i \in I \end{cases} \tag{34}$$

gdzie: c_i – *i*-ta iteracja *nośnika*,
s^i – *steganogram* utworzony przez zakodowanie *komunikatu* m w *nośniku* c^i,
m – *komunikat* kodowany przez funkcję f w *nośniku* c^i,
I – zbiór indeksów kodujących,
Z – liczba kroków kodowania.

Iteracyjna funkcja dekodująca f^{-1} jest zdefiniowana jako przyporządkowanie *steganogramu* s^i do zdekodowanego *komunikatu* m^{-1} zgodnie z wzorami (35) i (36).

$$f^{-1}: \mathcal{S} \to \mathcal{M} \tag{35}$$

$$m^{-1} := f^{-1}(s^i) \tag{36}$$

$$m^{-1} \in \mathcal{M}$$

gdzie: s^i –steganogram utworzony przez zakodowanie komunikatu m w nośniku c^i,
m^{-1} – komunikat zdekodowany przez funkcję f^{-1} ze steganogramu s^i.

Celem funkcji dekodującej jest odzyskanie *informacji* zakodowanej w oryginalnym *komunikacie* m. Jeśli zdekodowany *komunikat* m^{-1} jest identyczny z oryginalnym *komunikatem* m, cel ten zostaje w pełni osiągnięty. W przeciwnym razie, jeśli $m^{-1} \neq m$, możliwość poprawnego odczytania informacji z zdekodowanego *komunikatu* zależy głównie od dwóch czynników: (1) ilości zachowanej *informacji* w komunikacie m^{-1}, którą można określić na przykład za pomocą miary odległości między zdekodowanym i oryginalnym *komunikatem*; oraz (2) właściwości zastosowanej techniki *steganograficznej*, w tym stopnia *redundancji* zastosowanego podczas kodowania *informacji* w *steganogramie*.

W pracy *(Pery i Waszkowski, Analyzing natural pixel color variations in consecutive video frames to enhance spatial domain video steganography, 2024)* zdefiniowano funkcję $diff$, która przyporządkowuje wybranej *klatce wideo* i indeksowi iteracji macierz zawierającą wektory różnic kolorów piksela z danej *klatki* i *klatki* poprzedniej, zgodnie ze wzorami (37) i (38).

$$diff: \mathcal{V} \times \{2..Z\} \rightarrow \{-255 \dots 255\}^{X \times Y} \tag{37}$$

$$diff(v,i) := \begin{bmatrix} p^i_{0,0} - p^{i-1}_{0,0} & \cdots & p^i_{X-1,0} - p^{i-1}_{X-1,0} \\ \cdots & p^i_{x,y} - p^{i-1}_{x,y} & \cdots \\ p^i_{0,Y-1} - p^{i-1}_{0,Y-1} & \cdots & p^i_{X-1,Y-1} - p^{i-1}_{X-1,Y-1} \end{bmatrix} \tag{38}$$

$$v \in \mathcal{V}, i \in \{2 \dots Z\}$$

gdzie: v – *wideo*,
i – indeks *klatki wideo*,
$p^i_{x,y}$ – piksel o współrzędnych (x, y) w *i-tej klatce*,
$p^{i-1}_{x,y}$ – piksel o współrzędnych (x, y) w *(i-1)-tej klatce.*

Poziomem kodowania[131] l nazywa się liczbę naturalną z zakresu $\{1 \dots lmax\}$ określającą wielkość zmian dokonywanych w poszczególnych pikselach *klatek steganogramu* podczas kodowania *komunikatu*, zgodnie ze wzorem (39). *Poziom kodowania* 1 (jeden) oznacza najmniejszą ingerencję w nośnik, a im wyższy *poziom kodowania*, tym *nośnik* będzie mocniej zmieniony poprzez coraz większe zaburzenie kolorów pikseli.

$$l \in \{1 \dots lmax\} \tag{39}$$

$$lmax \in \mathbb{N}$$

gdzie: l – poziom kodowania.

[131] **Poziom kodowania** - parametr *podstawowego algorytmu kodującego c-RAI* określający wielkość zmian dokonywanych w pikselach kodowanych *klatek steganogramu*. Im większy *poziom kodowania*, tym zmiany są bardziej znaczące.

Maską[132] $mask(l)$ dla *nośnika* c, *poziomu kodowania* l, indeksu *klatki* oraz piksela (x, y) nazywa się krotkę zdefiniowaną zgodnie ze wzorami (40) i (41).

$$\bigwedge_{l\in\{1\ldots lmax\}} \bigwedge_{\substack{i\in\{2\ldots Z\}\\ x\in<0\ldots X)\\ y\in<0\ldots Y)}} mask(l)^i_{x,y} = \left(mask(l)^i_{x,y}[0], mask(l)^i_{x,y}[1], mask(l)^i_{x,y}[2]\right) \tag{40}$$

$$\bigwedge_{color\in\{0\ldots 2\}} mask(l)^i_{x,y}[color] = \begin{cases} 1 & gdy\ diff(c,i)_{x,y}[color] \in \langle -l, l\rangle \\ 2 & gdy\ diff(c,i)_{x,y}[color] < l \\ 4 & gdy\ diff(c,i)_{x,y}[color] > l \end{cases} \tag{41}$$

$$v \in \mathcal{V}, i \in \{2\ldots Z\}$$

gdzie: l – poziom kodowania,
$mask(l)^i_{x,y}[color]$, – maska dla koloru *color* określająca, jaka różnica występuje na tym kolorze pomiędzy pikselami *klatki i*-tej oraz (*i-1*)-tej.

Maską $mask1$ kodującą *bit* 1 nazywa się krotkę zdefiniowaną zgodnie ze wzorami (42) i (43).

$$mask1 = (mask1[0], mask1[1], mask1[2]) \tag{42}$$

$$\bigwedge_{color\in\{0\ldots 2\}} mask1[color] \in \{1,2,4\} \tag{43}$$

Przykładową *maską* kodującą bit 1 może być np. *mask1* = $(4, 2, 4)$, co ma interpretację następującą: na danym pikselu w danej *klatce* jest zapisany *bit* 1, jeżeli jednocześnie zajdą następujące warunki: wartość koloru czerwonego tego piksela jest większa od koloru tego samego piksela w poprzedniej ramce, wartość koloru zielonego jest mniejsza, a wartość koloru niebieskiego jest większa.

[132] **Maska** - trójka symboli "+- " lub liczb {1,2,4} opisująca różnicę pomiędzy składowymi kolorystycznymi dwóch pikseli A - B. Wartości " " lub 1 oznaczają, że piksele A i B mają ten sam składowy kolor, wartości "-" lub 2 oznaczają, że piksel A ma mniejszą daną składową kolorystyczną niż piksel B, a wartość "+" lub 4 oznacza, że piksel B ma mniejszą składową kolorystyczną niż piksel A.

Maską $mask0$ kodującą bit 0 nazywa się krotkę zdefiniowaną zgodnie ze wzorami (44) i (45).

$$mask0 = (mask0[0], mask0[1], mask0[2]) \tag{44}$$

$$\bigwedge_{color \in \{0\ldots2\}} mask0[color] = \begin{cases} 1 & gdy\ mask1[color] = 1 \\ 2 & gdy\ mask1[color] = 4 \\ 4 & gdy\ mask1[color] = 2 \end{cases} \tag{45}$$

Przykładową *maską* kodującą bit 0 może być *mask0* $=(2, 4, 2)$, co ma z kolei interpretację następującą: na danym pikselu w danej *klatce* jest zapisany *bit* 0, jeżeli jednocześnie zajdą następujące warunki: wartość koloru czerwonego tego piksela jest mniejsza od koloru tego samego piksela w poprzedniej ramce, wartość koloru zielonego jest większa, a wartość koloru niebieskiego jest mniejsza. Istotnym założeniem, jest pewnego typu *"odwrotność"* $mask1$ oraz $mask0$. Na pozycji koloru, dla którego w $mask1$ jest wartość 4, w $mask0$ jest wartość 2 (i odwrotnie). Jest to związane w sposób oczywisty z faktem, że jeżeli do kodowania *bitu* 1 używana jest różnica kolorystyczna dodatnia, to do kodowania *bitu* przeciwnego (*bitu* 0) musi być użyta wartość ujemna. Wartości 1, oznaczające brak różnic kolorystycznych pomiędzy pikselami, mogą występować na tych samych pozycjach w obu maskach, gdyż nie są wykorzystywane do kodowania *komunikatu*.

Maską kodującą[133] kolejne *bity komunikatu* $cmask(bit)$ nazywa się krotkę zdefiniowaną zgodnie ze wzorami (46) i (47).

$$cmask(bit) = \begin{cases} mask1 & gdy\ bit = 1 \\ mask0 & gdy\ bit = 0 \end{cases} \tag{46}$$

$$cmask(bit) = (cmask(bit)[0], cmask(bit)[1], cmask(bit)[2]) \tag{47}$$

[133] **Maska kodująca** - używana w *podstawowym algorytmie kodującym RAI maska* kodująca bit 0 lub *maska* kodująca bit 1 *komunikatu*.

4.1.2 Algorytm kodujący c-RAI

W niniejszej rozdziale zdefiniowano *podstawowy algorytm kodujący c-RAI*[134], który koduje *komunikat* w *steganogramie wideo* metodą *RAI*.

Podstawowy algorytm kodujący c-RAI posiada dwa założenia:

1. *poziom kodowania* jest stały,
2. *krok kodowania*[135] jest stały.

4.1.2.1 Model formalny

Obszarem kodowania $cblocks$ nazywa się wektor bloków (obszarów) będących rozłącznymi podzbiorami pikseli *klatek steganogramu,* zgodnie ze wzorami (48) i (49).

$$cblocks = (cb_1, cb_2, \dots, cb_k, \dots, cb_L) \tag{48}$$

$$\bigwedge_{k\in\{1\dots L\}} cb_k \in \left((cx_k^1, cy_k^1), (cx_k^1, cy_k^2), (cx_k^2, cy_k^1), (cx_k^2, cy_k^2)\right) \tag{49}$$

$$cx_k^1 \in \langle 0 \dots X), cx_k^2 \in \langle 0 \dots X), cy_k^1 \in \langle 0 \dots Y), cy_k^2 \in \langle 0 \dots Y)$$

gdzie: (cx_k^1, cy_k^1) – współrzędne lewego górnego rogu k-tego bloku obszaru kodowania,
(cx_k^1, cy_k^2) – współrzędne prawego górnego rogu k-tego bloku obszaru kodowania,
(cx_k^2, cy_k^1) – współrzędne lewego dolnego rogu k-tego bloku obszaru kodowania,
(cx_k^2, cy_k^2) – współrzędne prawego dolnego rogu k-tego bloku obszaru kodowania.

Zakłada się, że liczba bloków obszaru kodowania $cblocks$ jest taka sama, jak długość *komunikatu* L, a każdy blok odpowiada jednemu *bitowi* kodowanego *komunikatu*.

[134] **Podstawowy algorytm kodujący c-RAI** - algorytm kodujący *komunikat* w *steganogramie wideo* metodą *RAI*.

[135] **Krok kodowania** - parametr *podstawowego algorytmu kodującego RAI* określający, w co której *klatce* ma być kodowany *komunikat*.

Funkcją $block$ nazywa się przyporządkowanie każdemu pikselowi o współrzędnych (x, y) numeru bloku obszaru kodowania, do którego należy ten piksel, zgodnie ze wzorami (50) i (51).

$$block\colon([X] \times [Y]) \to \{1 \dots L\} \tag{50}$$

$$\bigwedge_{\substack{k\in\{1\dots L\}\\ x\in<0\dots X)\\ y\in<0\dots Y)}} block\big((x,y)\big) = k \quad \Leftrightarrow \quad (x \in \langle cx_k^1, cx_k^2\rangle \wedge y \in \langle cy_k^1, cy_k^2\rangle) \tag{51}$$

W pracy *(Pery i Waszkowski, A Model and Quantitative Framework for Evaluating Iterative Steganography, 2024)* zaproponowano definicję funkcji kodującej. W przypadku niniejszej pracy pomocnicza funkcja kodująca f' posiada dodatkowy argument - *poziom kodowania* i ostatecznie jej kształt zdefiniowany jest wzorami (52), (53), (54), (55) i (56).

$$f'\colon \mathcal{I} \times \mathcal{M} \times \{1 \dots lmax\} \to \mathcal{I} \tag{52}$$

$$\bigwedge_{i\in\{1\dots Z\}} s_i := f'^{(c_i, m, l)} \tag{53}$$

$$\bigwedge_{i\in\{1\dots Z\}} s_i := \begin{bmatrix} {p_{coded}}_{0,0}^{\,i} & \cdots & {p_{coded}}_{X-1,0}^{\,i} \\ \cdots & {p_{coded}}_{x,y}^{\,i} & \cdots \\ {p_{coded}}_{0,Y-1}^{\,i} & \cdots & {p_{coded}}_{X-1,Y-1}^{\,i} \end{bmatrix} \tag{54}$$

$$\bigwedge_{\substack{i\in\{1\dots Z\}\\ x\in<0\dots X)\\ y\in<0\dots Y)}} {p_{coded}}_{x,y}^{\,i} = \left[{p_{coded}}_{x,y}^{\,i}[0], {p_{coded}}_{x,y}^{\,i}[1], {p_{coded}}_{x,y}^{\,i}[2]\right] \tag{55}$$

$${p_{coded}}_{x,y}^{\,i}[color] = \tag{56}$$
$$\bigwedge_{color\in\{0\dots 2\}} p_{x,y}^{i} + \begin{cases} l & gdy\ mask(l)_{x,y}^{i}[color] = 1 \wedge cmask\left(m_{block((x,y))}\right)[\text{color}] = 4 \\ -l & gdy\ mask(l)_{x,y}^{i}[color] = 1 \wedge cmask\left(m_{block((x,y))}\right)[\text{color}] = 2 \end{cases}$$

gdzie:
c_i – *i*-ta iteracja *nośnika*,
s_i – *i*-ta iteracja *steganogramu*,
m – kodowany komunikat,
Z – liczba kroków kodujących,
${p_{coded}}_{x,y}^{\,i}$ – modyfikowany piksel (x, y) w *i*-tej *klatce steganogramu*.

Krokiem kodowania $step$ nazywa się liczbę naturalną z zakresu $\{2 \ldots Z\}$, która określa, co którą *klatkę podstawowy algorytm kodujący RAI* ma kodować *komunikat* w *steganogramie*, zgodnie ze wzorem (57).

$$step \in \{2 \ldots Z\} \tag{57}$$

Przy założeniu, że dla danego algorytmu kodującego i dekodującego dany jest krok kodowania $step$, zdefiniowany w pracy *(Pery i Waszkowski, A Model and Quantitative Framework for Evaluating Iterative Steganography, 2024)*, zbiór indeksów kodujących w niniejszej pracy zdefiniowany jest zgodnie ze wzorem (58).

$$I = \left\{0, step, 2step, \ldots, \left\lfloor \frac{Z}{step} \right\rfloor step\right\} \tag{58}$$

gdzie: $step$ – krok kodowania,
Z – liczba iteracji w nośniku.

Zbiór indeksów kodujących I wyznacza te iteracje - *klatki wideo*, w których będzie kodowany *komunikat*.

Na bazie pierwotnej definicji iteracyjnej funkcji kodującej f definiuje się iteracyjną funkcję $RAIcode$ kodującą *komunikat* w *nośniku* poprzez utworzenie *steganogramu*. W odróżnieniu od pierwotnej iteracyjnej funkcji kodującej f, definicja funkcji $RAIcode$ ma dodatkowy parametr określający poziom kodowania, zgodnie ze wzorami (59), (60) i (61).

$$RAIcode: \mathcal{C} \times \mathcal{M} \times \{1 \ldots lmax\} \rightarrow \mathcal{S} \tag{59}$$

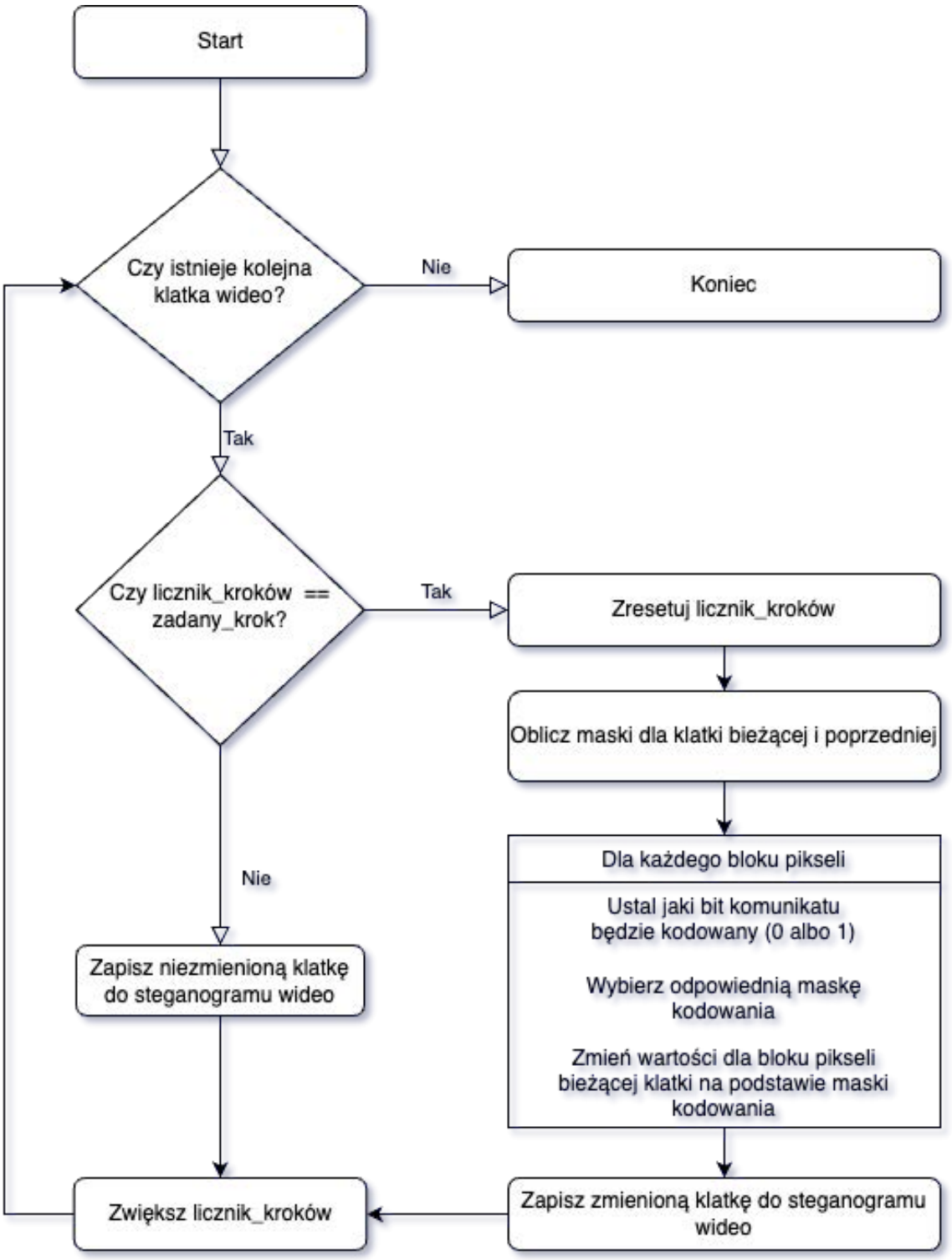
$$\bigwedge_{i \in \{1 \ldots Z\}} s^i := RAIcode(c^i, m, l) \tag{60}$$

$$\bigwedge_{i \in \{1 \ldots Z\}} RAIcode(c^i, m, l) = \begin{cases} (c_1) & , i = 1 \\ RAIcode(c^{i-1}, m, l) \oplus c_i, & , i \notin I \\ RAIcode(c^{i-1}, m, l) \oplus f'^{(c_i, m, l)} & , i \in I \end{cases} \tag{61}$$

gdzie: c_i – *i*-ta iteracja *nośnika*,
s^i – *steganogram* utworzony przez zakodowanie *komunikatu* m w nośniku c^i,
m – komunikat kodowany przez funkcję f w nośniku c^i,
$step$ – krok kodowania,
I – zbiór indeksów kodujących,
Z – liczba kroków kodowania.

4.1.2.2 Schemat blokowy

Na rysunku 43 przedstawiono schemat blokowy *podstawowego algorytmu kodującego c-RAI*.



Rysunek 43 - Schemat blokowy *podstawowego algorytmu kodującego c-RAI*
Źródło: *Opracowanie własne*

4.1.2.3 Implementacja

Kod 14 zawiera implementację *podstawowego algorytmu kodującego c-RAI* bazującą na formalnym modelu funkcji *RAIcode* przedstawionym wcześniej.

```
# Definicja stosowanych w algorytmie RAI masek kodujących
mask_1 = np.array([4, 2, 4], dtype=np.int8)         # maska kodująca bit 1
mask_0 = np.array([2, 4, 2], dtype=np.int8)         # maska kodująca bit 0

# Funkcja przygotowująca podział obrazu na bloki
def get_blocks_list(height, width, number_of_blocks):
    """
    Tworzy listę bloków na potrzeby kwantyzacji obrazu.

    Argumenty:
    height (int): wysokość obrazu
    width (int): szerokość obrazu
    number_of_blocks (int) -- liczba bloków (powiększana do boku kwadratu).

    Wyniki:
    (list): lista współrzędnych wierzchołków bloków
    """

    side_of_square = int(number_of_blocks**0.5)+(number_of_blocks**0.5 % 1 > 0)
    length = min(height, width)
    y_0, x_0 = (height - length) // 2, (width - length) // 2
    size = length / side_of_square

    return [
        (
            (int(y_0 + size * y), int(x_0 + size * x)),
            (int(y_0 + size * (y + 1)), int(x_0 + size * (x + 1)))
        )
        for y in range(side_of_square)
        for x in range(side_of_square) ]

# Podstawowy algorytm kodujący RAI
def RAI_code(cover_video, message, steganogram_video, level, step=3):
    """
    Koduje komunikat za pomocą techniki RAI (Robust Adaptive Incremental).

    Argumenty:
    cover_video (cv2.VideoCapture): nośnik
    message (tablica tekstowa): komunikat - bity tekstowo, np. '00101010'
    steganogram_video (cv2.VideoWriter): wynikowy steganogram
    level (int): poziom kodowania
    step (int): krok kodowania (domyślnie 3)
    """

    height = int(cover_video.get(cv2.CAP_PROP_FRAME_HEIGHT))
    width = int(cover_video.get(cv2.CAP_PROP_FRAME_WIDTH))

    # Przygotowanie do kwantyzacji blokowej
    blocks_list = get_blocks_list(height, width, len(message))

    frame_count, step_count = 0, 0
    ret, frame_prev = cover_video.read()
```

```
while cover_video.isOpened():
    # Przetwarzanie wideo klatka po klatce
    ret, frame_curr = cover_video.read()
    if not ret:
        break

    # Przetwarzane są tylko co 'step'-krotne ramki
    step_count += 1
    if step_count == step:
        step_count = 0

        # Obliczana jest różnica między bieżącą a poprzednią ramką
        diff_frame =frame_curr.astype(np.int16)-frame_prev.astype(np.int16)

        # Obliczane są maski przy uwzględnieniu zadanego poziomu kodowania
        mask_frame = np.ones_like(diff_frame, np.int8)
        mask_frame[diff_frame < -level] = 2
        mask_frame[diff_frame > level] = 4

        # Kodowanie bloków na podstawie komunuikatu
        for i, block in enumerate(blocks_list):
            # Wybierana jest odpowiednia maska do kodowania
            mask=mask_1 if i<len(message) and message[i]=='1' else mask_0
            (y0, x0), (y1, x1) = block

            curr_block = frame_curr[y0:y1, x0:x1].astype(np.int16)
            prev_block = frame_prev[y0:y1, x0:x1].astype(np.int16)
            mask_block = mask_frame[y0:y1, x0:x1]

            # Kodowanie pikseli w zależności od wybranej maski
            for c in range(3):
                condition = (mask_block[:, :, c] == 1)
                adjustment = level if mask[c] == 4 else -level
                curr_block[condition,c]= prev_block[condition,c]+adjustment

            frame_curr[y0:y1, x0:x1] = np.clip(curr_block, 0, 255)

        steganogram_video.write(frame_curr)
        frame_prev = frame_curr.copy()
        frame_count += 1
```

Kod 14 - *Podstawowy algorytm kodujący RAI*
Źródło: *Opracowanie własne*

4.1.3 Algorytm dekodujący d-RAI

W niniejszym rozdziale zdefiniowano *podstawowy algorytm dekodujący d-RAI*[136] służący do dekodowania *komunikatu* zakodowanego *podstawowym algorytmem kodującym c-RAI*.

Podstawowy algorytm dekodujący d-RAI zakłada następującą wiedzę o zastosowanych parametrach *podstawowego algorytmu kodującego c-RAI*:

1. *krok kodowania* jest znany i stały,
2. znane są *maski kodujące,*
3. znany jest numer *klatki*, od którego zaczął się proces kodowania *komunikatu*.

W przypadku *końcowego steganogramu* w *scenariuszu zniekształceń analogowych* założenie o znanej początkowej *klatce* kodowania jest praktycznie niemożliwe do spełnienia. Stworzenie nowego pliku *wideo* z dużym prawdopodobieństwem spowoduje przesunięcie względem *klatki* początkowej *oryginalnego steganogramu*. Dlatego konieczne było zaprojektowanie dodatkowego algorytmu dekodującego, który byłby odporny na przesunięcie *klatki* początkowej. Algorytm ten został nazwany *adaptacyjnym algorytmem dekodujący d-RAI*[137]. Zakłada on mniejszą wiedzę o parametrach *podstawowego algorytmu kodującego c-RAI*, obejmującą następujące założenia:

1. *krok kodowania* jest znany i stały,
2. znane są *maski kodujące.*

Podstawowy algorytm dekodujący d-RAI i *adaptacyjny algorytm dekodujący d-RAI* będą nazywane zamiennie *algorytmem dekodującym d-RAI*[138].

---

[136] **Podstawowy algorytm dekodujący d-RAI** - algorytm dekodujący *komunikat* ze *steganogramu* utworzonego przy pomocy *podstawowego algorytmu kodującego c-RAI*, zakładający pewną wiedzę o zastosowanych parametrach algorytmu kodującego.

[137] **Adaptacyjny algorytm dekodujący d-RAI** - algorytm dekodujący *komunikat* ze *steganogramu* utworzonego przy pomocy *podstawowego algorytmu kodującego c-RAI*, zakładający pewną wiedzę o zastosowanych parametrach algorytmu kodującego, adaptacyjnie dostosowujący się do przesunięcia początkowej *klatki* kodującej w *steganogramie*.

[138] **Algorytm dekodujący d-RAI** - *podstawowy algorytm dekodujący d-RAI* lub *adaptacyjny algorytm dekodujący d-RAI.*

4.1.3.1 Model formalny

Wektor bloków pikseli, będący rozbiorem zbioru pikseli *klatek steganogramu s,* stanowiący obszar dekodowania dla *algorytmu dekodującego d-RAI* nazywa się $dblocks$ i definiuje się zgodnie ze wzorami (62) i (63).

$$dblocks = (db_1, db_2, \dots, db_k, \dots, db_L) \tag{62}$$

$$\bigwedge_{k\in\{1\dots L\}} db_k \in \left((dx_k^1, dy_k^1), (dx_k^1, dy_k^2), (dx_k^2, dy_k^1), (dx_k^2, dy_k^2)\right) \tag{63}$$

$$dx_k^1 \in \langle 0 \dots X), dx_k^2 \in \langle 0 \dots X), dy_k^1 \in \langle 0 \dots Y), dy_k^2 \in \langle 0 \dots Y)$$

gdzie: (dx_k^1, dy_k^1) – współrzędne lewego górnego rogu k-tego obszaru dekodowania,
(dx_k^1, dy_k^2) – współrzędne prawego górnego rogu k-tego obszaru dekodowania,
(dx_k^2, dy_k^1) – współrzędne lewego dolnego rogu k-tego obszaru dekodowania,
(dx_k^2, dy_k^2) – współrzędne prawego dolnego rogu k-tego obszaru dekodowania.

Zgodnie z definicją iteracji dekodujących zawartej w pracy *(Pery i Waszkowski, A Model and Quantitative Framework for Evaluating Iterative Steganography, 2024)*, przy założeniu, że dany jest ustalony krok kodowania $step$, można zdefiniować zbiór iteracji dekodujących wzorem (64).

$$s^I = \{s_i : i \in I\} \tag{64}$$

Każdy iteracja dekodująca oznacza wykonanie iteracji *algorytmu dekodującego d-RAI*, podczas której dokonuje się analiza *klatki steganogramu wideo s* oraz dokonywana jest próba zdekodowania *komunikatu*.

Macierz *informacji* dekodowanej w $i'tej$ klatce *steganogramu s* nazywa się $increment(s_i)$ i definiuje się zgodnie ze wzorami (65) i (66).

$$\bigwedge_{i\in s^I} increment(s_i) = \begin{bmatrix} inc_{0,0}^i & \dots & inc_{X-1,0}^i \\ \dots & inc_{x,y}^i & \dots \\ inc_{0,Y-1}^i & \dots & inc_{X-1,Y-1}^i \end{bmatrix} \tag{65}$$

$$\bigwedge_{\substack{i\in s^I \\ x\in<0\dots X) \\ y\in<0\dots Y)}} inc_{x,y}^i = \begin{cases} 1 & gdy\ mask(0)_{x,y}^i = mask1 \\ -1 & gdy\ mask(0)_{x,y}^{i,} = mask0 \\ 0 & w\ przeciwnych\ przypadkach \end{cases} \tag{66}$$

Zakłada się, że liczba bloków obszaru dekodowania $dblocks$ jest równa długości *komunikatu* L, a każdy blok odpowiada jednemu *bitowi* dekodowanego *komunikatu*.

Informację związaną z blokiem dekodującym k zebraną przez wszystkie *iteracje dekodujące* z zakresu $\{1 \dots j\}$ nazywa się $infblock_k^j$ i definiuje się zgodnie ze wzorem (67).

$$\bigwedge_{\substack{j \in s^I \\ k \in \{1 \dots L\}}} infblock_k^j = \sum_{\substack{dx_k^1 \le x \le dx_k^2 \\ dy_k^1 \le y \le dy_k^2}} \sum_{jj=1}^{j} increment\left(s_{iteration_{jj}^s}\right) \tag{67}$$

Progiem istotności *informacji* zebranych przez wszystkie *iteracje dekodujące* z zakresu $\{1 \dots j\}$ nazywa się $inftreshhold^j$ i definiuje się zgodnie ze wzorem (68).

$$\bigwedge_{k \in \{1 \dots L\}} \bigwedge_{j \in s^I} inftreshhold^j = median\left(\{infblock_k^j\}\right) \tag{68}$$

Na bazie pierwotnej definicji iteracyjnej funkcji dekodującej f^{-1} definiuje się iteracyjną funkcję *RAIdecode* dekodującą *komunikat* ze *steganogramu* zgodnie ze wzorami (69), (70), (71) i (72).

$$RAIdecode: \mathcal{S} \to \mathcal{M} \tag{69}$$

$$m^{-1} := RAIdecode(s^i) \tag{70}$$

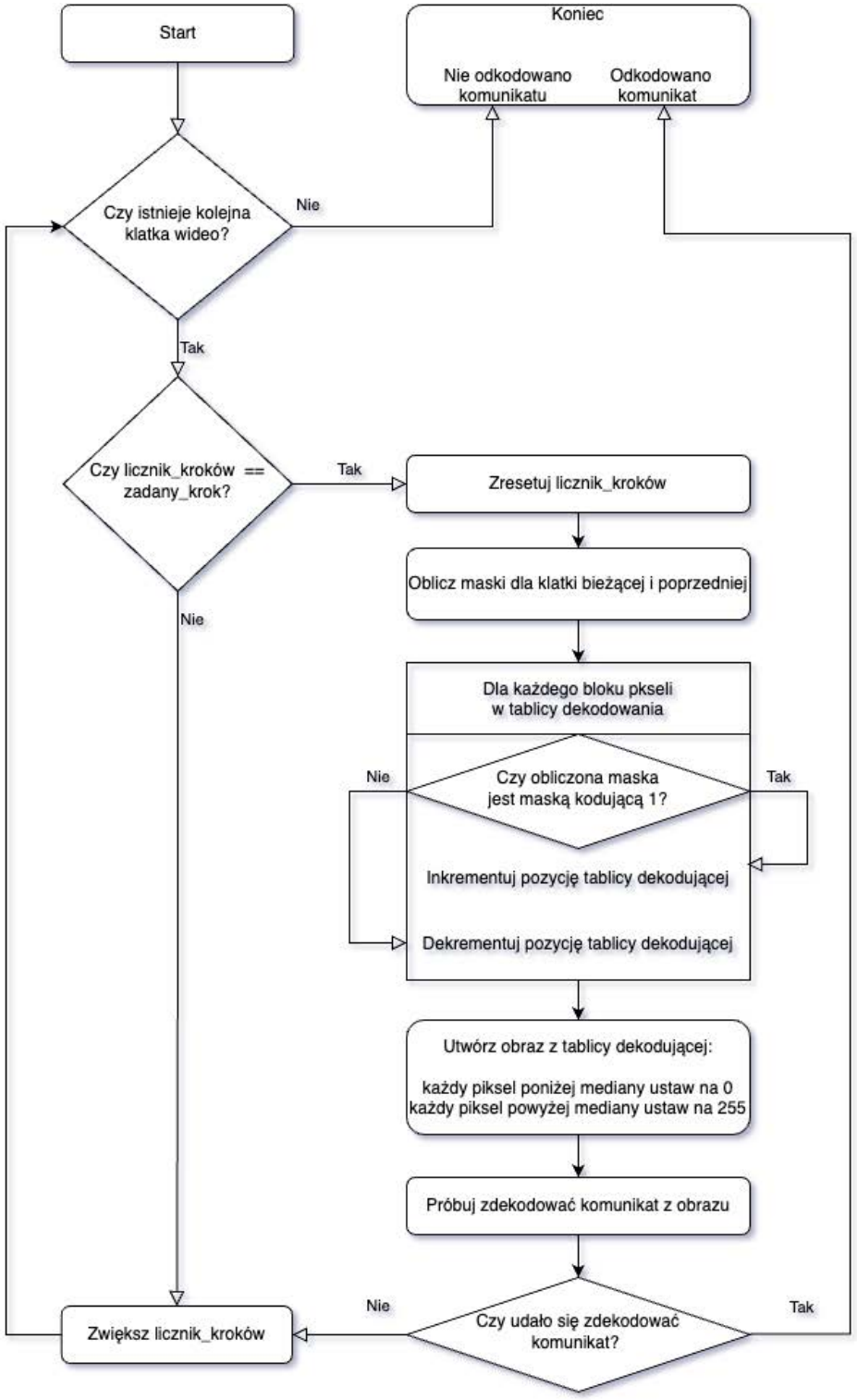
$$m^{-1} = (m_1, m_2, \dots, m_k, \dots, m_L) \tag{71}$$

$$\bigwedge_{\substack{j \in s^I \\ k \in \{1 \dots L\}}} m_k = \begin{cases} 1 & gdy\ infblock_k^j > inftreshhold^j \\ 0 & w\ przeciwnym\ przypadku \end{cases} \tag{72}$$

gdzie: s^i – *steganogram* utworzony przez zakodowanie *komunikatu* m w *nośniku* c^i,
$step$ – *krok kodowania*,
M – liczba bloków dekodujących pikseli,
m^{-1} – *komunikat* zdekodowany przez funkcję f^{-1} ze *steganogramu* s^i.

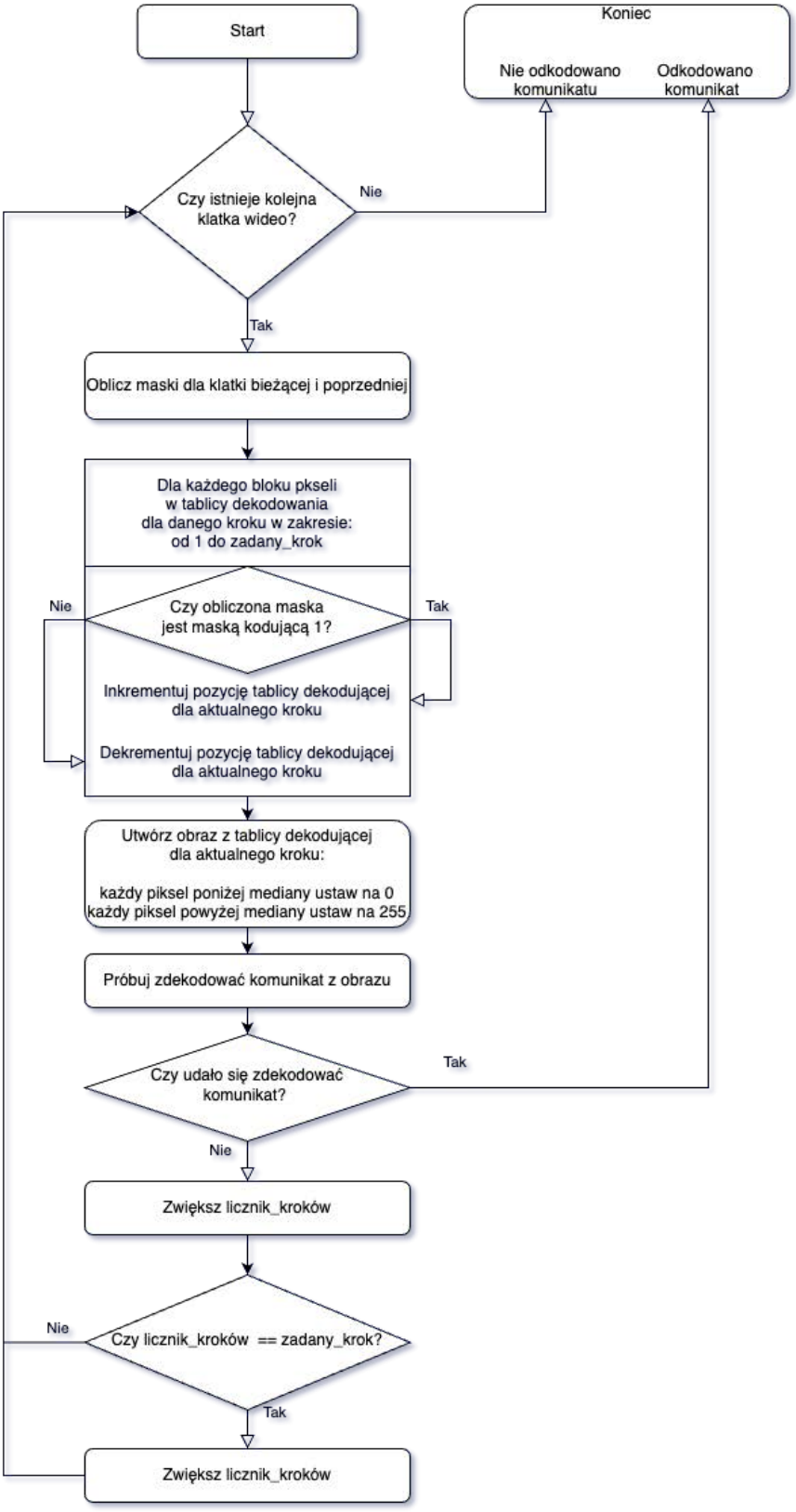
4.1.3.2 Schemat blokowy

Na rysunku 44 przedstawiono schemat blokowy *podstawowego algorytmu dekodującego d-RAI*.



Rysunek 44 - Schemat blokowy *podstawowego algorytmu dekodującego d-RAI w metodzie RAI*
Źródło: *Opracowanie własne*

Na rysunku 45 przedstawiono schemat blokowy *adaptacyjnego algorytmu dekodującego d-RAI*.



Rysunek 45 - Schemat blokowy *adaptacyjnego algorytmu dekodującego d-RAI w metodzie RAI*
Źródło: *Opracowanie własne*

4.1.3.3 Implementacja

Implementacja *podstawowego algorytmu dekodującego d-RAI* przedstawiona jest w kodzie 15. Bazuje ona na formalnym modelu funkcji *RAIdecode* przedstawionym wcześniej.

```
from dbr import BarcodeReader, EnumImagePixelFormat        #version 9.6.40.2

def RAI_decode_message(image):
    """
    Funkcja odczytująca zawartość kodu QR z biało-czarnego obrazu.

    Argumenty:
    - image (np.array): obraz z kodem QR

    Wynik:
    - message (string): zawartość kodu QR
    """

    # Inicjalizacja czytnika kodów
    qrcode = BarcodeReader()

    # Ustawienia trybu detekcji tekstury, z wrażliwością na poziomie 9
    qrcode.set_mode_argument("TextureDetectionModes",0,"Sensitivity", "9")

    # Inicjalizacja pustej wiadomości
    message = ''

    try:
        # Próba odczytania kodu QR
        decoded_texts = qrcode.decode_buffer(
            image=image,
            image_pixel_format=EnumImagePixelFormat.IPF_GRAYSCALED )

        # Jeśli uda się, przypisuje pierwszy wynik do zmiennej message
        if decoded_texts is not None:
            message = decoded_texts[0].barcode_text
    finally:
        # Zwraca zdekodowany komunikat lub pusty string
        return message



def RAI_decode(steganogram, block_size=4, number_of_steps=-1, step=3):
    """
    Funkcja dekodująca komunikat zakodowany w steganogramie wideo metodą RAI.

    Argumenty:
    - steganogram (cv2.VideoCapture): steganogram
    - block_size (int): rozmiar bloku pikseli (domyślnie 4)
    - number_of_steps(int): maksymalna liczba kroków (domyślnie bez ograniczeń)
    - step (int): krok_kodowania (domyślnie 3)

    Wyniki:
    - string: zdekodowany komunikat
    - int: liczba przetworzonych kroków
```

```
    - liczba przetworzonych klatek
    """

    # Inicjalizacja zmiennych
    decoded_text = ''  # Zdekodowany komunikat
    steps_processed = 0  # Liczba przetworzonych kroków
    frame_count = 0  # Liczba przetworzonych klatek

    # Pobranie wysokości i szerokości wideo
    height_steganogram = int(steganogram.get(cv2.CAP_PROP_FRAME_HEIGHT))
    width_steganogram = int(steganogram.get(cv2.CAP_PROP_FRAME_WIDTH))

    # Obliczenie rozmiaru w blokach na podstawie rozdzielczości i block_size
    height = height_steganogram // block_size
    width = width_steganogram // block_size

    # Inicjalizacja tablicy do wyników dekodowania różnic w blokach
    iim = np.zeros((height, width), dtype=np.int64)

    # Generowanie współczynników x i y do iteracji po blokach pikseli
    y_factors = np.arange(0, height_steganogram, block_size)
    x_factors = np.arange(0, width_steganogram, block_size)

    # Tablice przechowujące aktualną i poprzednią ramkę wideo
    frame_current=np.zeros((height_steganogram, width_steganogram,3), np.uint8)
    frame_previous=np.zeros((height_steganogram,width_steganogram,3), np.uint8)

    # Zmienna do zliczania liczby kroków
    step_count = 0

    # Główna pętla przetwarzająca klatki
    while steganogram.isOpened():

        # Odczyt kolejnej klatki wideo
        ok, frame_current = steganogram.read()

        # Przerwanie, jeśli wideo się skończyło lub maksymalna liczba kroków
        if not ok or (number_of_steps>0 and steps_processed==number_of_steps):
            break

        # Inkrementacja licznika kroków
        step_count += 1

        # Jeśli osiągnięto liczbę kroków 'step', przetwórz ramkę
        if step_count == step:
            # Resetowanie licznika kroków i inkrementacja przetworzonych kroków
            step_count = 0
            steps_processed += 1

            # Obliczanie różnicy pomiędzy bieżącą a poprzednią klatką
            frame_decode_diff = (frame_current.astype(np.int16) -
                                 frame_previous.astype(np.int16))

            # Tworzenie schematu dekodowania różnic
            frame_decode = np.ones_like(frame_decode_diff, np.uint8)
            frame_decode[frame_decode_diff < 0] = 2  # 2 jeśli różnica<0
            frame_decode[frame_decode_diff > 0] = 4  # 4 jeśli różnica>0

            # Iteracja po współczynnikach x i y dla bloków pikseli
```

```
            for y_factor, y_start in enumerate(y_factors):
                for x_factor, x_start in enumerate(x_factors):
                    # Wyznaczanie granic bloku pikseli
                    y_end = min(y_start + block_size, height_steganogram)
                    x_end = min(x_start + block_size, width_steganogram)

                    # Pobieranie bloku pikseli w schemacie dekodowania
                    block_schema=frame_decode[y_start:y_end,x_start:x_end]

                    # Zliczanie pikseli, gdzie schemat spełnia kryteria
                    count_on = np.count_nonzero
                          (np.all(block_schema & mask_1 != 0, axis=2))
                    count_off = np.count_nonzero
                          (np.all(block_schema & mask_0 != 0, axis=2))

                    # Aktualizacja tablicy dekodowania różnic
                    iim[y_factor, x_factor] += count_on - count_off

            # Tworzenie obrazu z dekodowanych danych
            decoded_image = np.full((height, width), 255, np.uint8)

            # Wyznaczenie progu istotności informacji
            median_imm = np.median(iim)
            decoded_image[iim > median_imm] = 0

            # Próba dekodowania komunikatu z obrazu
            decoded_text = RAI_decode_message(image=decoded_image)
            if decoded_text != '':
                break

            # Aktualizacja poprzedniej klatki
            frame_previous = frame_current.copy()
            frame_count += 1

    # Zwracanie zdekodowanego komunikatu, liczby przetworzonych kroków i klatek

    return decoded_text, steps_processed, frame_count
```

Kod 15 - *Podstawowy algorytm dekodujący d-RAI*
Źródło: *Opracowanie własne*

Implementacja *adaptacyjnego algorytmu dekodującego d-RAI* przedstawiona jest w kodzie 16.

```
def RAI_decode_adaptive(steganogram, block_size=4, number_of_steps=-1, step=3):
    """
    Funkcja dekodująca komunikat zakodowany w steganogramie wideo metodą RAI,
    z adaptacyjnym podejściem do przesunięcia początkowej klatki kodującej.

    Argumenty:
    - steganogram (cv2.VideoCapture): steganogram
    - block_size (int): rozmiar bloku pikseli (domyślnie 4)
    - number_of_steps(int): maksymalna liczba kroków (domyślnie bez ograniczeń)
    - step (int): krok_kodowania (domyślnie 3)

    Wyniki:
    - string: zdekodowany komunikat
    - int: liczba przetworzonych kroków
    int: liczba przetworzonych klatek

    # Inicjalizacja zmiennych
    decoded_text = ''  # zdekodowany komunikat
    steps_processed = 0  # Liczba przetworzonych kroków
    frame_count = 0  # Liczba przetworzonych klatek

    # Pobranie wysokości i szerokości wideo
    height_steganogram = int(steganogram.get(cv2.CAP_PROP_FRAME_HEIGHT))
    width_steganogram = int(steganogram.get(cv2.CAP_PROP_FRAME_WIDTH))

    # Obliczenie rozmiaru w blokach pikseli na podstawie block_size
    height = height_steganogram // block_size
    width = width_steganogram // block_size

    # Inicjalizacja tablicy do przechowywania wyników dekodowania różnic
    iim = np.zeros((step, height, width), dtype=np.int64)

    # Generowanie współczynników x i y dla bloków pikseli
    y_factors = np.arange(0, height_steganogram, block_size)
    x_factors = np.arange(0, width_steganogram, block_size)

    # Inicjalizacja bieżącej i poprzedniej klatki wideo
    frame_current=np.zeros((height_steganogram,width_steganogram, 3),np.uint8)
    frame_previous=np.zeros((height_steganogram,width_steganogram, 3),np.uint8)

    # Zmienna do zliczania liczby kroków
    step_count = 0

    # Główna pętla przetwarzająca klatki wideo
    while steganogram.isOpened():

        # Odczyt kolejnej klatki
        ok, frame_current = steganogram.read()

        # Przerwanie, jeśli wideo się skończyło lub maksymalna liczba kroków
        if not ok or (number_of_steps>0 and steps_processed==number_of_steps):
            break

        # Przetwarzanie klatek, począwszy od drugiej
```

```
        if frame_count > 0:
            # Obliczanie różnicy pomiędzy bieżącą a poprzednią ramką
            frame_decode_diff = (frame_current.astype(np.int16) -
                                 frame_previous.astype(np.int16))

            # Tworzenie schematu dekodowania różnic
            frame_decode_schema = np.ones_like(frame_decode_diff, np.uint8)
            frame_decode_schema[frame_decode_diff < 0] = 2  # 2 jeśli różnica<0
            frame_decode_schema[frame_decode_diff > 0] = 4  # 4 jeśli różnica>0

            # Iteracja po blokach pikseli na podstawie współczynników
            for y_factor, y_start in enumerate(y_factors):
                for x_factor, x_start in enumerate(x_factors):
                    # Wyznaczanie granic bloków pikseli
                    y_end = min(y_start + block_size, height_steganogram)
                    x_end = min(x_start + block_size, width_steganogram)

                    # Pobieranie bloku pikseli w schemacie dekodowania
                    block_schema=frame_decode_schema[y_start:y_end,
                          x_start:x_end]

                    # Zliczanie pikseli, gdzie schemat spełnia kryteria
                    count_on = np.count_nonzero
                          (np.all(block_schema & mask_1 != 0, axis=2))
                    count_off = np.count_nonzero
                          (np.all(block_schema & mask_0 != 0, axis=2))

                    # Aktualizacja tablicy dekodowania różnic
                    iim[step_count, y_factor, x_factor] += (count_on-count_off)

            # Tworzenie obrazu z dekodowanych danych
            decoded_image = np.full((height, width), 255, np.uint8)

            # Wyznaczenie progu istotności informacji
            median_imm = np.median(iim[step_count])
            decoded_image[iim[step_count] > median_imm] = 0

            # Próba dekodowania komunikatu z obrazu
            decoded_text = decode_message(image=decoded_image)
            if decoded_text != '':
                break

        # Aktualizacja poprzedniej klatki
        frame_previous = frame_current.copy()
        frame_count += 1
        step_count += 1

        # Resetowanie liczników kroków
        if step_count == step:
            step_count = 0
            steps_processed += 1


    # Zwracanie zdekodowanego komunikatu, liczby przetworzonych kroków i klatek
    return decoded_text, steps_processed, frame_count
```

Kod 16 - *Adaptacyjny algorytm dekodujący d-RAI*
Źródło: *Opracowanie własne*

4.2 Definicje najważniejszych pojęć w metodzie RAI

4.2.1 Funkcja IIF, wartości charakterystyczne chIteration oraz chIIF

Ponieważ metoda *RAI* jest techniką *steganografii iteracyjnej*, kluczowym pojęciem ułatwiającym badanie i zrozumienie jej właściwości jest *IIF*[139] - funkcja opisująca iteracyjne przyrosty *informacji* w procesie iteracyjnego dekodowania *komunikatu (Pery i Waszkowski, A Model and Quantitative Framework for Evaluating Iterative Steganography, 2024),* zdefiniowana zgodnie z wzorami (73) i (74).

IIF przyporządkowuje *steganogramowi* oraz *komunikatowi* wartość z przedziału $\langle 0,1\rangle$ oceniając, ile *informacji* udało się *algorytmowi dekodującemu* danej techniki *steganograficznej* zdekodować. Okolice wartości 0,5 oznaczają *szum*, a wartość 1 oznacza, że zdekodowano dokładnie całą *informację*.

$$IIF: \mathcal{S} \times \mathcal{M} \to \langle 0,1\rangle \tag{73}$$

$$\bigwedge_{i\in\{1\dots Z\}} IIF\left(s^i, m\right) = IIF_{(0,0)}\left(s^i, m\right)\cdot\frac{m(0)}{L} + IIF_{(1,1)}\left(s^i, m\right)\cdot\frac{m(1)}{L} = \frac{\sum_{k=1}^{L} 1_{\left\{m_k = f^{-1}\left(s^i\right)_k\right\}}}{L} \tag{74}$$

gdzie: L – długość *komunikatu*,
Z – liczba iteracji dekodujących,
f^{-1} – *algorytm dekodujący*,
m – *komunikat* do zdekodowania ze *steganogramu* s^i,
$m(0)$ – liczba *bitów* zerowych w *komunikacie* m,
$m(1)$ – liczba *bitów* jedynkowych w *komunikacie* m,
m_k –k-ty *bit komunikatu* m,
s^i –*steganogram* utworzony przez zakodowanie *komunikatu* m w *nośniku* c^i.

Funkcja *IIF* posiada następujące właściwości:

1. Zakres wartości: Wartości funkcji *IIF* mieszczą się w przedziale domkniętym od 0 do 1, przy czym szczególnie istotne są wartości z zakresu około 0,5 do 1. Interpretacja tych wartości jest analogiczna (ale odwrotna) do wskaźnika *BER*, gdzie mniejsze

[139] **IIF (Information Incremental Function)** - funkcja opisująca przyrosty *informacji* w kolejnych iteracjach *algorytmu dekodującego RAI.*

wartości wskazują na większy poziom *szumów*, a większe wartości oznaczają większą ilość zakodowanej *informacji*.

2. Monotoniczność: W przypadku prawidłowo działającej techniki *steganograficznej*, obejmującej dobrze zdefiniowane algorytmy kodowania i dekodowania, funkcja *IIF* nie powinna wykazywać tendencji malejącej, z wyjątkiem możliwych pojedynczych, izolowanych przypadków, w których iteracyjny *steganogram* może wykazywać niejednorodność prowadzącą do okresowych zaburzeń monotoniczności. Kolejne iteracje *algorytmu dekodowania RAI* powinny prowadzić do zwiększenia ilości *informacji* zawartej w zdekodowanym *komunikacie* lub w najgorszym przypadku nie powodować jej zmniejszenia.
3. Zachowanie asymptotyczne: Funkcja *IIF* wykazuje charakter asymptotyczny, dążąc do osiągnięcia maksymalnej możliwej wartości *informacji*, jaka może być zdekodowana z danego *steganogramu* przy określonych warunkach *szumu* oraz zadanych parametrach *podstawowego algorytmu kodowania RAI*. W większości przypadków wartość ta nie będzie równa 1, co odpowiadałoby pełnemu *zdekodowaniu informacji* zakodowanej w *oryginalnym steganogramie*. Niemniej jednak powinna ona stanowić na tyle znaczną część *informacji*, aby umożliwić *odbiorcy* poprawne odczytanie treści *komunikatu*.

Wartością charakterystyczną *chIIF*[140] nazywa się taką wartość funkcji *IIF*, dla której ilość *informacji* odczytana ze *steganogramu* s jest wystarczająca do zdekodowania *komunikatu*, zgodnie ze wzorem (75). Wartość charakterystyczną *chIIF* funkcja *IIF* będzie osiągać w *iteracji dekodującej* równej wartości charakterystycznej *chIteration*[141], która jest najmniejszym indeksem *iteracji dekodującej*, dla której nastąpiło skuteczne zdekodowanie *komunikatu* ze *steganogramu* s.

$$chIIF^{s} = IIF(chIteration^{s}) \tag{75}$$

W przypadku, gdy z danego *steganogramu* s nie można odczytać *komunikatu*, przyjmuje się, że wartość $chIIF^{s}$ dla takiego *steganogramu* s wynosi 0.

---

[140] **chIIF** - wartość charakterystyczna funkcji *IIF* - wartość funkcji IIF, dla której ilość *informacji* odczytana ze *steganogramu* jest wystarczająca do zdekodowania *komunikatu*.

[141] **chIteration** - wartość charakterystyczna *iteracji dekodujących* - pierwsza *iteracja dekodująca*, dla której *algorytm dekodujący d-RAI* poprawnie dekoduje *komunikat* ze *steganogramu.*

4.2.2 Wartość charakterystyczna chLevel

Wartością charakterystyczną *chLevel*[142] dla *nośnika c,* zdefiniowaną wzorem (76), nazywa się najmniejszy *poziom kodowania* l, który użyty jako parametr w *podstawowym algorytmie kodowania c-RAI* umożliwia utworzenie takiego *steganogramu* s, z którego udaje się zdekodować *komunikat*.

$$chIIF^{RAIcode(c,m,chLevel^c)} > 0 \ \wedge \bigwedge_{l \in \{1 \ldots chLevel^c - 1\}} chIIF^{RAIcode(c,m,l)} = 0 \tag{76}$$

4.2.3 Czas dekodowania, wartość charakterystyczna chTime

Czasem dekodowania $decodetime_j^s$ dla $iteration_j^s$ mierzonym w sekundach [s] nazywa się, zgodnie ze wzorem (77), czas odtwarzania *steganogramu* s od pierwszej *klatki* do *klatki* o indeksie $iteration_j^s$.

$$decodetime_j^s = time\left(s_{iteration_j^s}\right) - time(s_1) \tag{77}$$

gdzie: $time(f_i^s)$ – chwila pomiaru odtwarzania *i*-tej klatki steganogramu s.

Wartość czasu dekodowania $decodetime_j^s$ *steganogramu* s dla *iteracji dekodującej* równej $chIteration^s$ nazywa się *chTime*[143] i definiuje się zgodnie ze wzorem (78).

$$chTime^s = decodetime^s_{chIteration^s} \tag{78}$$

[142] **chLevel** - wartość charakterystyczna *poziomu kodowania* - minimalny *poziom kodowania*, który musi być użyty w *algorytmie kodującym c-RAI*, aby ze *steganogramu* dało się zdekodować *komunikat*.

[143] **chTime** - wartość charakterystyczna czasu dekodowania - minimalny czas potrzebny do zdekodowania *komunikatu* w metodzie *steganograficznej RAI* definiowany jako czas odtwarzania *steganogramu* od pierwszej *klatki* do *klatki* określonej przez wartość charakterystyczną *chIteration*.

4.2.4 Pojemność, wartość charakterystyczna chCapacity

Pojemnością $capacity_j^s$ dla $iteration_j^s$ mierzoną w *bitach* na sekundę [b/s] nazywa się, zgodnie ze wzorem (79), iloraz długości *komunikatu* L i czasu dekodowania $decodetime_j^s$.

$$capacity_j^s = \frac{L}{decodetime_j^s} \tag{79}$$

$$j \in \{1 \ldots Y\}$$

Wartość *pojemności* $capacity_j^s$ *steganogramu* s dla *iteracji dekodującej* równej $chIteration^s$ nazywa się *chCapacity*[144] i definiuje się zgodnie ze wzorem (80).

$$chCapacity^s = capacity^s_{chIteration^s} \tag{80}$$

4.3 Właściwości metody steganograficznej RAI

W celu zbadania właściwości proponowanej metody *steganograficznej RAI* przeprowadzono serię eksperymentów polegających na kodowaniu *komunikatów* w *steganogramach wideo* za pomocą *podstawowego algorytmu kodującego c-RAI* i dekodowaniu ich przy pomocy *algorytmu dekodującego d-RAI*.

Eksperymenty były przeprowadzone przy różnych *poziomach kodowania* oraz w różnych warunkach zakłócających *kanał* pomiędzy *nadawcą* a *odbiorcą komunikatu steganograficznego*, w tym w warunkach rzeczywistych *zniekształceń analogowych*. Analizie poddano m.in. otrzymane wyniki miar *odporności* i *pojemności*. Jednym z najważniejszych eksperymentów było badanie *niewykrywalności* metody RAI zrealizowane na próbie osób.

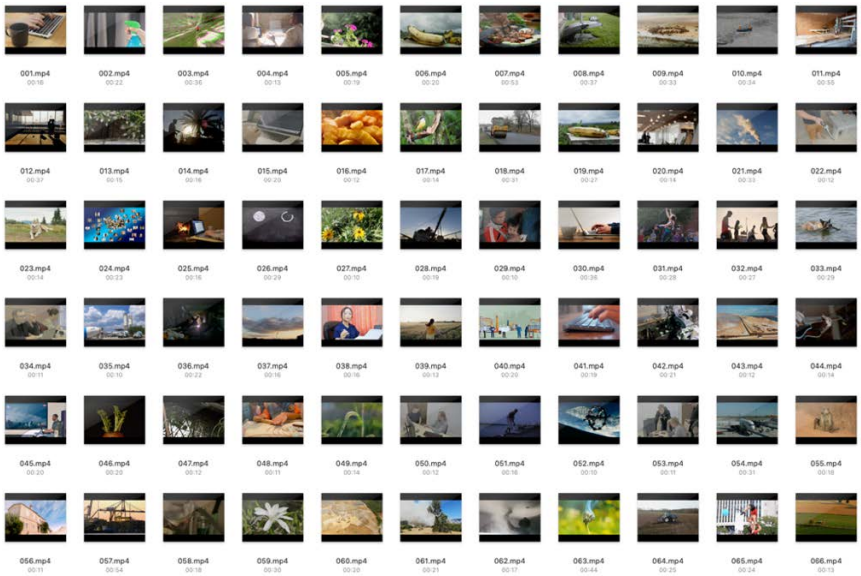
[144] **chCapacity** - wartość charakterystyczna *pojemności* metody *steganograficznej RAI* definiowana jako iloraz długości kodowanego *komunikatu* do *chTime* - minimalnego czasu potrzebnego na zdekodowanie *komunikatu* ze *steganogramu*.

4.3.1 Przyjęte założenia do eksperymentów badawczych

Do wszystkich badań, analiz i eksperymentów, wykonanych w ramach niniejszej pracy, wykorzystano pliki *wideo* pobrane z ogólnodostępnej bazy w serwisie *pixabay.com* zgodnie z zapisami licencji o możliwym wykorzystaniu w celach niekomercyjnych *(Pixabay.com, 2024)*. Są to te same pliki wideo, które zostały wykorzystane do badań w opracowaniu *(Pery i Waszkowski, Analyzing natural pixel color variations in consecutive video frames to enhance spatial domain video steganography, 2024)*.

Z serwisu wylosowano łącznie 666 plików *wideo* o łącznym czasie trwania ok. 15 tysięcy sekund. Najkrótsze pliki miały 10 sekund, najdłuższe 72 sekundy, a mediana czasu trwania dla wszystkich plików *wideo* wyniosła ok. 17 sekund. Pliki zostały wylosowane z różnych kategorii tematycznych oraz z różną charakterystyką dynamiczną i kolorystyczną. Każde kolejne *wideo* zostało zakodowane *kodekiem H.264* w *rozdzielczości* 1920x1080 oraz zostało zapisane pod losowym numerem w formacie ‘000.mp4’. Tak utworzoną bazę 666 plików *wideo* stanowiących *nośniki* testowe dla prowadzonych w ramach niniejszej pracy badań nazwano *bazą nośników*[145].

Na rysunku 46 przedstawiono fragment *bazy nośników* zapisany w dedykowanym katalogu na serwerze.



Rysunek 46 - Przykład: fragment *bazy nośników*
Źródło: *(Pixabay.com, 2024)*

[145] **Baza nośników** - baza plików *wideo* pobranych z serwisu *(Pixabay.com, 2024)*.

W pierwszej części badań nad właściwościami metody *RAI* wykorzystano techniki automatyczne. Stworzono różne bazy *steganogramów* o różnych *poziomach kodowania* przy użyciu *podstawowego algorytmu kodującego c-RAI*, które następnie poddano automatycznym analizom. Przeprowadzono również automatyczne symulacje różnych typów przekształceń i zniekształceń *steganogramów wideo*, a także zbadano ich właściwości w kontekście wpływu tych modyfikacji na wartości miar *odporności* oraz *pojemności*.

Druga część badań nad właściwościami metody *RAI* została przeprowadzona na grupie osób i miała na celu ocenę, czy metoda *RAI* cechuje się wystarczającą *niewykrywalnością*.

Wszystkie wyniki eksperymentów i analiz są szczegółowo omówione w kolejnych rozdziałach.

4.3.1.1 Wybór komunikatu na potrzeby eksperymentów

Komunikatem, który był kodowany w *steganogramach,* był tekst *"Grom"* zapisany w postaci kodu *QR*[146] w wersji 1 z poziomem korekcji błędów H, co oznacza możliwość zakodowania do 72 *bitów komunikatu* w postaci matrycy 21x21 modułów z korektą błędów na poziomie do 30% *(DensoWave, 2024)*. Zastosowanie w przeprowadzanych eksperymentach *komunikatu* w postaci kodu *QR* z jednej strony w naturalny sposób wykorzystywało posiadane przez kody *QR* cechy *redundancji* i *kodowania korekcyjnego*, co istotnie zwiększało *odporność* tworzonych *steganogramów*, a z drugiej strony umożliwiało wykorzystanie automatycznych narzędzi do zdekodowania *komunikatu*, które w sposób obiektywny i znormalizowany odpowiadały na pytanie, czy ilość *informacji* przekazywana w *steganogramie* była wystarczająca do odczytania *komunikatu*. Dodatkową zaletą wykorzystania mechanizmu kodowania *informacji* w *komunikacie* w formacie kodów *QR* jest perspektywa podjęcia dalszych prac badawczo-rozwojowych nad praktycznymi narzędziami wykorzystującymi metodę *steganografii RAI* w rzeczywistych zastosowaniach z wykorzystaniem istniejących komercyjnych narzędzi i aplikacji.

---

[146] **QR code (Quick Response code)** - dwuwymiarowy kod kreskowy opracowany przez japońską firmę Denso Wave. Dzięki swojej matrycowej strukturze kody *QR* mogą przechowywać znacznie więcej danych niż jednowymiarowe kody kreskowe. Cechują się szybkością skanowania, możliwością przechowywania różnorodnych informacji i odpornością na częściowe uszkodzenia dzięki mechanizmom korekcji błędów *(DensoWave, 2024)*.

Na rysunku 47 przedstawiono zakodowany w postaci kodu *QR komunikat* tekstowy "Grom″.

***Rysunek 47** - Przykład: komunikat "Grom"* zapisany w postaci kodu QR
Źródło: *Opracowanie własne*

Jako narzędzia do odczytywania kodów *QR* użyto komercyjnego czytnika kodów *QR* o nazwie Bar Reader *(Dynamsoft, 2024)*. Możliwe jest zastąpienie tego narzędzia przez ogólnodostępne darmowe oprogramowanie, np. funkcje z biblioteki *OpenCV*[147].

4.3.1.2 Wybór masek kodowania na potrzeby eksperymentów

Maską nazywa się trójkę, w której na każdej pozycji może się znajdować jeden z trzech symboli oznaczających pewną grupę różnic wartości kolorów pikseli następujących po sobie *klatek steganogramu wideo*. W opracowaniu "Analyzing natural pixel color variations in consecutive video frames to enhance spatial domain video steganography″ *(Pery i Waszkowski, Analyzing natural pixel color variations in consecutive video frames to enhance spatial domain video steganography, 2024) przyjęto, że takimi symbolami są: "+","-","* ", które oznaczają następujące zbiory wartości: $(\{1\ldots255\},\{0\},\{-255\ldots-1\})$.

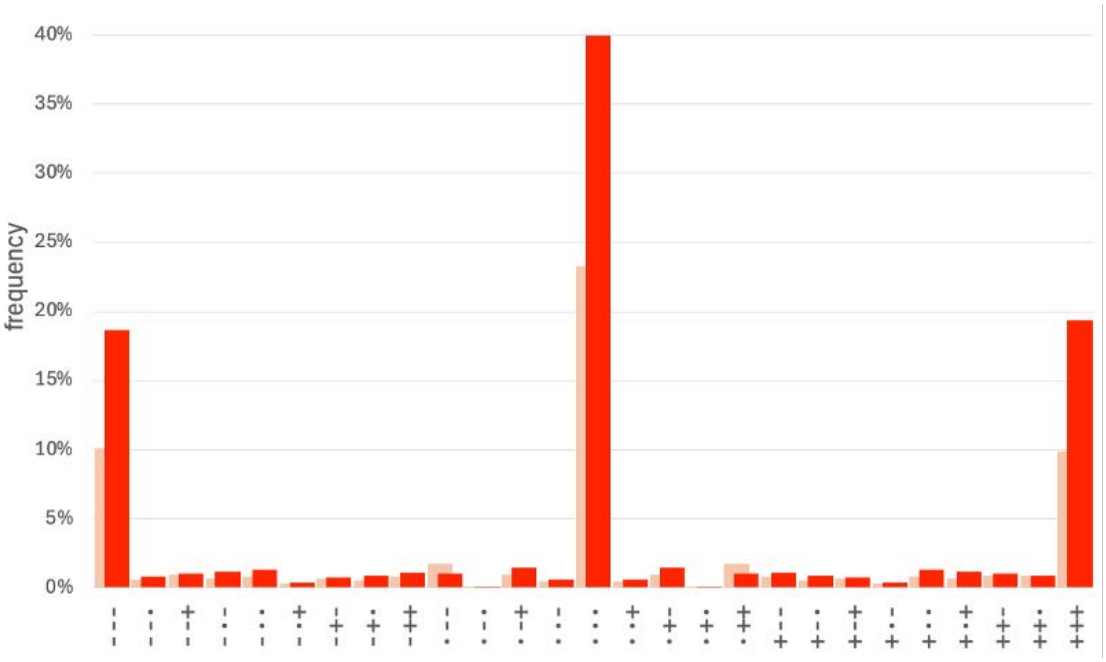
[147] **OpenCV (Open Source Computer Vision Library)** - biblioteka programistyczna zaprojektowana do realizacji zadań związanych z komputerowym przetwarzaniem obrazów i widzeniem maszynowym. Oferuje szeroki zestaw algorytmów do analizy obrazów i *wideo*, takich jak detekcja i rozpoznawanie obiektów, segmentacja, śledzenie ruchu, rekonstrukcja 3D, oraz przetwarzanie obrazu w czasie rzeczywistym *(Opencv.org, 2024)*.

Z wyników badań opublikowanych w pracy *(Pery i Waszkowski, Analyzing natural pixel color variations in consecutive video frames to enhance spatial domain video steganography, 2024)* wynika, dla plików *wideo* nie zakłóconych żadnymi procesami kodowania obrazu (w tym kompresji lub *steganografii*), istnieją pewne charakterystyczne, naturalne częstotliwości występowania poszczególnych masek. W tabeli 14 przedstawione są uzyskane wartości częstotliwości występowania poszczególnych *masek* w *nośnikach.*

Tabela 14 - Wyniki: wartości częstotliwości występowania *masek* w *bazie nośników*
Źródło: *(Pery i Waszkowski, Analyzing natural pixel color variations in consecutive video frames to enhance spatial domain video steganography, 2024)*

groups	frequency	σ
---	18,69%	10,08%
--	0,84%	0,62%
--+	1,03%	0,98%
- -	1,21%	0,71%
-	1,34%	0,83%
- +	0,39%	0,37%
-+-	0,79%	0,73%
-+	0,90%	0,58%
-++	1,15%	0,81%
--	1,05%	1,79%
-	0,12%	0,15%
-+	1,47%	0,97%
-	0,65%	0,49%
	39,95%	23,30%
+	0,64%	0,49%
+-	1,48%	0,96%
+	0,13%	0,15%
++	1,07%	1,77%
+--	1,15%	0,85%
+-	0,90%	0,57%
+-+	0,78%	0,70%
+ -	0,38%	0,32%
+	1,37%	0,85%
+ +	1,21%	0,69%
++-	1,04%	0,94%
++	0,91%	0,92%
+++	19,37%	9,88%

Na rysunku 48 przedstawiono histogram częstotliwości występowania *masek* w *bazie nośników.*



Rysunek 48 - Wyniki: histogram częstotliwości występowania *masek* w *bazie nośników*
Źródło: *(Pery i Waszkowski, Analyzing natural pixel color variations in consecutive video frames to enhance spatial domain video steganography, 2024)*

W celu wyznaczenia *masek kodujących* dla metody *RAI* rozważono następujące kwestie:

1. Ze względu na dużą częstotliwość występowania oraz praktyczną nierozróżnialność od szumu, *maski "---", "   " i "+++" nie mogą być wykorzystane do kodowania bitów komunikatu.*
2. *Każda pozycja maski o wartości " " oznacza brak zmiany wartości koloru piksela między klatkami, co praktycznie uniemożliwia skuteczne kodowanie informacji na tym kolorze. Dlatego maski zawierające " " są nieodpowiednie do kodowania informacji w metodzie RAI.*
3. *Pozostałe dostępne maski to: "--+", "-+-", "-++", "+--", "+-+", "++-". Można je pogrupować w przeciwstawne pary: "--+" z "++-", "-+-" z "+-+", "-++" z "+--".*
4. *Ponieważ naturalne częstotliwości występowania wybranych par masek są małe i zbliżone do siebie, kodowanie bitów komunikatu przy użyciu dowolnej z tych par daje porównywalne wyniki, a każda z wybranych par jest praktycznie równoważna pod względem możliwości wykorzystania w metodzie RAI.*

W pracy przyjęto, że spośród dostępnych par masek, podstawowy algorytm kodujący RAI oraz algorytm dekodujący RAI będą używać maski "+-+" do kodowania bitu 1 komunikatu, a maski przeciwstawnej "-+-" do kodowania bitu 0 komunikatu.

W kodach algorytmu kodującego c-RAI i algorytmów dekodujących d-RAI, ze względu na wygodę implementacji, przyjęto, że symbol " " kodowany będzie ustawionym bitem zerowym (co odpowiada wartości 1), symbol "-" kodowany będzie ustawionym bitem pierwszym (co odpowiada wartości 2), a symbol "+" kodowany będzie ustawionym *bitem* trzecim (co odpowiada wartości 4). Wybrany sposób reprezentacji kodowania kolejnych stanów za pomocą wartości 1, 2 i 4 (i tym samym odpowiadającym im kolejnych *bitów masek kodujących*), umożliwia wykonywanie efektywnych operacji na macierzach reprezentujących kolejne *klatki steganogramu wideo*. W celu wybrania pikseli, które spełniają warunek zdefiniowanej *maski,* wystarczy wówczas wykonać na macierzy pikseli kolorów bitową operację logiczną. Dla innych ewentualnych implementacji *algorytmu kodującego c-RAI* i *algorytmów dekodujących d-RAI* efektywniejsze mogą okazać się inne reprezentacje *masek kodowania* (z innymi wartościami odpowiadającymi poszczególnym stanom różnicy kolorystycznych).

4.3.1.3 Wybór poziomów kodowania na potrzeby eksperymentów

Eksperymenty kodowania *podstawowym algorytmem kodującym c-RAI* były przeprowadzane na pięciu *poziomach kodowania* od 1 do 5. Przy kodowaniu na *poziomie kodowania* 5 zmiany w *steganogramach* były już w większości przypadków widoczne gołym okiem, więc przeprowadzanie eksperymentów z wyższym *poziomem kodowania* nie miało praktycznego uzasadnienia. Na rysunku 49 pokazano przykładowe *klatki* oryginalnego *nośnika* oraz *steganogramów* zakodowanych *podstawowym algorytmem kodującym RAI* z *poziomami kodowania* od 1 do 5. W przypadku *poziomu kodowania* 1 i 2 zmiany w *steganogramie* są niezauważalne ludzkim okiem, natomiast na wyższych *poziomach kodowania* zmiany są coraz bardziej widoczne i po uważnym przyjrzeniu się można spostrzec przebarwienia kolorów w centralnych miejscach obrazu.

przykładowa *klatka wideo* oryginalnego *nośnika*
(bez zakodowanego *komunikatu*)

przykładowa *klatka wideo steganogramu*
(z *komunikatem* w postaci *QR kodu*, poziom kodowania=1)

przykładowa *klatka wideo* oryginalnego *nośnika*
(z *komunikatem* w postaci *QR kodu*, poziom kodowania=2)

przykładowa *klatka wideo steganogramu*
(z *komunikatem* w postaci *QR kodu*, poziom kodowania=3)

przykładowa *klatka wideo* oryginalnego *nośnika*
(z *komunikatem* w postaci *QR kodu*, poziom kodowania=4)

przykładowa *klatka wideo steganogramu*
(z *komunikatem* w postaci *QR kodu*, poziom kodowania=5)

Rysunek 49 - Przykład: porównanie tej samej *klatki nośnika* oraz *steganogramów* z coraz większymi *poziomami kodowania*
Źródło: *Opracowanie własne*

4.3.1.4 Wybór kroku kodowania na potrzeby eksperymentów

Z punktu widzenia maksymalizacji miary *pojemności* metody *RAI* do kodowania *steganograficznego* powinien zostać wybrany najmniejszy możliwy *krok kodowania*. Im mniejszy jest *krok kodowania*, tym więcej jest w jednostce czasu *iteracji dekodujących*, co oznacza szybsze narastanie *informacji* dekodowanej ze *steganogramu*.

Z matematycznego punktu widzenia najmniejszym *krokiem kodowania* jest wartość 2. Jednakże z praktycznego punktu widzenia przyjęcie *kroku kodowania* o wartości 2 uniemożliwiałoby porównanie *masek* pomiędzy *klatkami* zmienionymi przez *steganografię* i *masek* nie zmienionych przez proces kodowania *komunikatu*.

Uwzględniając powyższe przyjęto, że *krokiem kodowania* dla *podstawowego algorytmu kodującego c-RAI* oraz *algorytmu dekodującego d-RAI* będzie wartość 3.

4.3.1.5 Wybór maksymalnej liczby iteracji na potrzeby eksperymentów

Bazując na doświadczeniu uzyskanym w wyniku przeprowadzenia wielu eksperymentów kodowania i dekodowania *komunikatów* ze *steganogramów* z wykorzystaniem metody *RAI* oraz przyjmując praktyczne założenia dotyczące oczekiwanej efektywności metody *RAI* założono, że wartością graniczną czasu odtwarzania *steganogramu wideo*, podczas którego powinno nastąpić zdekodowanie *komunikatu* jest czas 10 sekund. Przy założeniu najmniejszej praktycznie wykorzystywanej liczby *fps = 25* odpowiada to około 250 *klatkom*, a przy założonym *kroku kodowania* = 3 oznacza to maksymalnie 84 *iteracje dekodujące*.

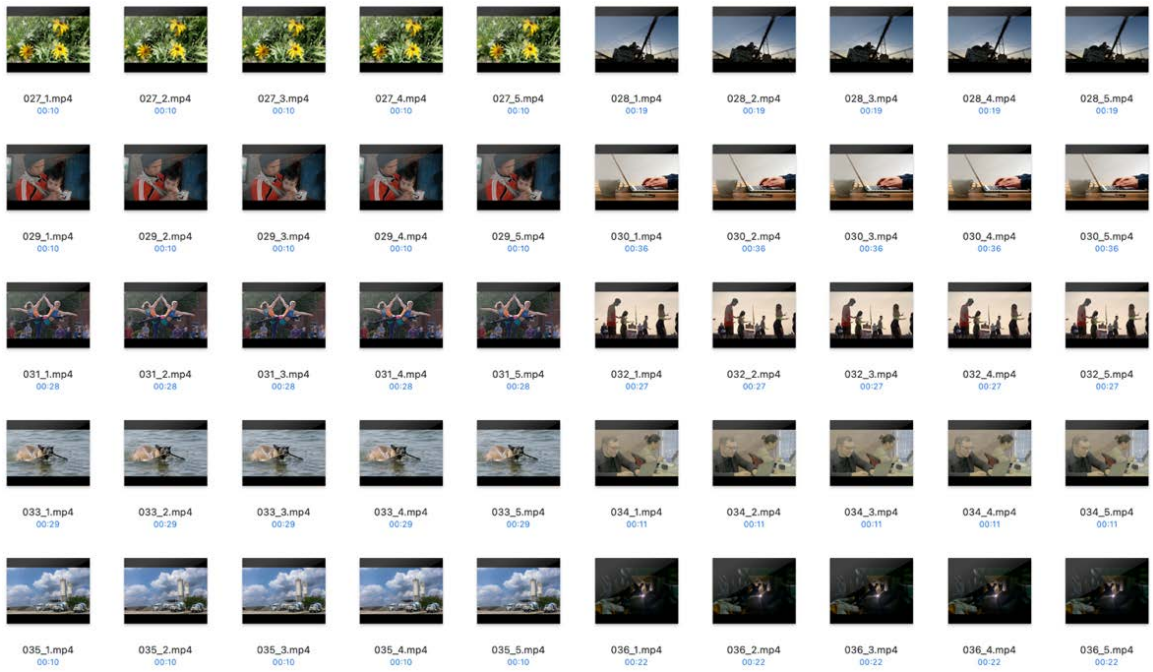
Wyniki eksperymentów umieszczone w dalszej części pracy pokazują, że przyjęte założenie jest wystarczające. Liczby *iteracji dekodujących* potrzebnych do zdekodowania *komunikatu* ze *steganogramów* były albo niższe niż przyjęta maksymalna granica 84, albo z danego *steganogramu* nie dało się w ogóle zdekodować *komunikatu*.

4.3.2 Badanie podstawowych właściwości steganogramów RAI

W każdym *nośniku* z *bazy nośników* przy zastosowaniu pięciu *poziomów kodowania* od 1 do 5 zostały zakodowane *podstawowym algorytmem kodującym RAI komunikaty* „Grom” tworząc z każdego *nośnika* pięć nowych plików *wideo* nazwanych *oryginalnymi steganogramami*. Każdy *oryginalny steganogram* zakodowano *kodekiem H.264* w identycznej *rozdzielczości* jak *nośnik* 1920x1080 nadając mu nazwę w formacie ‘000_0.mp4’ zawierającej informację o unikalnym numerze *nośnika* oraz zastosowanym *poziomem kodowania*. Tak utworzoną bazę 3330 plików *wideo* nazwano *bazą oryginalnych steganogramów*[148]. Na rysunku 50 przedstawiono fragment *bazy oryginalnych steganogramów* zapisany w dedykowanym katalogu na serwerze.

---

[148] **Baza oryginalnych steganogramów** - baza plików *wideo* będących *oryginalnymi steganogramami* utworzonymi w wyniku zakodowania w *plikach z bazy nośników komunikatu "Grom"* w postaci kodu *QR podstawowym algorytmem kodującym RAI* z kolejnymi *poziomami kodowania*.



Rysunek 50 - Przykład: fragment *bazy oryginalnych steganogramów*
Źródło: *Opracowanie własne*

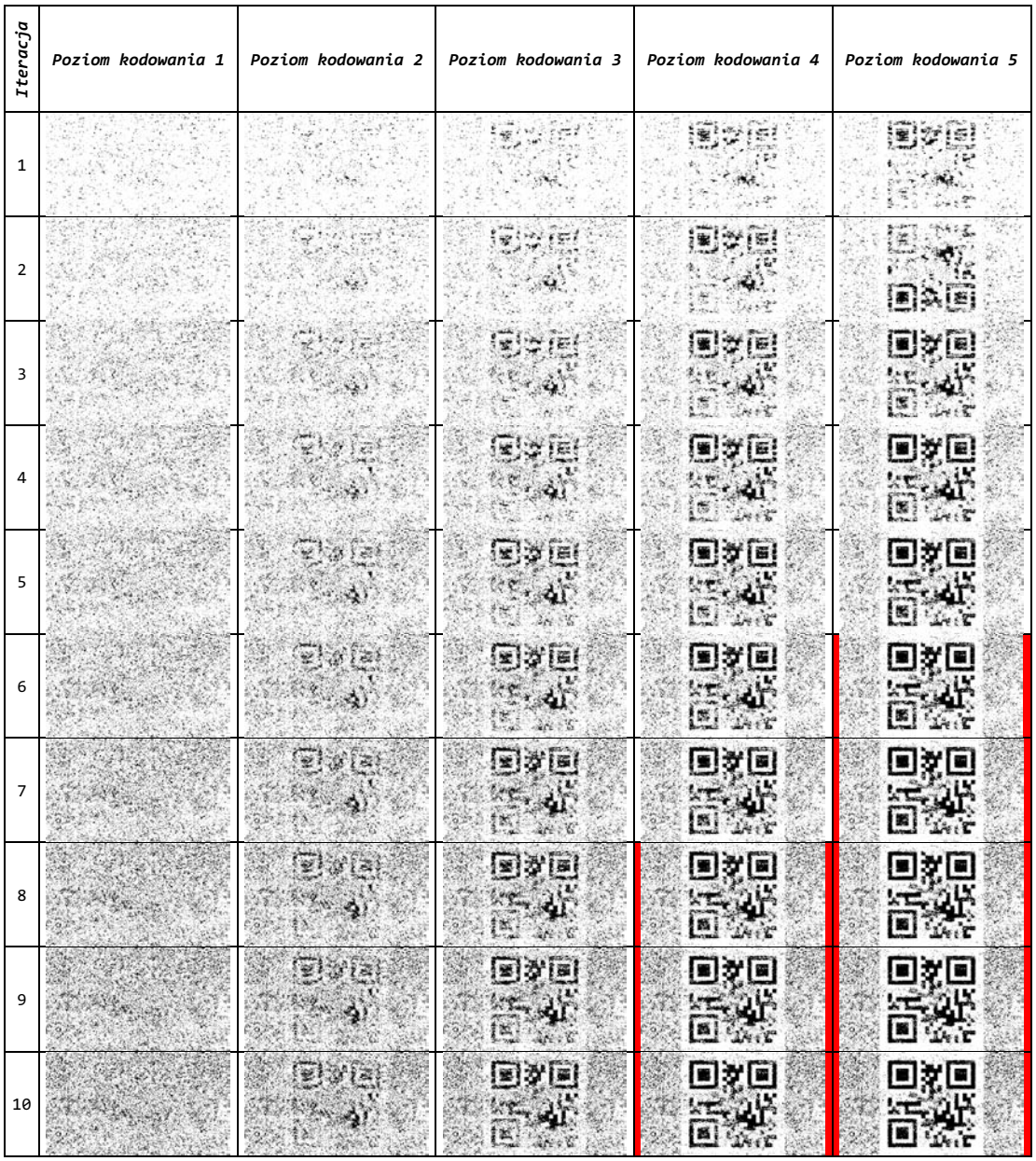
4.3.2.1 Wartości charakterystyczne chIIF

Dla każdego z 3330 *oryginalnych steganogramów* z *bazy oryginalnych steganogramów* wyznaczono przebieg funkcji *IIF* w całym zakresie *iteracji dekodujących*. Dla każdej *iteracji dekodującej* podjęto również próbę automatycznego zdekodowania *komunikatu*.

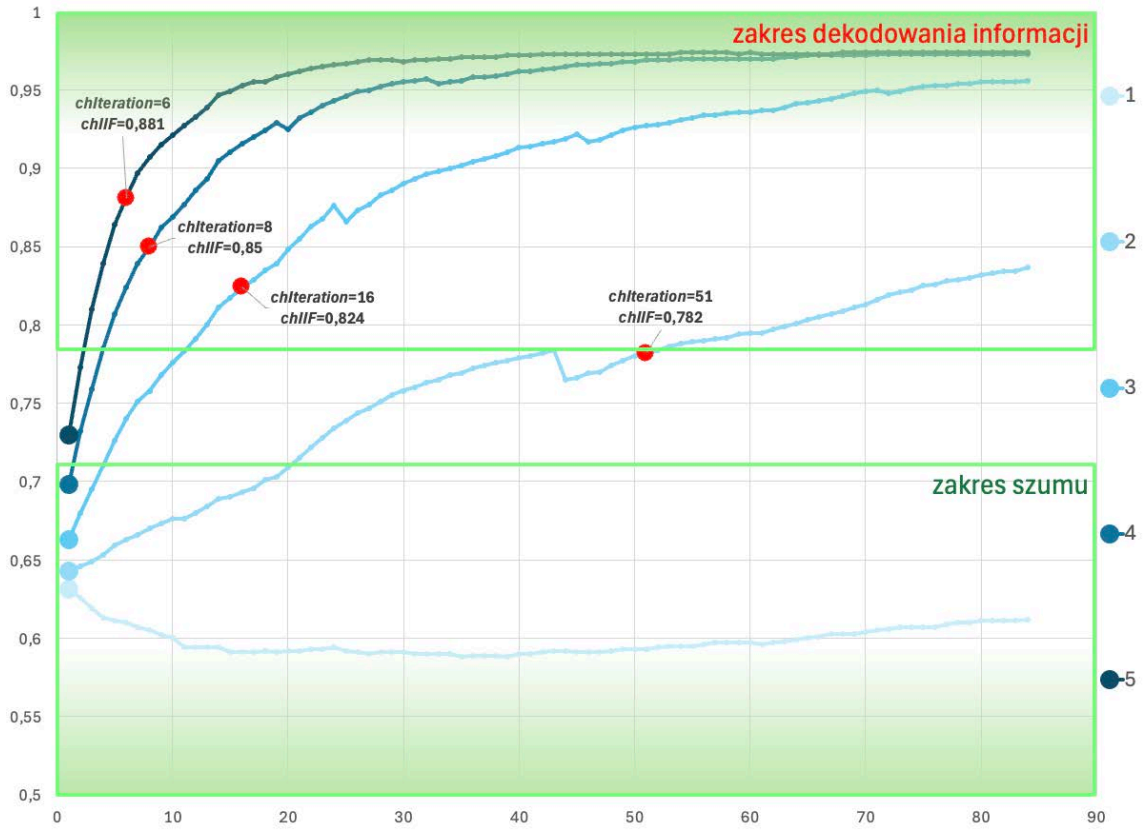
Wartość graniczną *chIIF* funkcji *IIF* umożliwiającą zdekodowanie *komunikatu* udało się osiągnąć, przy różnych *poziomach kodowania* i różnych liczbach *iteracji dekodujących*, łącznie dla 1975 *oryginalnych steganogramów*. Dla pozostałych steganogramów wartości funkcji *IIF* w badanym zakresie *iteracji dekodujących* nie wychodziły poza obszar *szumu* oscylując wokół wartości 0,5.

W tabeli 15 przedstawiony jest przykład procesu przyrostowego zwiększania *informacji* w dekodowanym *komunikacie* w postaci kodu *QR* dla *steganogramu* zakodowanego różnymi *poziomami kodowania*. Dla każdej *iteracji dekodującej* wraz z wzrostem ilości *informacji*, kod *QR* reprezentujący *komunikat* jest coraz bardziej wyraźny, co jest związane ze wzrostem wartości funkcji *IFF,* osiągając przy pewnej liczbie *iteracji dekodujących* wartość graniczną pozwalającą zdekodować *komunikat*.

Tabela 15 - Przykład: przyrostowe zwiększanie *informacji* zakodowanej w *komunikacie* w *oryginalnych steganogramach* w kolejnych iteracjach *algorytmu dekodującego* d-*RAI* dla różnych *poziomów kodowania* z zaznaczonymi na czerwono iteracjami >= *chIteration,* dla których występuje automatyczne zdekodowanie *komunikatu*
Źródło: *Opracowanie własne*

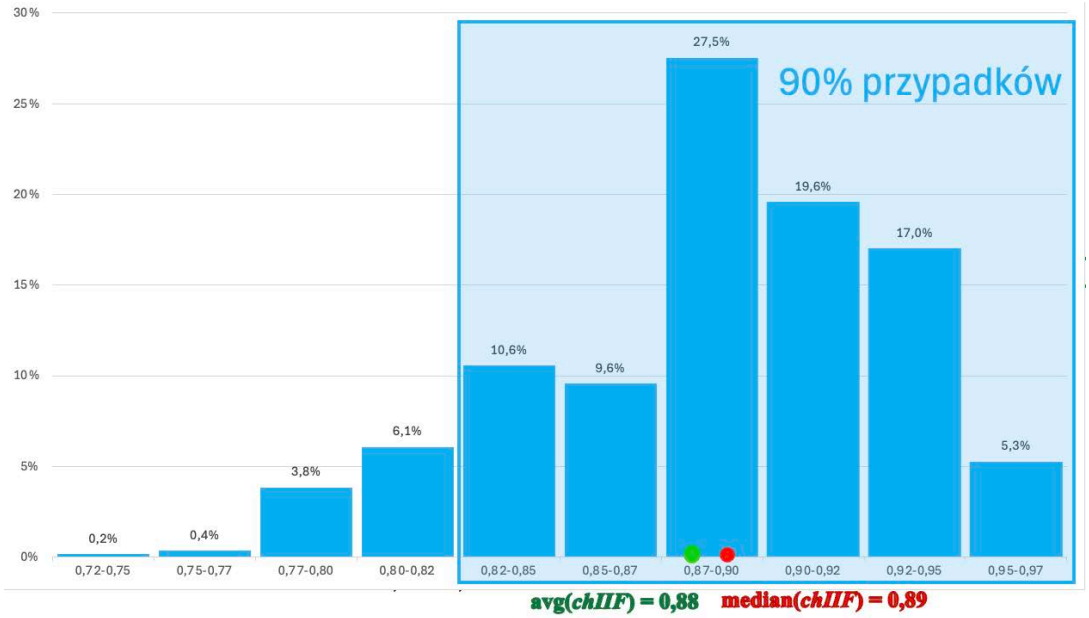


Na rysunku 51 dla tych samych przykładowych *oryginalnych steganogramów* przedstawiony jest wykres przebiegu funkcji *IFF*. Widać na nim wyraźnie, że w przypadku *poziomu kodowania* = 1 wartości funkcji *IIF* nie wychodzą poza obszar *szumu*, natomiast dla kolejnych *poziomów kodowania*, im wyższy jest *poziom kodowania*, tym szybciej wartości *IIF* osiągają wartości charakterystyczne *chIIF* oraz zbiegają asymptotycznie do wartości maksymalnej.



Rysunek 51 - Przykład: przyrost wartości *IIF* dla kolejnych iteracji *algorytmu dekodującego RAI* wraz z wyznaczonymi wartościami *chIIF* oraz *chIteration* dla *steganogramów* utworzonych z różnymi *poziomami kodowania* od 1 do 5
Źródło: *Opracowanie własne*

Na rysunku 52 przedstawiony jest histogram *chIIF* dla *bazy oryginalnych steganogramów* z wyznaczoną medianą, średnią i pierwszym decylem *chIIF*.



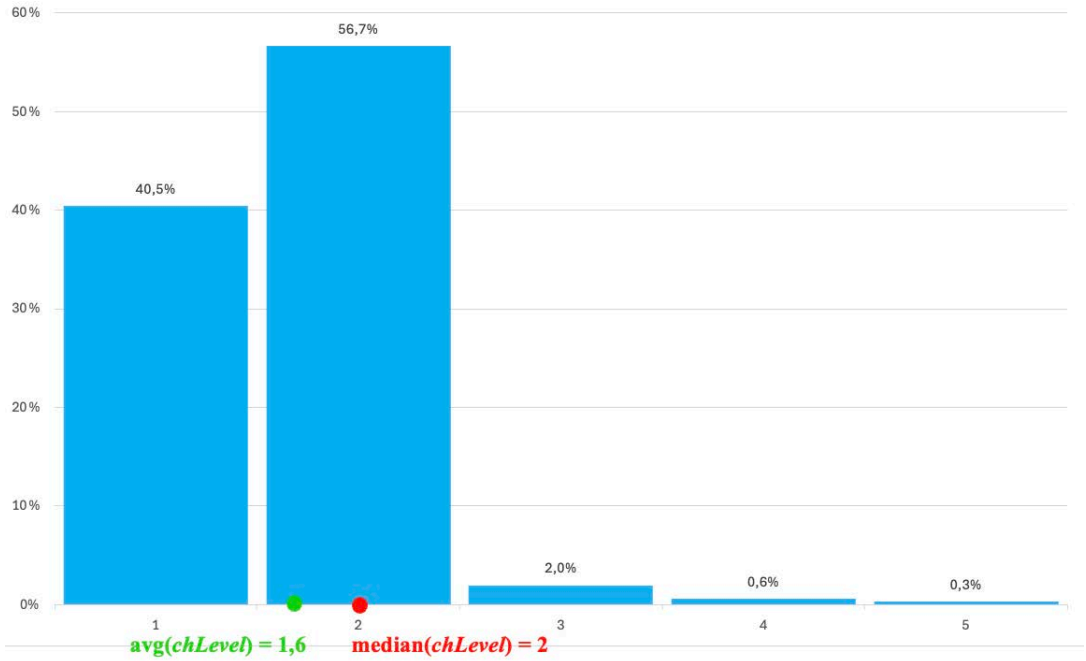
Rysunek 52 - Wyniki: histogram *chIIF* dla *bazy oryginalnych steganogramów* z wyznaczoną medianą, średnią i pierwszym decylem *chIIF*
Źródło: *Opracowanie własne*

4.3.2.2 Wartości charakterystyczne chLevel

Dla każdego *wideo* w *bazie oryginalnych steganogramów* wyznaczono minimalny *poziom kodowania chLevel*, dla którego udawało się algorytmowi dekodującemu *RAI* w zadanej maksymalnej liczbie *iteracji dekodujących* osiągnąć wartość charakterystyczną *chIIF* funkcji *IIF*, czyli zdekodować *komunikat*. Ze względu na przyjęte założenia dotyczące liczby *poziomów kodowania*, obliczona wartość charakterystyczna *chLevel* mieściła się w przedziale od 1 do 5.

Dla 6 spośród 666 *nośników* nie udało się na żadnym z pięciu *poziomów kodowania oryginalnych steganogramów* zdekodować *komunikatu* w badanym zakresie *iteracji dekodujących*. Były to głównie schematyczne, proste prezentacje komputerowe z małą zmiennością poszczególnych *klatek*, a więc z małymi możliwościami ukrycia dodatkowej *informacji* wewnątrz obrazu. W takich przypadkach danemu *nośnikowi* przypisywano jako wynik eksperymentu wartość pustą. W tabeli 22 (umieszczonej w załączniku) przedstawiono uzyskane wyniki *chLevel* przypisane poszczególnym *nośnikom* odpowiadającym poszczególnym *steganogramom* będących przedmiotem niniejszego eksperymentu.

Na podstawie uzyskanych wyników wyznaczono *histogram* wartości charakterystycznych *chLevel*, który zaprezentowano na rysunku 53. Na rysunku 53 pokazano również obliczoną medianę *chLevel* równą 2 i wartość średnią *chLevel* równą 1,6.

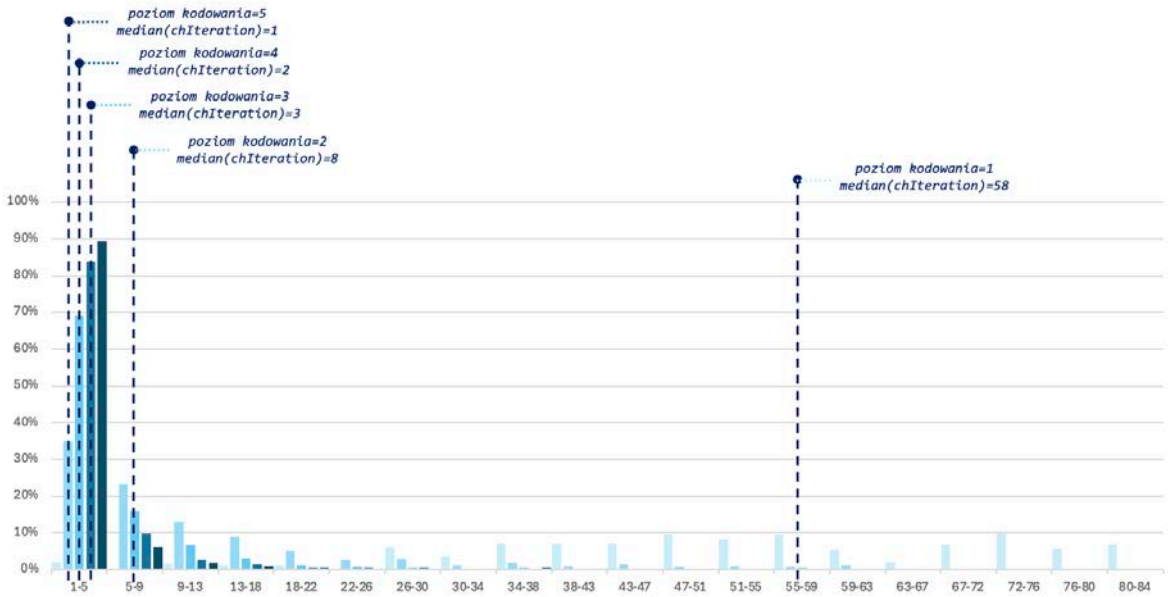


Rysunek 53 - Wyniki: histogram *chLevel* dla *bazy nośników* wraz z wyznaczoną medianą oraz średnią *chLevel*
Źródło: *Opracowanie własne*

4.3.2.3 Wartości charakterystyczne chIteration

Dla każdego *wideo* z *bazy oryginalnych steganogramów* na każdym z pięciu *poziomów kodowania*, dla których udało się zdekodować *komunikat* w zadanej maksymalnej liczbie *iteracji dekodujących*, znaleziono pierwszą *iterację dekodującą,* w której udało się zdekodować *komunikat*. Zgodnie z przyjętą definicją znaleziona *iteracja dekodująca* jest wartością charakterystyczną *chIteration*, która może być traktowana jako jedna z metryk *odporności* metody RAI, gdyż im mniejsza jest wartość *chIteration*, tym mniejsze są straty *informacji* pomiędzy *klatkami* i tym szybciej udaje się zdekodować *komunikat* ze *steganogramu*.

W tabelach 23, 24, 25, 26 oraz 27 (umieszczonych w załączniku) przedstawione są wyniki obliczeń *chIteration* dla wszystkich *steganogramów* z *bazy oryginalnych steganogramów* dla wszystkich pięciu *poziomów kodowania*. Na podstawie uzyskanych wyników wyznaczono *histogramy* i obliczono mediany dla *bazy oryginalnych steganogramów* dla poszczególnych poziomów kodowania, które zaprezentowano na rysunku 54.



Rysunek 54 - Wyniki: *histogramy chIteration* dla *oryginalnych steganogramów*
Źródło: *Opracowanie własne*

4.3.3 Badanie odporności metody RAI

Najważniejszą miarą z punktu widzenia *scenariusza zniekształceń analogowych* oraz proponowanej metody *steganograficznej RAI* jest *odporność*. W tej części pracy sprawdzono jak *odporne* są *steganogramy* utworzone metodą *RAI* w przypadku zastosowania różnego typu przekształceń i zniekształceń.

Kluczowymi metrykami oceny *odporności* metody *RAI* są wartości charakterystyczne *poziomów kodowania chLevel* oraz liczby *iteracji dekodujących chIteration* dla poszczególnych *poziomów kodowania* oraz ich statystyki opisowe takie jak mediana i średnia. Niższe wartości *chLevel* świadczą o większej *odporności steganogramu*, ponieważ algorytm kodujący *RAI* może wykorzystać niższe *poziomy kodowania*, co zwiększa podobieństwo *steganogramu* do oryginalnego *nośnika*. Podobnie mniejsze wartości *chIteration* oznaczają większą *odporność*, ponieważ algorytmy dekodujące potrzebują mniejszej liczby iteracji do poprawnego dekodowania ukrytego *komunikatu*, co oznacza mniejszą utratę *informacji* pomiędzy kolejnymi *klatkami steganogramu*.

W kolejnych podrozdziałach opisane zostaną eksperymenty, które wykonano na *bazie oryginalnych steganogramów* poddając je różnego typu zniekształceniom i przekształceniom. Za każdym razem tworzono nową bazę przekształconych *steganogramów*, które poddawano badaniom i dla których wyznaczono wskaźniki *chIteration* przy wszystkich *poziomach kodowania*. Otrzymane wyniki poddano następnie analizie.

4.3.3.1 Odporność metody RAI na kompresję

W tym eksperymencie zbadano *odporność* metody *RAI* na zastosowanie kompresji. Każdy *steganogram* z *bazy oryginalnych steganogramów* poddano kompresji *kodekiem H.264* przy użyciu narzędzia *FFmpeg*[149]. W procesie kompresji zastosowano domyślną wartość 23 parametru CRF (Constant Rate Factor) przyjmującego wartości od 0 do 51 i określającego kompromis pomiędzy jakością a rozmiarem *wideo*. Kod 17 przedstawia przykład użycia *FFmpeg* do tworzenia skompresowanych *steganogramów*.

[149] **FFmpeg** - narzędzie do przetwarzania multimediów obsługujące kodowanie, dekodowanie, transkodowanie i strumieniowanie plików *audio* i *wideo (FFmpeg, 2024)*.

```
# Wywołanie ffmpeg kompresującego wideo

ffmpeg -i input.mp4 -vcodec libx264 -crf 23 output.mp4
# -i input.mp4: specyfikuje plik wejściowy (input.mp4)
# -vcodec libx264: ustawia format H.264
# -crf 23: ustawia współczynnik jakości CRF na wartość standardową
# output.mp4: plik wyjściowy (output.mp4)
```

Kod 17 - Przykład wywołania narzędzia *FFmpeg* kompresującego *wideo* z użyciem *kodeka H.264* oraz wartością CRF = 23
Źródło: *Opracowanie własne*

Z tak przekształconych 3330 plików *wideo* utworzono *bazę skompresowanych steganogramów*[150]. Średni stopień kompresji dla wszystkich plików poddanych kompresji wyniósł 2,58.

Rysunek 55 przedstawia przykład tej samej *klatki oryginalnego steganogramu* i *steganogramu* skompresowanego. Ludzkim okiem nie można wykryć zmian pomiędzy obrazami, ale prezentowane w dalszej części pracy wyniki badań pokazują, że kompresowanie wpływa na zwiększenie trudności w dekodowaniu *komunikatu* poprzez degradację *informacji* zakodowanych w *steganogramie*.

przykładowa *klatka steganogramu* nie poddanego *kompresji stratnej*

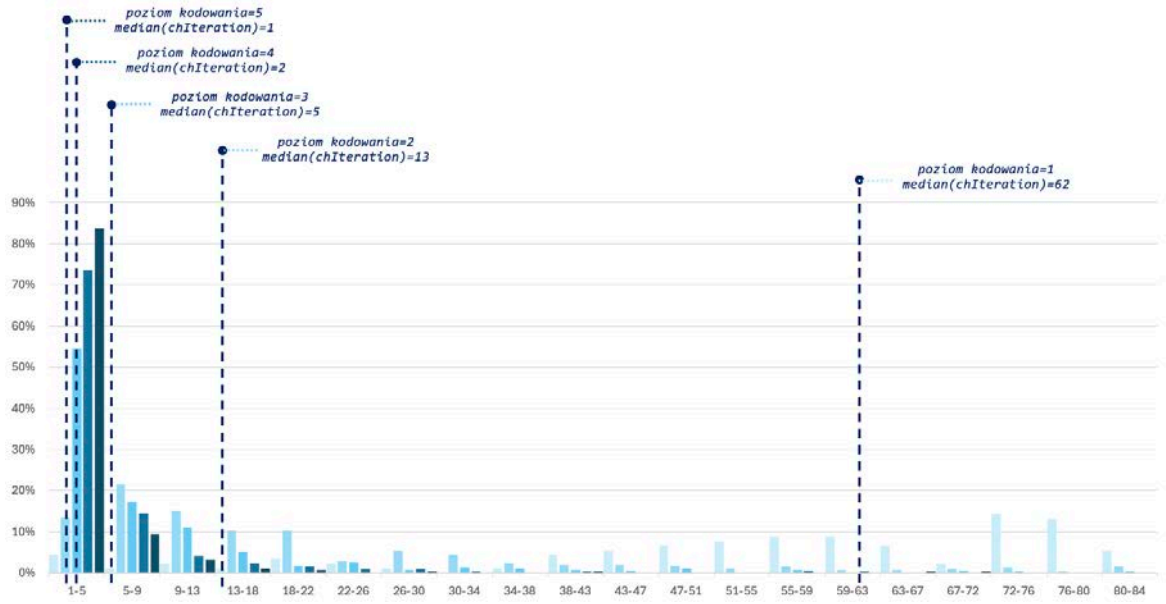
przykładowa *klatka steganogramu* poddanego *kompresji stratnej*

Rysunek 55 - Przykład: porównanie *klatek steganogramów z bazy oryginalnych steganogramów* (nie poddanych *kompresji*) *oraz z bazy skompresowanych steganogramów* (poddanych *kompresji*)
Źródło: *Opracowanie własne*

W tabelach 28, 29, 30, 31 oraz 32 (umieszczonych w załączniku) przedstawione są wyniki obliczeń *chIteration* dla wszystkich *steganogramów* z *bazy skompresowanych steganogramów* dla wszystkich pięciu *poziomów kodowania*.

[150] **Baza skompresowanych steganogramów** - baza plików *wideo* będących skompresowanymi *oryginalnymi steganogramami* z *bazy oryginalnych steganogramów.*

Na podstawie uzyskanych wyników obliczono *histogramy* i wyznaczono mediany *chIteration* dla *bazy skompresowanych steganogramów* dla kolejnych *poziomów kodowania* i zaprezentowano je na rysunku 56.



Rysunek 56 - Wyniki: *histogramy chIteration* dla *bazy skompresowanych steganogramów*
Źródło: *Opracowanie własne*

4.3.3.2 Odporność metody RAI na transformacje geometryczne

W tym eksperymencie zbadano *odporność* metody *RAI* na zastosowanie kilku transformacji geometrycznych. Każdą *klatkę* każdego *steganogramu* z *bazy oryginalnych steganogramów* poddano po kolei następującym przekształceniom:

1. rzutowi perspektywicznemu przekształcającemu *klatkę steganogramu* na czworokąt zgodnie z macierzą transformacji: (0,24; 0,12; 0,18; 0,22),
2. obrotowi wokół środka *klatki* o 10 stopni w przeciwnym kierunku do ruchu wskazówek zegara,
3. osadzeniu *klatki steganogramu* na tle *klatki nośnika* o numerze o jeden większym niż numer *steganogramu*, przy czym dla *steganogramu* o numerze 666 tłem był *nośnik* o numerze 1.

Kod 18 zawiera przykładowy algorytm, który realizuje powyższe transformacje geometryczne.

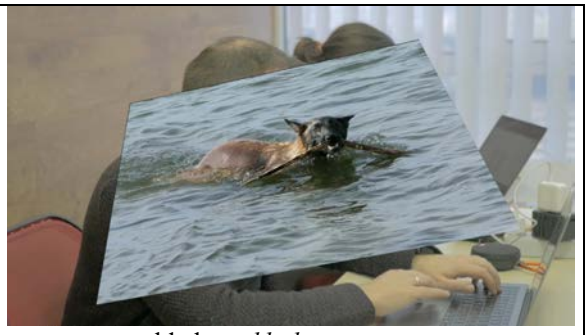
```
def transform_image(image, background, rotation_angle=10,
                    transformation_matrix=[0.24, 0.12, 0.18, 0.22]):
    """
    Przekształca obraz przez rzut perspektywiczny, obrót i osadzenie w tle.
    Argumenty:
    - image (np.array): obraz do przekształcenia
    - background (np.array): obraz tła
    - rotation_angle (float): kąt rotacji obrazu (domyślnie 10 stopni)
    - transformation_matrix (list): macierz określająca perspektywę
    Wynik:
    - result (np.array): obraz wynikowy z przekształconym obrazem na tle.
    """

    # Wysokość i szerokość obrazu
    height, width = image.shape[:2]

    # Definiowanie punktów źródłowych (rogi obrazu)
    src_points = np.float32([[0, 0], [width - 1, 0],
                             [width - 1, height - 1], [0, height - 1]])

    # Definiowanie punktów docelowych w oparciu o macierz transformacji
    dst_points = np.float32([
        [width * transformation_matrix[0], height * transformation_matrix[3]],
        [width * (1 - transformation_matrix[0]), height *
             transformation_matrix[2]],
        [width * (1 - transformation_matrix[1]), height * (1 -
             transformation_matrix[2])],
        [width * transformation_matrix[1], height * (1 -
             transformation_matrix[2])] ])

    # Obliczenie macierzy transformacji perspektywy
    M = cv2.getPerspectiveTransform(src_points, dst_points)
    # Przekształcenie perspektywy obrazu
    warped = cv2.warpPerspective(image, M, (width, height))
    # Macierz do rotacji obrazu wokół środka
    M_rot = cv2.getRotationMatrix2D((width / 2, height / 2), rotation_angle, 1)
    # Zastosowanie rotacji na przekształconym obrazie
    rotated = cv2.warpAffine(warped, M_rot, (width, height))
    # Utworzenie maski wypełnionej czarnym tłem
    mask = np.zeros((height, width), dtype=np.uint8)
    # Transformacja punktów w celu rotacji maski
    points = np.int32(cv2.transform(np.array([dst_points]), M_rot))[0]
    # Wypełnienie maski białym kolorem w obszarze wyznaczonym przez punkty
    cv2.fillPoly(mask, [points], 255)
    # Skopiowanie tła do wyniku
    result = background.copy()
    # Wstawienie obszaru z obróconym obrazem tam, gdzie maska jest biała
    result[mask > 0] = rotated[mask > 0]

    return result
```

Kod 18 - Przykład implementacji kilku transformacji geometrycznych *klatki steganogramu*: rzutu perspektywicznego, obrotu i osadzenia na tle innego obrazu
Źródło: *Opracowanie własne*

Z tak przekształconych 3330 plików *wideo* utworzono *bazę transformowanych steganogramów*[151].

Rysunek 57 przedstawia przykład tej samej *klatki oryginalnego steganogramu* i odpowiadającego mu *steganogramu* z *bazy transformowanych steganogramów*. Ze względu na charakter przekształceń różnice pomiędzy obrazami są całkowicie widoczne ludzkim okiem. Zastosowane transformacje redukują znacznie ilość *informacji* kodowanych w *steganogramie*, co widać w wynikach pomiarów *steganogramów* z *bazy transformowanych steganogramów* przedstawionych w dalszej części pracy.

przykładowa *klatka steganogramu* nie poddanego transformacji



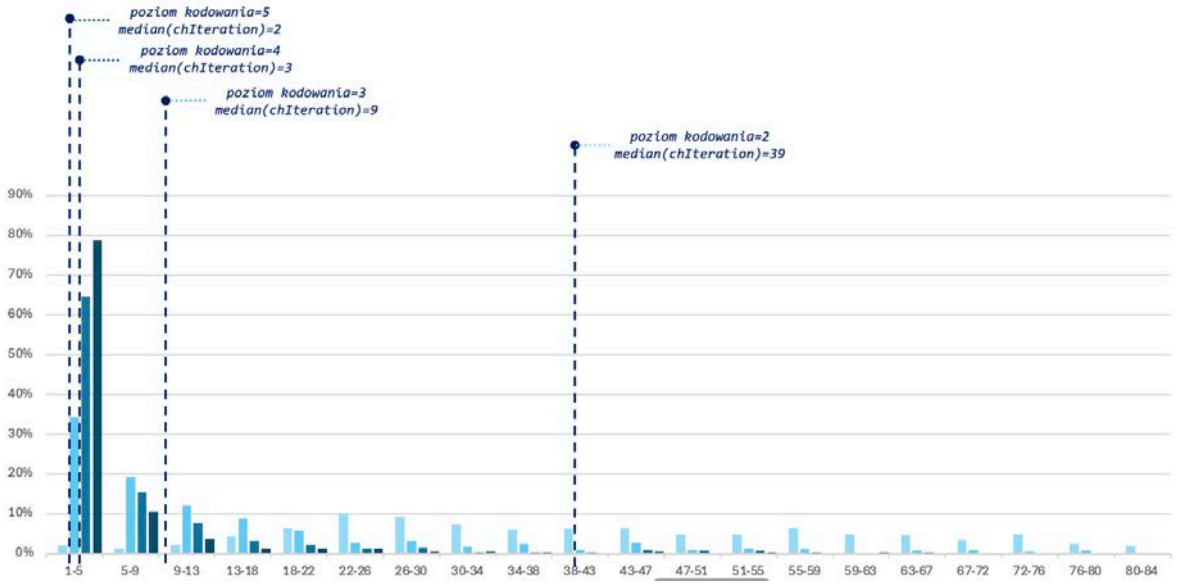
przykładowa *klatka steganogramu* poddanego transformacji

Rysunek 57 - Przykład: porównanie *klatek steganogramów z bazy oryginalnych steganogramów* (nie poddanych transformacji) oraz *bazy transformowanych steganogramów* (poddanych transformacji)
Źródło: *Opracowanie własne*

W tabelach 33, 34, 35, 36 oraz 37 (umieszczonych w załączniku) przedstawione są wyniki obliczeń *chIteration* dla wszystkich *steganogramów* z *bazy transformowanych steganogramów* dla wszystkich *poziomów kodowania*.

Na podstawie uzyskanych wyników obliczono *histogramy* i wyznaczono mediany *chIteration* dla *bazy transformowanych steganogramów* dla *poziomów kodowania* od 2 do 5 przedstawione na rysunku 58. Dla *poziomu kodowania* 1 nie udało się zdekodować *komunikatu* z żadnego ze *steganogramów* z *bazy transformowanych steganogramów*.

[151] **Baza transformowanych steganogramów** - baza plików *wideo* utworzona poprzez zastosowanie do *oryginalnych steganogramów* z *bazy oryginalnych steganogramów* kilku przekształceń geometrycznych, tj. rzutu perspektywicznego, obrotu i osadzenia na tle innego nośnika.



Rysunek 58 - Wyniki: *histogramy chIteration* dla *bazy transformowanych steganogramów*
Źródło: *Opracowanie własne*

4.3.3.3 Odporność metody RAI na szum

W tym eksperymencie zbadano *odporność* metody *RAI* na szum. Każdą *klatkę* każdego *steganogramu* z *bazy oryginalnych steganogramów* poddano zaszumieniu *szumem Gaussa* o *odchyleniu standardowym* równym 20, tworząc w ten sposób *bazę zaszumionych steganogramów*[152] o liczebności 3330 plików *wideo*. Przykład algorytmu realizującego zaszumienie *klatki* steganogramu *szumem Gaussa* zamieszczono w kodzie 19.

```
def add_noise_to_image(image, noise_level=20):
    """
    Dodaje szum Gaussa do obrazu.

    Argumenty:
    - image (np.array): obraz, do którego dodawany jest szum
    - noise_level (int): poziom szumu (domyślnie 20)

    Wynik:
    - result (np.array): obraz z dodanym szumem
    """

    # Tworzenie szumu Gaussa o podanej intensywności
    noise = np.random.normal(0, noise_level, image.shape).astype(np.float32)
```

[152] **Baza zaszumionych steganogramów** - baza plików *wideo* utworzona poprzez zastosowanie do *oryginalnych steganogramów* z *bazy oryginalnych steganogramów* zaszumienia *szumem Gaussa*.

```
    # Dodanie szumu do obrazu
    result = cv2.add(image.astype(np.float32), noise, dtype=cv2.CV_8UC3)

    # Przycięcie wartości pikseli do zakresu 0-255
    result = np.clip(result, 0, 255).astype(np.uint8)

    return result
```

Kod 19 - Przykład funkcji realizującej zaszumienie *klatki steganogramu szumem Gaussa* o *odchyleniu standardowym*=20
Źródło: *Opracowanie własne*

Na rysunku 59 przedstawiony jest przykład tej samej *klatki oryginalnego steganogramu* i odpowiadającego mu *steganogramu* z *bazy zaszumionych steganogramów*. W przekształconym *steganogramie* widoczne są wyraźne zniekształcenia obrazu, które wpływają na zniszczenie części *informacji* utrudniając zdekodowanie *komunikatu*.

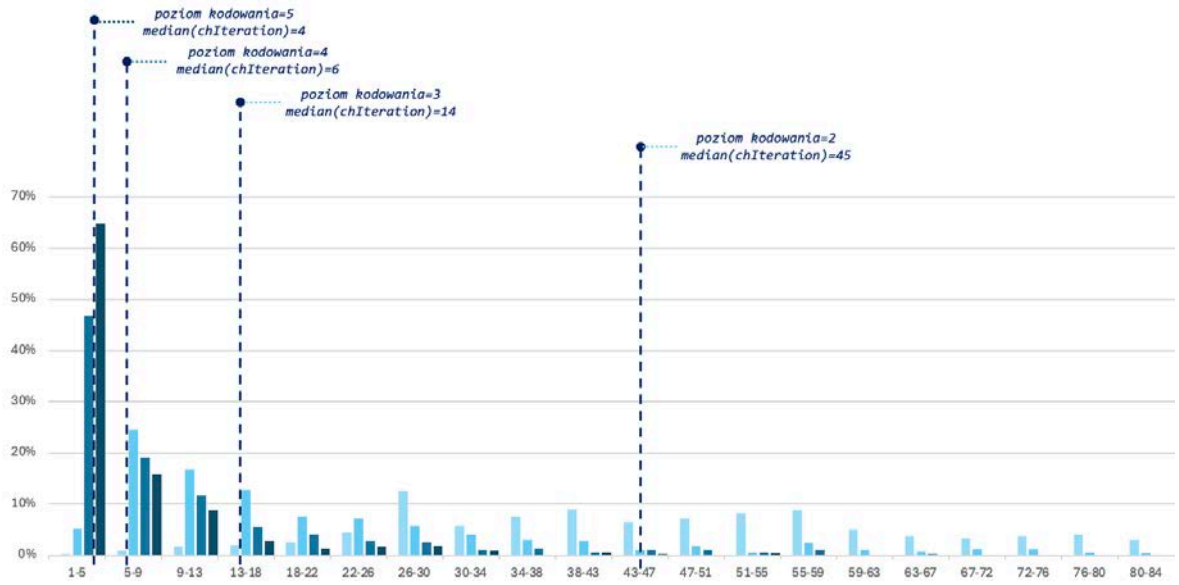
przykładowa *klatka steganogramu* niezaszumionego

przykładowa *klatka steganogramu* zaszumionego

Rysunek 59 - Przykład: porównanie *klatek steganogramów z bazy oryginalnych steganogramów* (niezaszumionych) oraz *bazy zaszumionych steganogramów* (zaszumionych)
Źródło: *Opracowanie własne*

W tabelach 38, 39, 40, 41 oraz 42 (umieszczonych w załączniku) przedstawione są wyniki obliczeń *chIteration* dla wszystkich *steganogramów* z *bazy zaszumionych steganogramów* dla wszystkich *poziomów kodowania*.

Na podstawie uzyskanych wyników obliczono *histogramy* i wyznaczono mediany *chIteration* dla *bazy zaszumionych steganogramów* dla *poziomów kodowania* od 2 do 5 przedstawione na rysunku 60. Dla *poziomu kodowania* 1 nie udało się zdekodować *komunikatu* z żadnego ze *steganogramów* z *bazy zaszumionych steganogramów*.



Rysunek 60 - Wyniki: *histogramy chIteration* dla *zaszumionych steganogramów*
Źródło: *Opracowanie własne*

4.3.3.4 Odporność metody RAI na wiele przekształceń jednocześnie

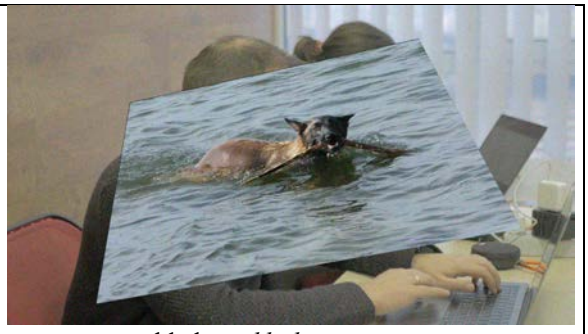
W tym eksperymencie zbadano *odporność* metody *RAI* na zastosowanie naraz kilku przekształceń i zniekształceń o różnym charakterze. Odpowiada to opisanemu uprzednio schematowi *scenariusza zniekształceń analogowych*. W tym celu każdą *klatkę* każdego *steganogramu* z *bazy oryginalnych steganogramów* poddano następującym przekształceniom:

1. kompresji narzędziem *FFmpeg* z zastosowaniem *kodeka H.264* oraz domyślnego poziomu kodowania 23,
2. rzutowi perspektywicznemu przekształcającemu *klatkę steganogramu* na czworokąt zgodnie z macierzą transformacji: (0,24; 0,12; 0,18; 0.22),
3. obrotowi wokół środka *klatki* o 10 stopni w przeciwnym kierunku do ruchu wskazówek zegara,
4. osadzeniu *klatki steganogramu* na tle *klatki nośnika* o numerze o jeden większym niż numer *steganogramu*, przy czym dla *steganogramu* o numerze 666 tłem był *nośnik* o numerze 1,
5. zaszumieniu utworzonej *klatki steganogramu szumem Gaussa* o *odchyleniu standardowym* równym 20.

Z tak przekształconych *steganogramów* utworzono *bazę zniekształconych steganogramów*[153] o liczebności 3330 plików *wideo*.

Na rysunku 61 przedstawiony jest przykład tej samej *klatki oryginalnego steganogramu* i odpowiadającego mu *steganogramu* z *bazy zniekształconych steganogramów*. Widoczne gołym okiem przekształcenia i zniekształcenia istotnie utrudniają zdekodowanie *komunikatu*, co będzie widoczne w prezentowanych dalej wynikach analizy *steganogramów* z *bazy zniekształconych steganogramów.*

przykładowa *klatka steganogramu* nie poddanego wielokrotnym zniekształceniom

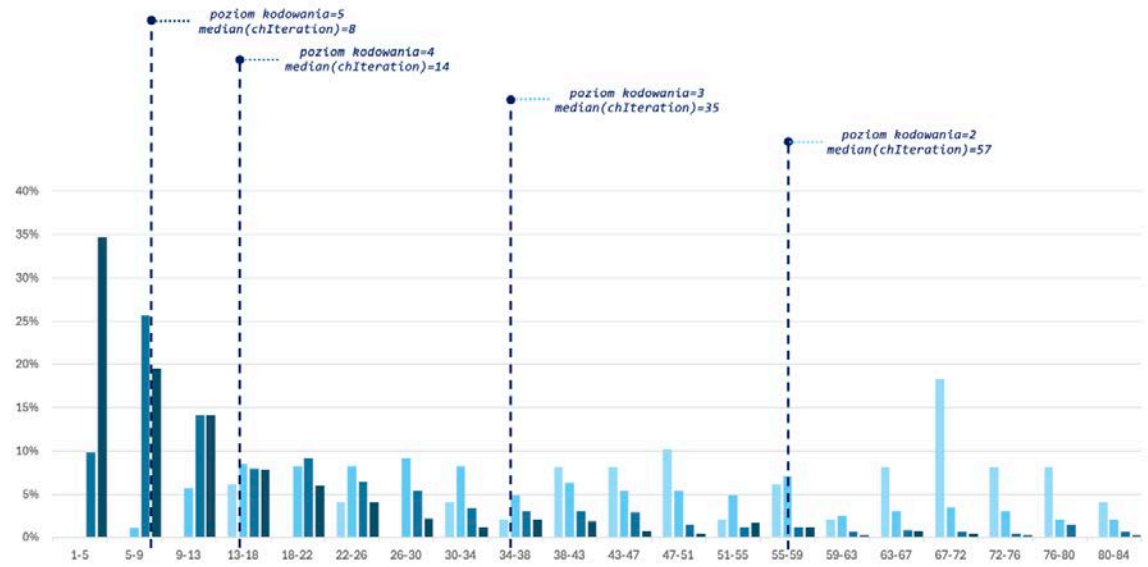


przykładowa *klatka steganogramu* poddanego wielokrotnym zniekształceniom

Rysunek 61 - Przykład: porównanie *klatek steganogramów z bazy oryginalnych steganogramów* (nie poddanych wielokrotnym zniekształceniom) *oraz bazy zniekształconych steganogramów* (poddanych wielokrotnym zniekształceniom)
Źródło: *Opracowanie własne*

W tabelach 43, 44, 45, 46 oraz 47 (umieszczonych w załączniku) przedstawione są wyniki obliczeń *chIteration* dla *steganogramów* z *bazy zniekształconych steganogramów* dla wszystkich *poziomów kodowania*. Na ich podstawie obliczono *histogramy* i wyznaczono mediany *chIteration* dla *bazy zniekształconych steganogramów* dla *poziomów kodowania* od 2 do 5, które przedstawione są na rysunku 62. Dla *poziomu kodowania* 1 nie udało się zdekodować *komunikatu* z żadnego ze *steganogramów* z *bazy zniekształconych steganogramów*.

[153] **Baza zniekształconych steganogramów** - baza plików *wideo* utworzona poprzez zastosowanie do *oryginalnych steganogramów* z *bazy oryginalnych steganogramów* wielu przekształceń i zniekształceń o różnym charakterze, tj. kompresja, rzut perspektywiczny, obrót, osadzenie na tle innego nośnika, zaszumienie.



Rysunek 62 - Wyniki: *histogramy chIteration* dla *bazy zniekształconych steganogramów*
Źródło: *Opracowanie własne*

4.3.3.5 Odporność metody RAI na transkodowanie

W tym eksperymencie zbadano *odporność* metody *RAI* na zastosowanie transkodowania przez domyślnie stosowany *kodek* serwisu *youtube.com*.

W tym celu najpierw stworzono *bazę wylosowanych steganogramów*[154] w taki sposób, że spośród numerów z przedziału od 1 do 666 wylosowano 30 liczb, a następnie do *bazy wylosowanych steganogramów* zapisano 150 plików z *bazy oryginalnych steganogramów* odpowiadających 30 wylosowanym numerom na wszystkich pięciu *poziomach kodowania.*

Następnie każdy *steganogram* z *bazy wylosowanych steganogramów* przesłano na serwer *youtube.com*, a potem pobrano z serwisu transkodowaną przez *kodeka youtube.com* wersję pliku *wideo* o zmniejszonej rozdzielczości 1280x720. 150 tak przekształconych plików *wideo* zapisano w *bazie transkodowanych steganogramów*[155].

Przesyłając i pobierając pliki *wideo* z serwisów internetowych takich jak np. youtube.com, nie można samodzielnie decydować o szczegółowych parametrach *kodeka*,

---

[154] **Baza wylosowanych steganogramów** - baza wylosowanych 150 *oryginalnych steganogramów* odpowiadających 30 wylosowanym numerom z wszystkimi pięcioma *poziomami kodowania*. *Baza wylosowanych steganogramów* stanowi bazę do utworzenia *bazy transkodowanych steganogramów* oraz *bazy końcowych steganogramów*.

[155] **Baza transkodowanych steganogramów** - baza plików *wideo* utworzonych poprzez transkodowanie przez *kodeka* serwisu youtube.com każdego *wideo* z *bazy wylosowanych steganogramów*.

w tym np. o stopniu stosowanej kompresji. Powstające w wyniku takiego transkodowania zniekształcenia *steganogramów* mogą mieć bardzo znaczący i nie do końca znany wpływ na niszczenie *informacji* tworzącej zakodowany *komunikat*.

Na rysunku 63 przedstawiony jest przykład tej samej *klatki oryginalnego steganogramu* i odpowiadającego mu *steganogramu* z *bazy transkodowanych steganogramów*. Choć przekształcenia i zniekształcenia wynikające z transkodowania nie są na pierwszy rzut oka widoczne, to degradują *informację* tworzącą zakodowany w *steganogramach komunikat.*

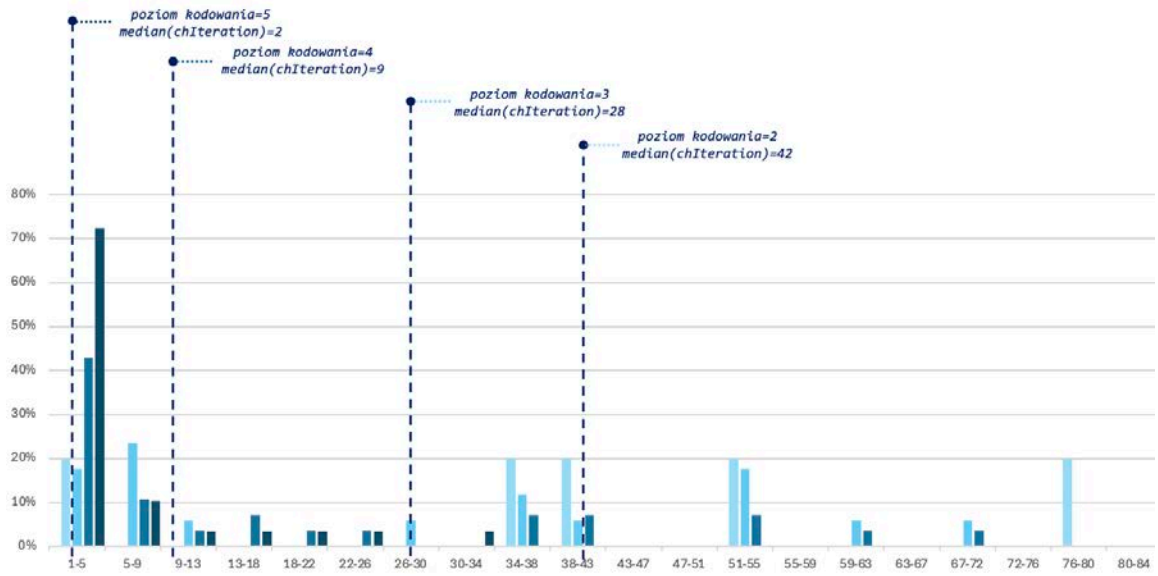
przykładowa *klatka steganogramu* nie poddanego transkodowaniu

przykładowa *klatka steganogramu* poddanego transkodowaniu

Rysunek 63 - Przykład: porównanie klatek *steganogramów z bazy oryginalnych steganogramów* (nie poddanych transkodowaniu) oraz *bazy transkodowanych steganogramów* (poddanych transkodowaniu)
Źródło: *Opracowanie własne*

W tabelach 48, 49, 50, 51 oraz 52 (umieszczonych w załączniku) przedstawione są wyniki obliczeń *chIteration* dla wszystkich *steganogramów* z *bazy transkodowanych steganogramów* dla wszystkich *poziomów kodowania*.

Na podstawie uzyskanych wyników obliczono *histogramy* i wyznaczono mediany *chIteration* dla *bazy transkodowanych steganogramów* dla *poziomów kodowania* od 2 do 5 przedstawione na rysunku 64. Dla *poziomu kodowania* 1 nie udało się zdekodować *komunikatu* z żadnego ze *steganogramów* z *bazy transkodowanych steganogramów*.



Rysunek 64 - Wyniki: *histogramy chIteration* dla *bazy transkodowanych steganogramów*
Źródło: *Opracowanie własne*

4.3.3.6 Odporność metody RAI na zniekształcenia analogowe

W tym eksperymencie zbadano *odporność* metody *RAI* na rzeczywisty *scenariusz zniekształceń analogowych*.

Każde *wideo* z *bazy wylosowanych steganogramów* wyświetlono na ekranie komputera w standardowych warunkach dziennego oświetlenia. Wyświetlane *wideo* na ekranie nagrano kamerą smartfona ze standardowymi parametrami i tak utworzono *bazę końcowych steganogramów*[156] liczącą łącznie 150 *steganogramów*, czyli 30 wylosowanych numerów ze wszystkimi pięcioma *poziomami kodowania*.

Na rysunku 65 przedstawiony jest przykład tej samej *klatki oryginalnego steganogramu* i odpowiadającego mu *steganogramu* z *bazy końcowych steganogramów*. W *końcowym steganogramie* widoczny jest na tle ogrodu ekran komputera, na którym wyświetlany jest *oryginalny steganogram*. Powstałe w ten sposób przekształcenia i zniekształcenia są nie tylko widoczne gołym okiem, ale również redukują istotnie *informację* tworzącą zakodowany w *steganogramach komunikat.*

[156] **Baza końcowych steganogramów** - baza plików *wideo* utworzonych w rzeczywistym *scenariuszu przerwanego kanału cyfrowego* poprzez nagranie kamerą smartfona wyświetlanych na ekranie komputera *steganogramów* z *bazy wylosowanych steganogramów*.

przykładowa *klatka steganogramu*
nie poddanego analogowym zniekształceniom

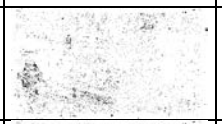
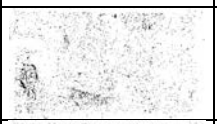
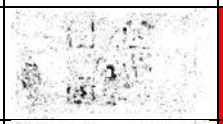
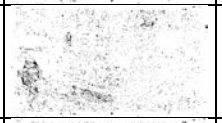
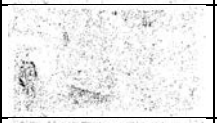
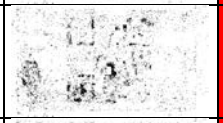
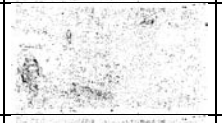
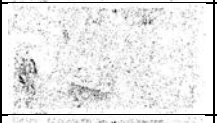
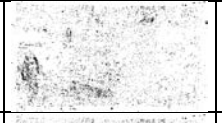
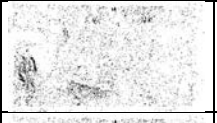
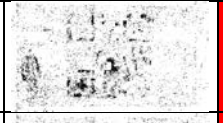
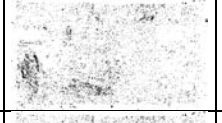
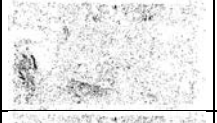
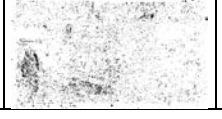
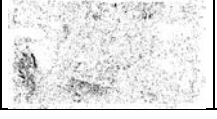
przykładowa *klatka steganogramu*
poddanego analogowym zniekształceniom

Rysunek 65 - Przykład: porównanie *klatek steganogramów z bazy oryginalnych steganogramów* (nie poddanych analogowym zniekształceniom) oraz *bazy końcowych steganogramów* (poddanych analogowym zniekształceniom)
Źródło: *Opracowanie własne*

W tabeli 16 przedstawiony jest przykład procesu przyrostowego zwiększania *informacji* w dekodowanym *komunikacie* w postaci kodu *QR* dla *końcowego steganogramu* zakodowanego różnymi *poziomami kodowania*. Podobnie, jak poprzednio w przypadku *oryginalnych steganogramów*, dla każdej *iteracji dekodującej* wraz z wzrostem ilości *informacji* dekodowanej ze *steganogramów*, kod *QR* reprezentujący *komunikat* jest coraz bardziej wyraźny, aż ostatecznie przy *iteracji* równej *chIteration* ilość zgromadzonej *informacji* pozwala zdekodować *komunikat*.

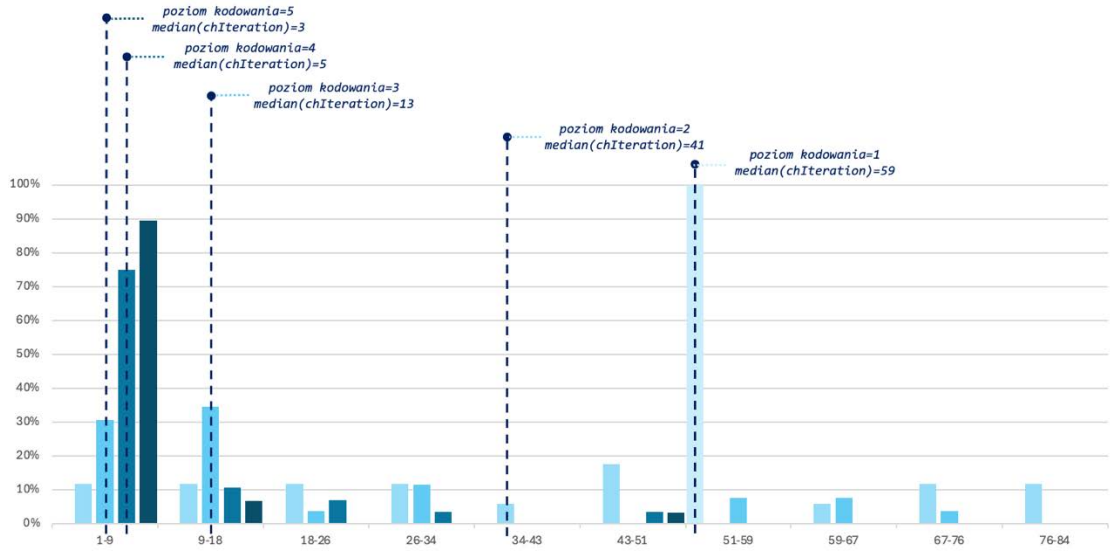
Tabela 16 - Przykład: przyrostowe zwiększanie *informacji* zakodowanej w *komunikacie* w *końcowych steganogramach* w kolejnych iteracjach *adaptacyjnego algorytmu dekodującego RAI* z zaznaczonymi iteracjami >= *chIteration*, dla których występuje zdekodowanie *komunikatu*
Źródło: *Opracowanie własne*

Iteracja	Poziom kodowania 1	Poziom kodowania 2	Poziom kodowania 3	Poziom kodowania 4	Poziom kodowania 5
1					
2					
3					
4					

Iteracja	Poziom kodowania 1	Poziom kodowania 2	Poziom kodowania 3	Poziom kodowania 4	Poziom kodowania 5
5					
6					
7					
8					
9					
10					

W tabelach 53, 54, 55, 56 oraz 57 (umieszczonych w załączniku) przedstawione są wyniki obliczeń *chIteration* dla wszystkich *steganogramów* z *bazy końcowych steganogramów* dla wszystkich pięciu *poziomów kodowania*.

Na podstawie uzyskanych wyników obliczono *histogramy* i mediany *chIteration* dla *bazy końcowych steganogramów*, które zostały przedstawione na rysunku 66.



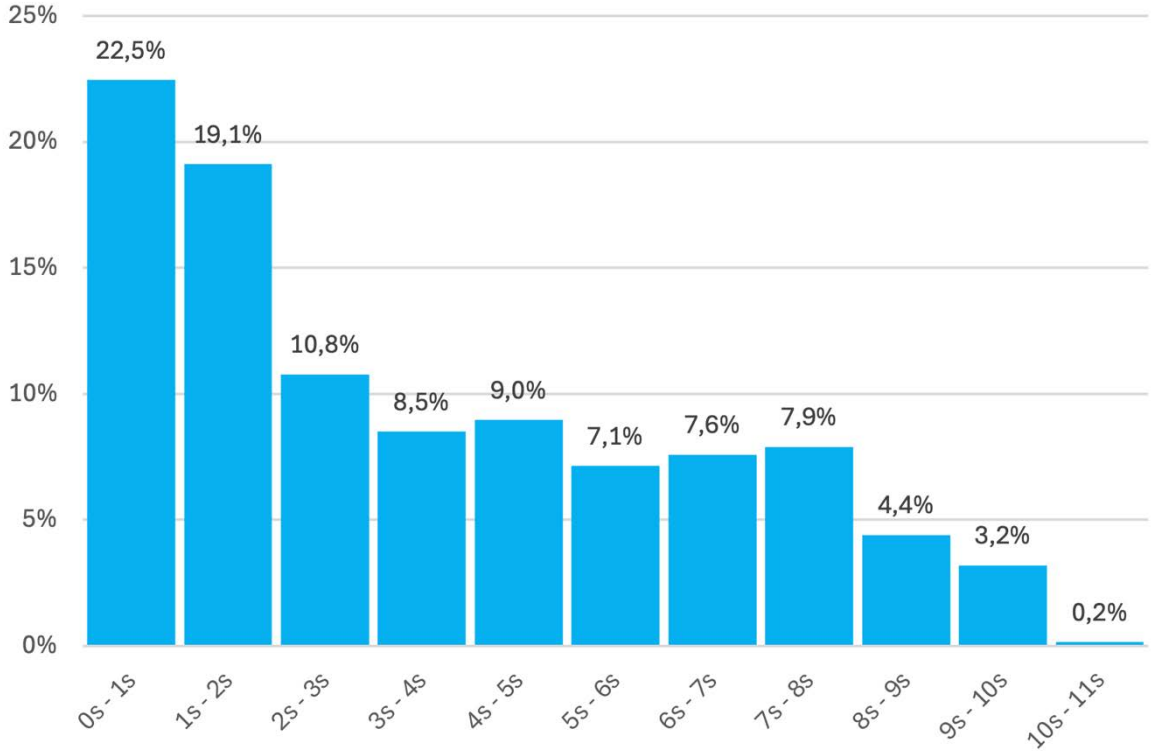
Rysunek 66 - Wyniki: *histogramy chIteration* dla *bazy końcowych steganogramów*
Źródło: *Opracowanie własne*

4.3.4 Badanie czasu dekodowania w metodzie RAI

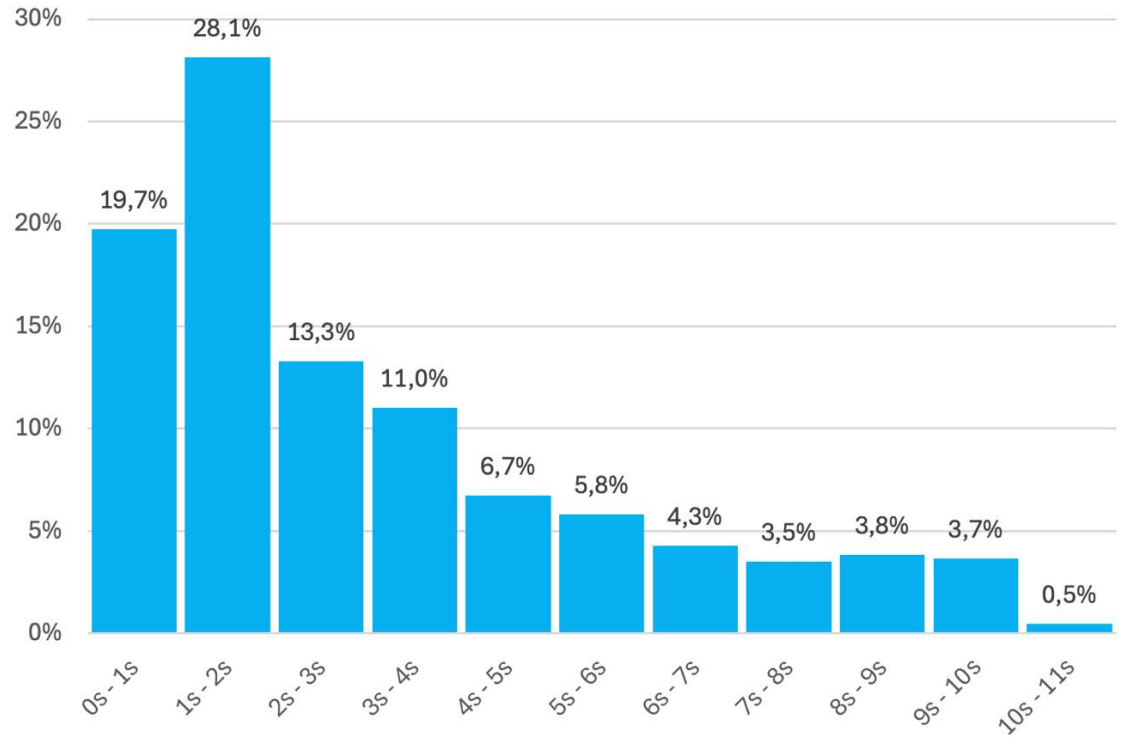
Z każdej bazy *steganogramów* wybrano *steganogramy* o *poziomie kodowania* równym *chLevel*, czyli najmniejszym *poziomie kodowania*, który pozwala na zdekodowanie *komunikatu*. Dla tak wybranych *steganogramów* obliczono wartości charakterystyczne czasu dekodowania *komunikatów chTime*.

Wyniki *chTime* dla poszczególnych baz *steganogramów* zamieszczono w tabelach 58, 59, 60, 61, 62, 63 oraz 64 (umieszczonych w załączniku).

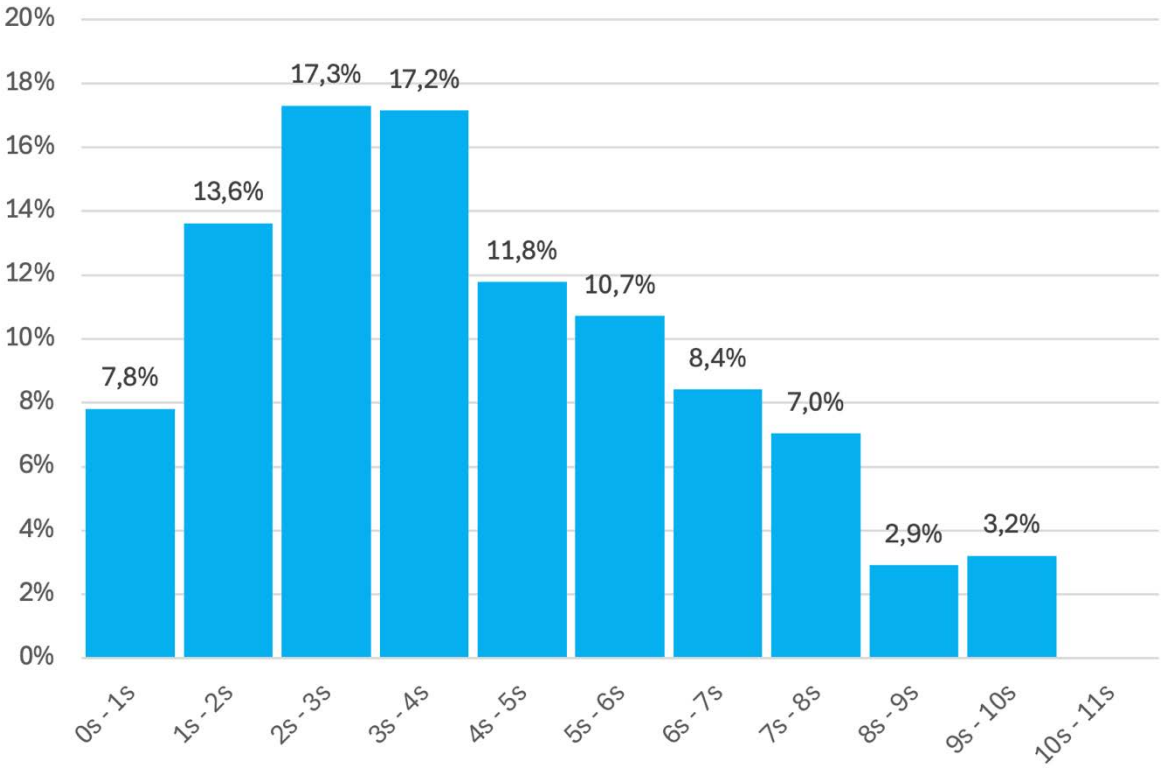
Na podstawie uzyskanych wartości obliczono odpowiadające im *histogramy*, które pokazano na rysunkach 67, 68, 69, 70, 71, 72 oraz 73.



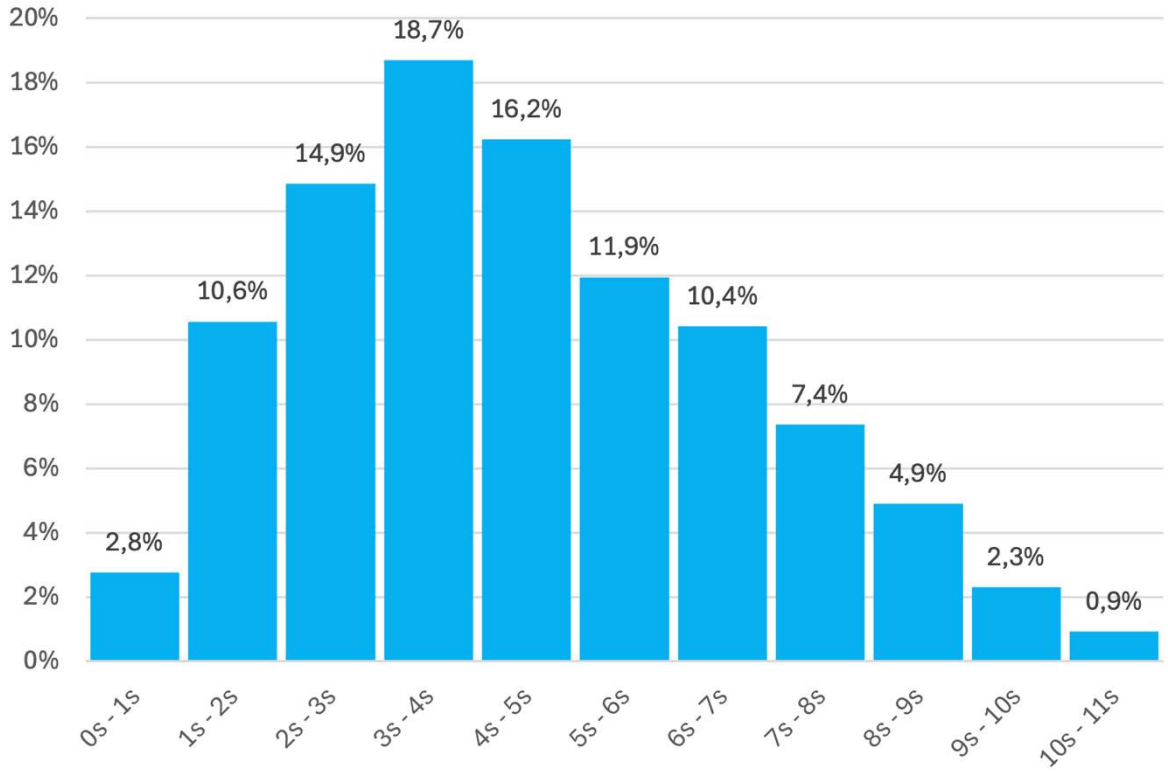
Rysunek 67 - Wyniki: histogram *chTime* dla *bazy oryginalnych steganogramów*
Źródło: *Opracowanie własne*



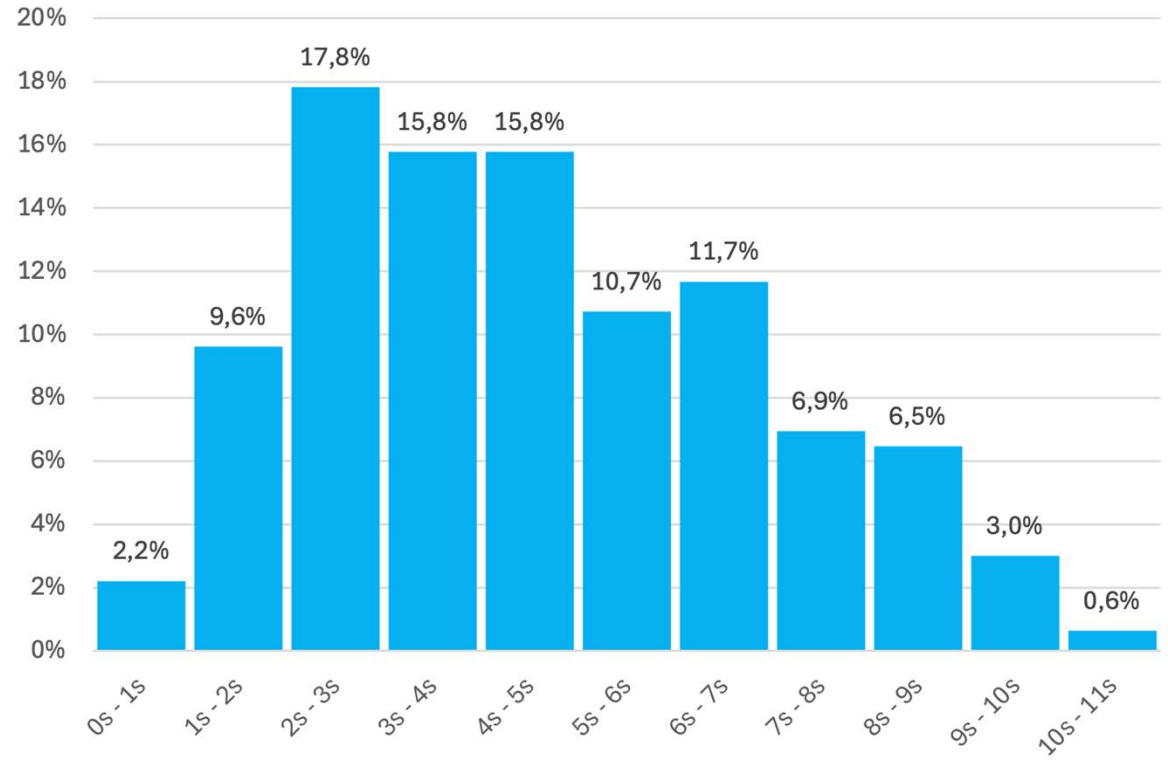
Rysunek 68 - Wyniki: histogram *chTime* dla *skompresowanych steganogramów*
Źródło: *Opracowanie własne*



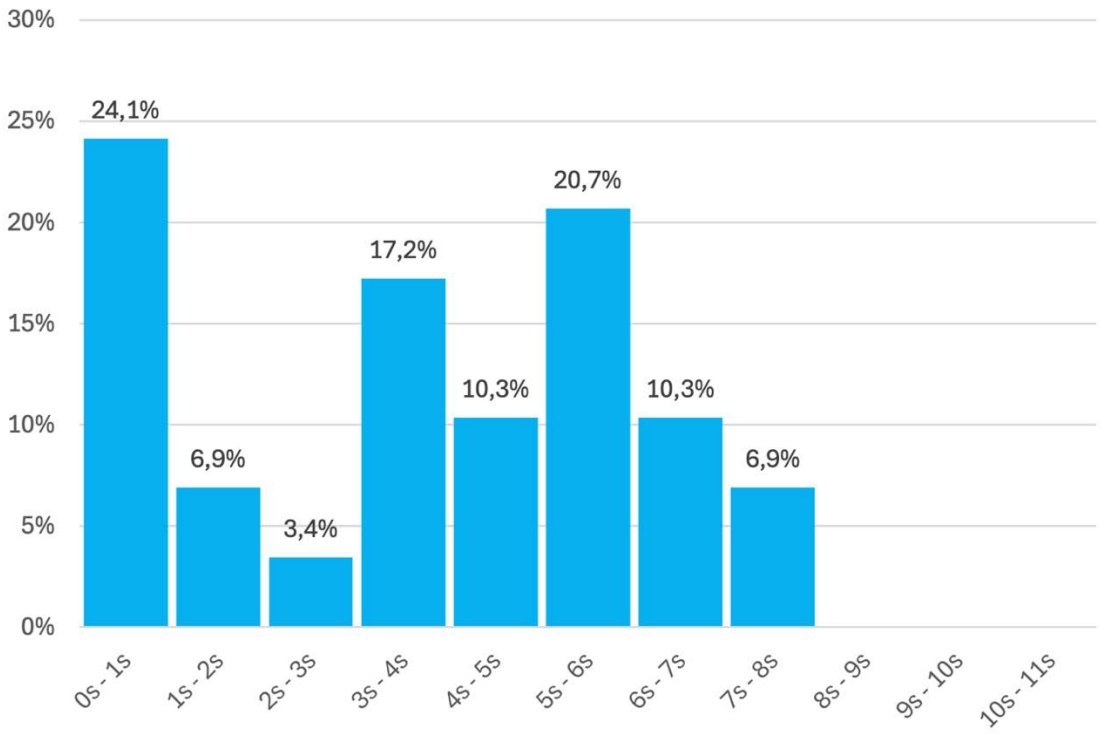
Rysunek 69 - Wyniki: histogram *chTime* dla *transformowanych steganogramów*
Źródło: *Opracowanie własne*



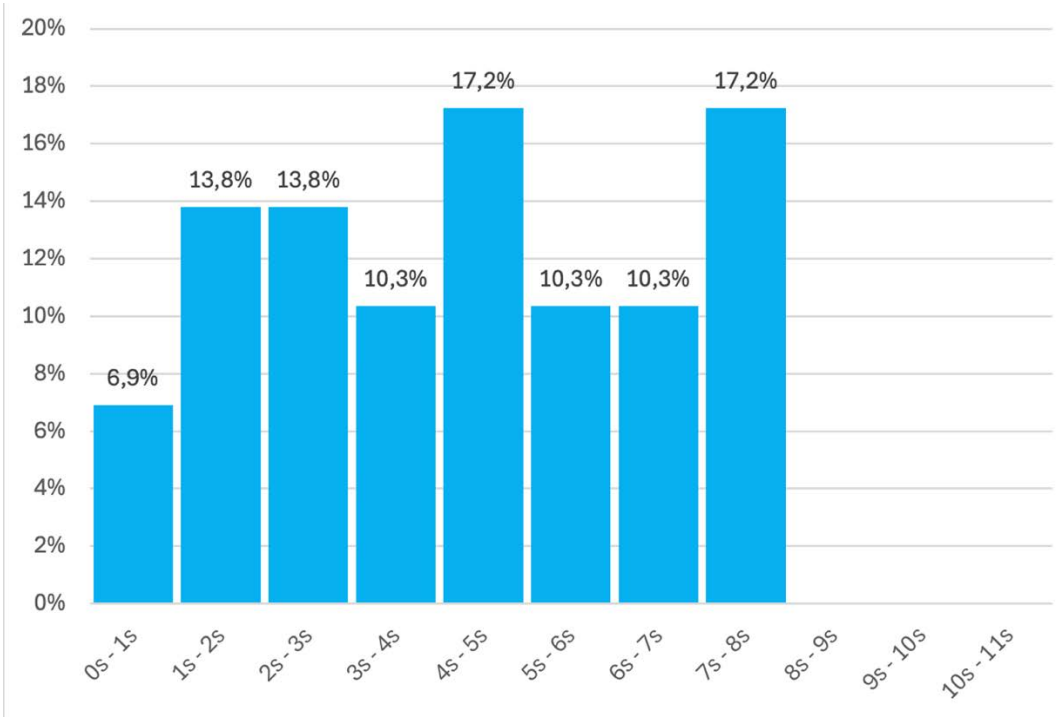
Rysunek 70 - Wyniki: histogram *chTime* dla *bazy zaszumionych steganogramów*
Źródło: *Opracowanie własne*



Rysunek 71 - Wyniki: histogram *chTime* dla *bazy zniekształconych steganogramów*
Źródło: *Opracowanie własne*

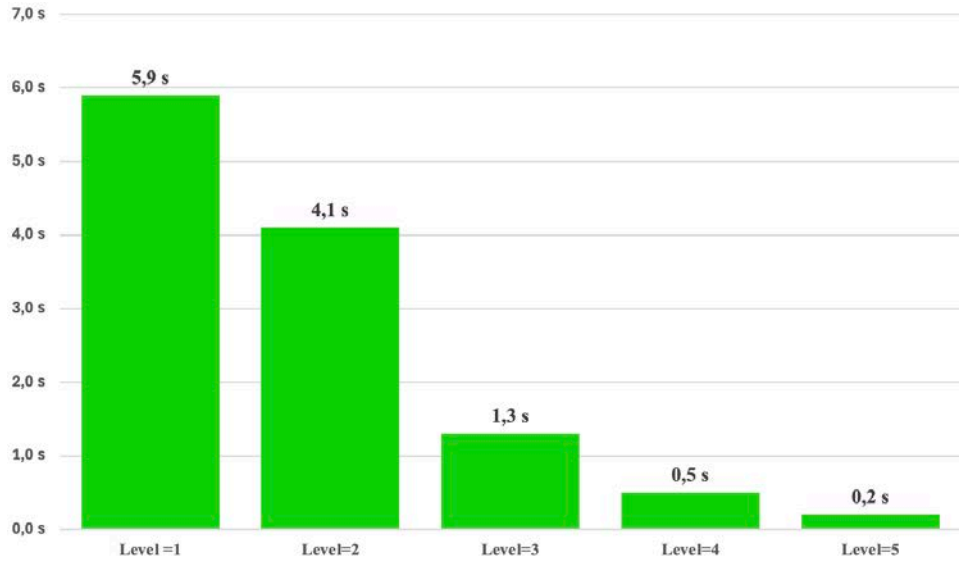


Rysunek 72 - Wyniki: histogram *chTime* dla *bazy transkodowanych steganogramów*
Źródło: *Opracowanie własne*



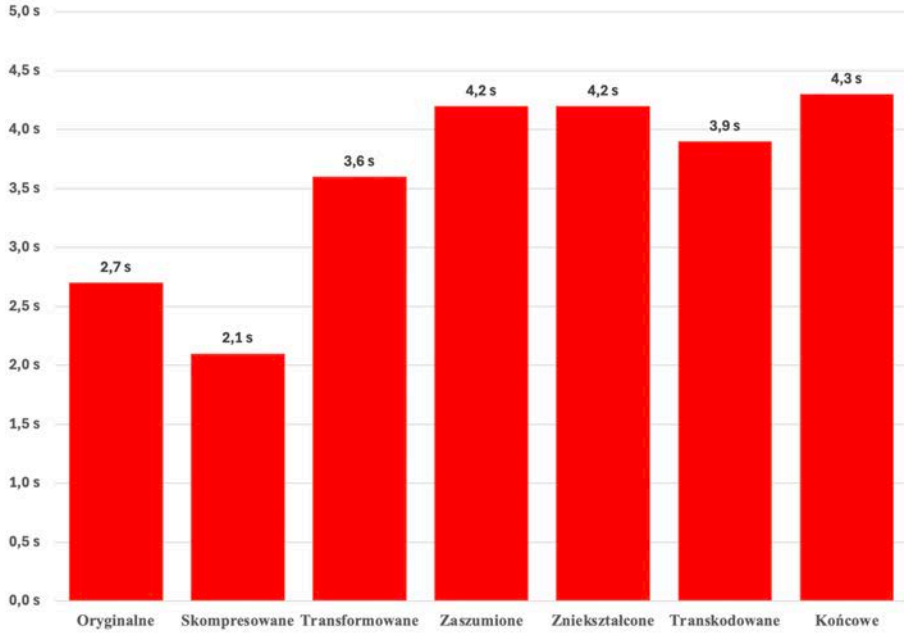
Rysunek 73 - Wyniki: histogram *chTime* dla *bazy końcowych steganogramów*
Źródło: *Opracowanie własne*

Na rysunku 74 przedstawiono mediany wartości charakterystycznych *chTime* dla poszczególnych *poziomów kodowania* obliczone ze wszystkich baz *steganogramów*.



Rysunek 74 - Wyniki: wartości mediany *chTime* dla różnych *poziomów kodowania*
Źródło: *Opracowanie własne*

Na rysunku 75 przedstawiono mediany wartości *chTime* dla poszczególnych baz *steganogramów* obliczone dla wszystkich *poziomów kodowania*.



Rysunek 75 - Wyniki: wartości mediany *chTime* dla różnych baz *steganogramów*
Źródło: *Opracowanie własne*

W tabeli 17 przedstawiono obliczone wartości charakterystyczne *chTime* wraz z medianami dla wszystkich baz *steganogramów* oraz wszystkich *poziomów kodowania*.

Tabela 17 - Wyniki: wartości mediany *chTime* dla różnych baz *steganogramów* oraz różnych *poziomów kodowania*
Źródło: *Opracowanie własne*

Baza steganogramów	Poziom kodowania = 1	Poziom kodowania = 2	Poziom kodowania = 3	Poziom kodowania = 4	Poziom kodowania = 5	mediana chTime
Oryginalne	5,3 s	0,8 s	0,3 s	0,2 s	0,1 s	2,7 s
Skompresowane	6,2 s	1,3 s	0,5 s	0,2 s	0,1 s	2,1 s
Transformowane		3,9 s	0,9 s	0,3 s	0,2 s	3,6 s
Zaszumione		4,5 s	1,4 s	0,6 s	0,4 s	4,2 s
Zniekształcone		5,7 s	3,5 s	1,4 s	0,8 s	4,2 s
Transkodowane		4,2 s	2,8 s	0,9 s	0,2 s	3,9 s
Końcowe	5,9 s	4,1 s	1,3 s	0,5 s	0,3 s	4,3 s
mediana chTime	5,9 s	4,1 s	1,3 s	0,5 s	0,2 s	3.5 s

Mediana wartości charakterystycznych *chTime* dla wszystkich *steganogramów* z wszystkich baz *steganogramów* i wszystkich *poziomów kodowania* wyniosła 3,5 s.

4.3.5 Badanie pojemności metody RAI

Uwzględniając przyjęte założenie, że kodowany w przeprowadzanych eksperymentach *komunikat* ma długość 72 *bitów* oraz biorąc pod uwagę obliczone we wcześniejszym rozdziale wartości charakterystyczne czasów dekodowania *komunikatu chTime* dla poszczególnych baz *steganogramów*, poszczególnych *poziomów kodowania* oraz dla wszystkich *steganogramów*, obliczono, zgodnie z przyjętą definicją, wartości charakterystyczne *pojemności chCapacity,* a wyniki umieszczono w tabeli 18.

Tabela 18 - Wyniki: wartości mediany *chCapacity* dla różnych baz *steganogramów* oraz różnych *poziomów kodowania*
Źródło: *Opracowanie własne*

	mediana chCapacity
Poziom kodowania 1	12,2 b/s
Poziom kodowania 2	17,6 b/s
Poziom kodowania 3	55,4 b/s
Poziom kodowania 4	144,0 b/s
Poziom kodowania 5	360,0 b/s

	mediana chCapacity
Baza oryginalnych steganogramów	26,7 b/s
Baza skompresowanych steganogramów	34,3 b/s
Baza transformowanych steganogramów	20,0 b/s
Baza zaszumionych steganogramów	17,1 b/s
Baza zniekształconych steganogramów	17,1 b/s
Baza transkodowanych steganogramów	18,5 b/s
Baza końcowych steganogramów	16,7 b/s
mediana chCapacity	***20,6 b/s***

4.3.6 Badanie niewykrywalności metody RAI

W *scenariuszu zniekształceń analogowych niewykrywalność* jest miarą określającą stopień trudności odkrycia przy użyciu ludzkich zmysłów, w tym wypadku wzroku, faktu zakodowania *komunikatu* w *steganogramie wideo*. W celu oceny skuteczności metody *RAI* w kontekście *niewykrywalności* przeprowadzono badanie eksperymentalne z udziałem losowo wybranej próby osób.

Celem badania była empiryczna ocena zdolności percepcyjnych uczestników do identyfikacji modyfikacji *steganograficznych* w *wideo*. W szczególności, celem była odpowiedź na pytanie, przy jakim maksymalnym *poziomie kodowania steganogramów wideo* w metodzie *RAI*, modyfikacje *steganograficzne* w *wideo* są dla obserwatorów *niewykrywalne*.

4.3.6.1 Schemat badania

Badanie zostało przeprowadzone przy użyciu zaprojektowanego i skonstruowanego systemu informatycznego, którego celem jest badanie różnic między progiem ludzkiej percepcji zmian w plikach *wideo* wprowadzanych przez *techniki steganografii wideo*, a minimalnym wymaganym *poziomem kodowania* ukrytego *komunikatu* umożliwiającym automatycznego zdekodowanie ukrytego *komunikatu*. Architektura i przykładowa implementacja systemu została opisana w pracy *(Pery i Waszkowski, Computational System for Evaluating Human Perception in Video Steganography, 2024)*.

4.3.6.1.1 Uczestnicy badania

W badaniu wzięły udział 72 osoby w wieku 18-55 lat, reprezentujące zróżnicowaną grupę pod względem płci, wieku oraz doświadczenia w pracy z grafiką, *wideo* lub fotografią. Struktura grupy:

- kobiety: 31 osób, mężczyźni: 41 osób
- wiek: 18-30: 11 osób, 31-45: 25 osób, 46-55: 36 osób
- osoby z doświadczeniem w obróbce obrazu: 27 osób

Uczestnicy zostali zakwalifikowani do badania po pozytywnym wyniku testu *Ishihary*[157] potwierdzającym brak wad wzroku takich jak *daltonizm*[158].

4.3.6.1.2 Materiały badawcze

Dla każdego z 29 *steganogramów* z *bazy końcowych steganogramów*, dla których określono wartość charakterystyczną *poziomu kodowania chLevel*, wygenerowano 29 zestawów *oryginalnych steganogramów* zakodowanych z *poziomami kodowania* od 1 do 10.

Przygotowano również oprogramowanie *(Pery i Waszkowski, Computational System for Evaluating Human Perception in Video Steganography, 2024)* do losowego wyświetlania *wideo* spośród wybranych 29 *steganogramów* oraz rejestrowania odpowiedzi uczestników w postaci dychotomicznych wskazań, czy według nich wyświetlone *wideo* było zmodyfikowane, czy nie.

4.3.6.1.3 Procedura badawcza

1. Rekrutacja i selekcja uczestników.

   Uczestnicy zostali wybrani zgodnie z opisanymi powyżej kryteriami.

2. Prezentacja *wideo*.

   Każdy uczestnik oglądał na ekranie komputera w losowej kolejności poszczególne *oryginalne steganogramy* od *poziomu kodowania* 1 - najmniej zmodyfikowanego, do *poziomu kodowania* 10 - najbardziej zmodyfikowanego. Prezentacja *wideo* odbyła się

---

[157] **Ishihara test** - narzędzie do diagnozy daltonizmu, składające się z tablic z kolorowymi kropkami tworzącymi liczby lub kształty. Osoby z prawidłowym widzeniem rozpoznają je, podczas gdy osoby z zaburzeniami barw mają trudności. Test pozwala określić rodzaj i stopień deficytu widzenia barw (Ishihara, 1917).

[158] **Daltonizm** - zaburzenie widzenia barw polegające na trudności w rozróżnianiu kolorów, najczęściej czerwonego i zielonego. Jest to dziedziczna wada genetyczna związana z mutacją chromosomu X, przez co częściej dotyka mężczyzn *(Gordon, 1998)*.

w warunkach otoczenia analogicznych do procedury tworzenia *bazy końcowych steganogramów*. Po każdym *wideo* uczestnik odpowiedział na pytanie: „Czy ten materiał wideo wydaje się naturalny, czy zmodyfikowany?”. Każda odpowiedź była zapisywana jako rozstrzygnięcie dychotomiczne („zmodyfikowany” lub „niezmodyfikowany”). W przypadku, gdy dla danego *poziomu kodowania* uczestnik odpowiadał, że *wideo* wydaje mu się zmodyfikowane, badanie danego *steganogramu* na tym *poziomie kodowania* się kończyło i następowało przejście do kolejnego losowo wybranego *steganogramu*.

3. Gromadzenie danych.

   Zebrane zostały odpowiedzi uczestników dotyczące identyfikacji zmian w postaci określenia dla każdego *oryginalnego steganogramu* takiego *poziomu kodowania*, przy którym rozpoznawali wprowadzone w *wideo* modyfikacje *steganograficzne*. 1 decyl tak obliczonych wartości *poziomów kodowania* dla poszczególnych *oryginalnych steganogramów* nazwano *percLevel*[159].

Wartości *percLevel* dla poszczególnych *steganogramów wideo* oznaczają najmniejszy *poziom kodowania* tych *oryginalnych steganogramów*, przy którym co najmniej 10% badanych osób widzi wprowadzone w *wideo* modyfikacje *steganograficzne*. Dla wszystkich *poziomów kodowania* poniżej wartości *percLevel* modyfikacje *steganograficzne* są niezauważalne dla co najmniej 90% badanych osób.

4.3.6.2 Otrzymane wyniki

W tabeli 19 przedstawiono wyniki przeprowadzonych badań na próbie osób z wyszczególnionymi dla każdego badanego *steganogramu* wartościami *chLevel - poziomu kodowania*, przy którym następuje automatyczne *zdekodowanie komunikatu* z *końcowych steganogramów*, *percLevel - poziomu kodowania*, przy którym następuje rozpoznanie modyfikacji w *oryginalnych steganogramach* przez osoby badane oraz obliczoną różnicą *percLevel-chLevel* obrazującą "bezpieczny dystans" pomiędzy granicami *niewykrywalności* i możliwości zdekodowania *komunikatu* ze *steganogramu*.

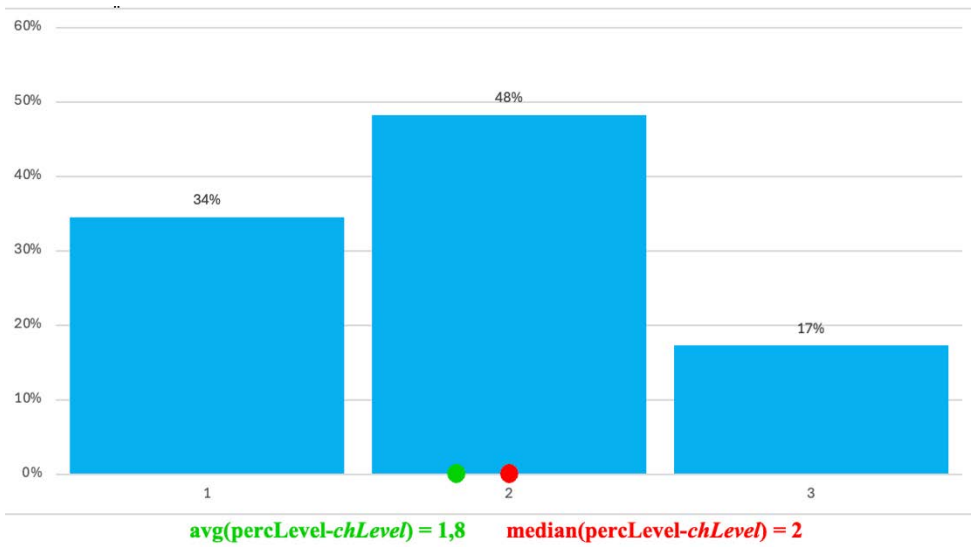
---

[159] **percLevel** - najmniejszy *poziom kodowania steganogramu wideo* utworzonego za pomocą techniki *RAI*, przy którym więcej niż 10% badanych osób widzi wprowadzone w *wideo* modyfikacje *steganograficzne*.

Tabela 19 - Wyniki: wartości *chLevel* oraz *percLevel* dla poszczególnych *steganogramów* obliczone na podstawie wskazań osób badanych rozpoznających modyfikacje w *steganogramach* zakodowanych *podstawowym algorytmem kodującym c-RAI*
Źródło: *Opracowanie własne*

steganogram	***chLevel*** *poziom kodowania, przy którym następuje automatyczne zdekodowanie komunikatu z końcowego steganogramu*	***percLevel*** *poziom kodowania, przy którym następuje rozpoznanie modyfikacji przez osoby badane w oryginalnym steganogramie*	***percLevel-chLevel***
011	3	5	2
031	3	6	3
037	2	5	3
111	4	5	1
117	2	5	3
123	3	6	3
196	4	6	2
208	2	4	2
235	3	4	1
238	3	4	1
281	4	6	2
282	3	5	2
308	3	5	2
410	3	4	1
417	4	7	3
442	2	4	2
450	4	5	1
457	4	6	2
484	3	5	2
509	3	5	2
523	4	5	1
533	4	5	1
540	3	5	2
547	4	5	1
554	2	4	2
563	4	5	1
568	3	5	2
571	4	5	1
652	4	6	2

Na rysunku 76 przedstawiono *histogram* różnic pomiędzy wartościami *percLevel* i *chLevel* dla badanych *steganogramów* wraz z obliczoną medianą wynoszącą 2 oraz średnią wynoszącą 1,8.



Rysunek 76 - Wyniki: *histogram* różnic pomiędzy wartościami *percLevel* oraz *chLevel* dla badanych *steganogramów* wraz z obliczoną medianą oraz średnią
Źródło: *Opracowanie własne*

4.4 Proponowana metoda steganoanalizy RAI

Oczywistą metodą *steganoanalityczną*, którą można wykorzystać do wykrycia zastosowania metody c-*RAI* wraz z jednoczesnym zdekodowaniem *komunikatu,* jest *algorytm dekodujący d-RAI* zaproponowany wcześniej. Jednakże zastosowanie tego algorytmu wymaga dużej wiedzy o konkretnych parametrach metody c-*RAI* zastosowanej do zakodowania *komunikatu* w *steganogramie*. Jeżeli dysponuje się taką wiedzą, to metoda *steganoanalityczna* wykorzystująca wprost *algorytm dekodujący d-RAI* jest najlepszym wyborem. W takim przypadku zgodnie z zaprezentowaną wcześniej kategoryzacją ma się do czynienia ze *steganoanalizą statystyczną*, *steganoanalizą ekstrakcyjną* oraz *steganoanalizą wykorzystującą wiedzę o metodzie steganograficznej*.

W przypadku, gdy wiedza o zastosowanych parametrach metody *steganografii* jest bardzo ograniczona albo żadna, a celem jest odpowiedzieć na pytanie, czy w danym *wideo* jest ukryty *komunikat* z zastosowaniem metody c-*RAI*, trzeba opracować nową metodę *steganoanalizy detekcyjnej*. Wprawdzie nie będzie ona w pełni *steganoanalizą ślepą* (bo będzie wykorzystywała wiedzę, iż metoda c-*RAI* do kodowania *komunikatu* używa *masek*

różnic kolorystycznych pikseli kolejnych *klatek steganogramu*), ale będzie istotnie zbliżała się do założenia o braku jakiejkolwiek wiedzy o zastosowanej metodzie *steganograficznej*.

Ponieważ metoda c-*RAI* optymalizuje *odporność* kosztem m.in. *niewykrywalności*, automatyczne wykrycie zastosowania metody c-*RAI* w przypadku dysponowania *końcowym steganogramem* jest stosunkowo proste i bazuje na wykorzystaniu analizy histogramów rozkładów częstotliwości *masek* pikseli pomiędzy kolejnymi *klatkami końcowego steganogramu*. Jeszcze prostsze jest wykrycie użycia metody c-*RAI* w przypadku dysponowania *oryginalnym steganogramem*. W tym przypadku różnice w rozkładach częstotliwości poszczególnych *masek* są jeszcze wyraźniejsze niż w przypadku *końcowego steganogramu*.

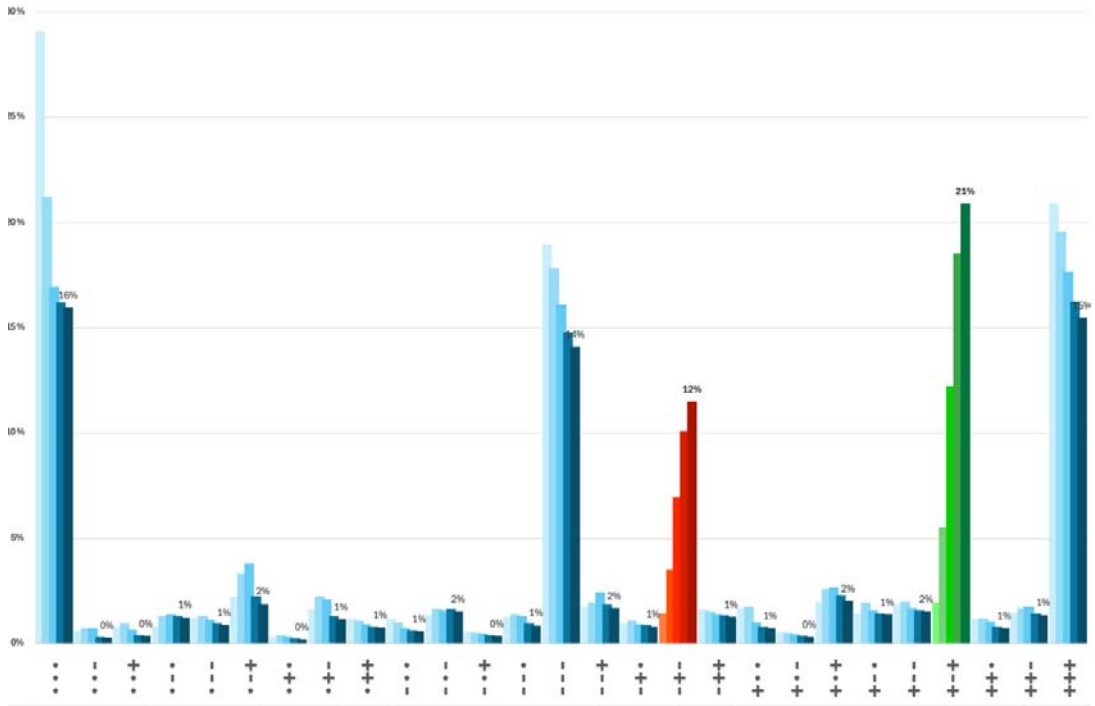
Kod 20 zawiera przykładowy algorytm obliczający częstotliwość występowania zadanej *maski* w *wideo*.

```
def RAI_mask_frequency(video, number_of_iterations, mask):
    """
    Oblicza częstotliwość występowania maski między kolejnymi klatkami.

    Argumenty:
    - video (cv2.VideoCapture): wideo, które ma być analizowane
    - number_of_iterations (int): liczba iteracji/klatek do analizy
    - mask (np.array): maska stosowana do kodowania (np. mask_1, mask_0)

    Wynik:
    - mask_frequency (np.array): tablica z częstotliwościami występowania maski
    """

    # Pobieramy wymiary wideo (wysokość i szerokość klatki)
    height = int(video.get(cv2.CAP_PROP_FRAME_HEIGHT))
    width = int(video.get(cv2.CAP_PROP_FRAME_WIDTH))

    # Obliczamy liczbę pikseli w klatce (potrzebne do normalizacji)
    number_of_pixels = height * width

    # Inicjalizujemy licznik i tablicę do przechowywania częstotliwości maski
    frame_count = 0
    mask_frequency = np.zeros((number_of_iterations), np.float64)

    # Odczytujemy pierwszą klatkę wideo
    ret, frame_previous = video.read()

    # Pętla przetwarzająca każdą kolejną klatkę wideo
    while video.isOpened() and frame_count < number_of_iterations:

        # Odczyt kolejnej klatki
        ret, frame_current = video.read()

        # Warunek zakończenia przetwarzania, gdy brak nowych ramek
        if not ret:
            break
```

```
        # Obliczamy różnicę między bieżącą a poprzednią klatką
        diff = frame_current.astype(np.int16) - frame_previous.astype(np.int16)

        # Tworzymy maski dla obliczonych różnic
        diff_masked = np.ones_like(diff, np.uint8)
        diff_masked[diff < 0] = 2
        diff_masked[diff > 0] = 4

        # Obliczamy, jaka część pikseli ma odpowiednie wartości maski
        mask_frequency[frame_count] = np.count_nonzero(
            np.all(diff_masked & mask != 0, axis=2)) / number_of_pixels

        frame_count += 1
        frame_previous = frame_current.copy()

    # Zwracamy tablicę z częstotliwościami maski
    return mask_frequency
```

Kod 20 - Algorytm obliczania częstotliwości występowania *masek* w *wideo*
Źródło: *Opracowanie własne*

Na rysunku 77 przedstawiony jest *histogram* częstotliwości występowania *masek* w *steganogramach*. W porównaniu do *histogramu* występowania *masek* w *nośnikach,* przedstawionego na rysunku 48, widać wyraźnie większą częstotliwość występowania *masek kodujących* zależną od *poziomu kodowania*.



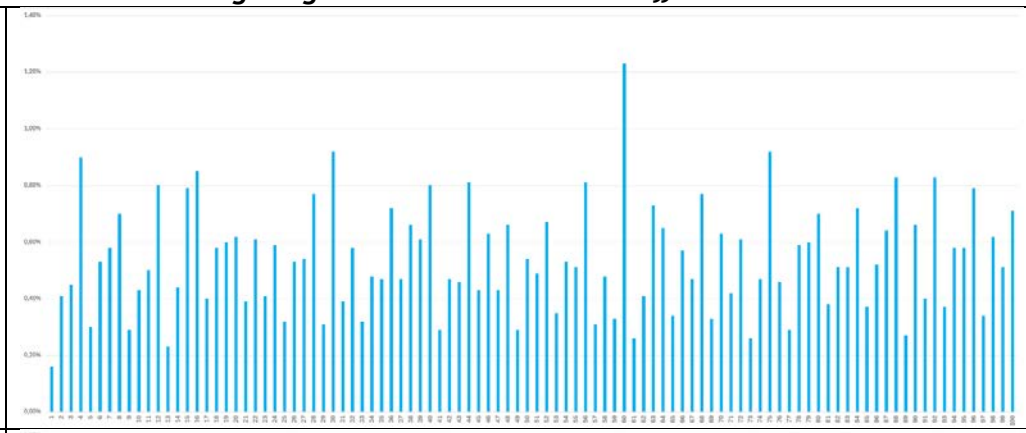
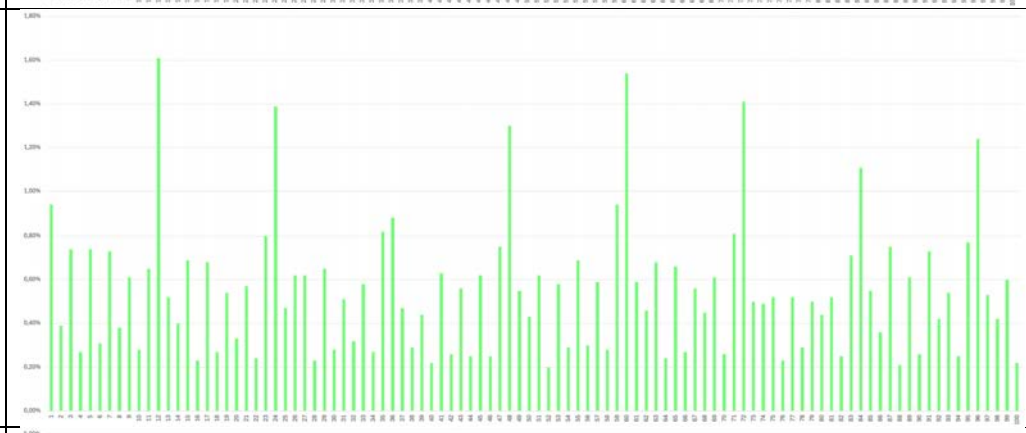
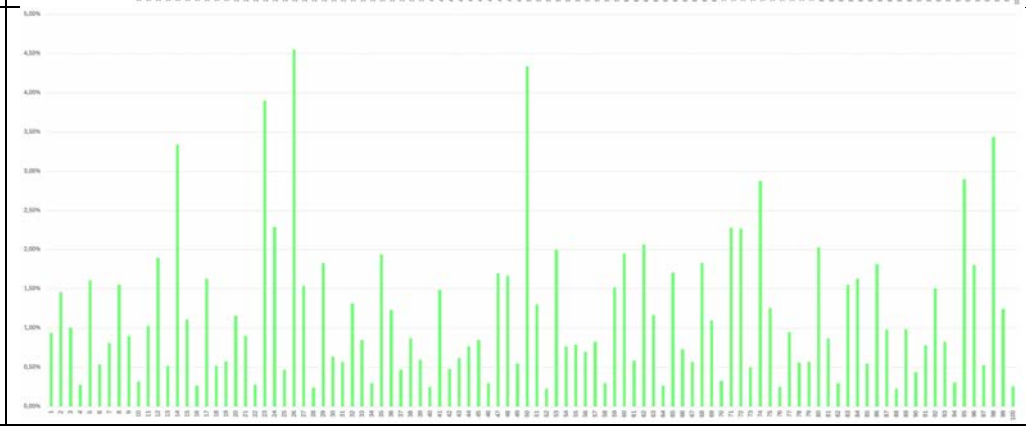
Rysunek 77 - Wyniki: histogram częstotliwości występowania *masek* w *steganogramach* *poziom kodowania* 1 - najjaśniejszy kolor, *poziom kodowania* 5 - najciemniejszy kolor na czerwono - *maska kodująca* „-+-”, na zielono - *maska kodująca* „+-+”
Źródło: *Opracowanie własne*

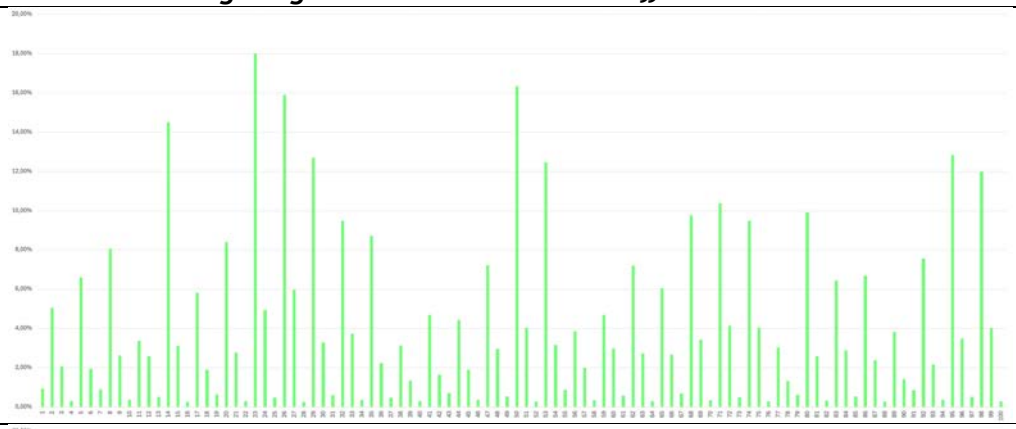
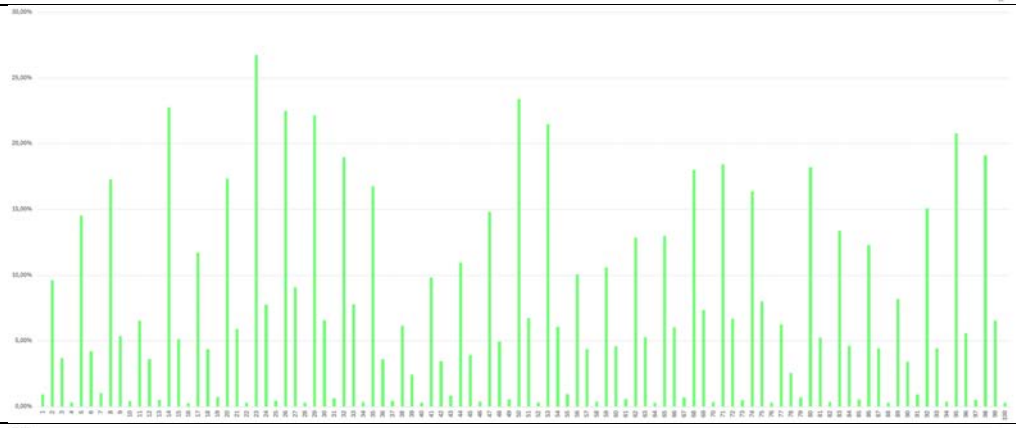
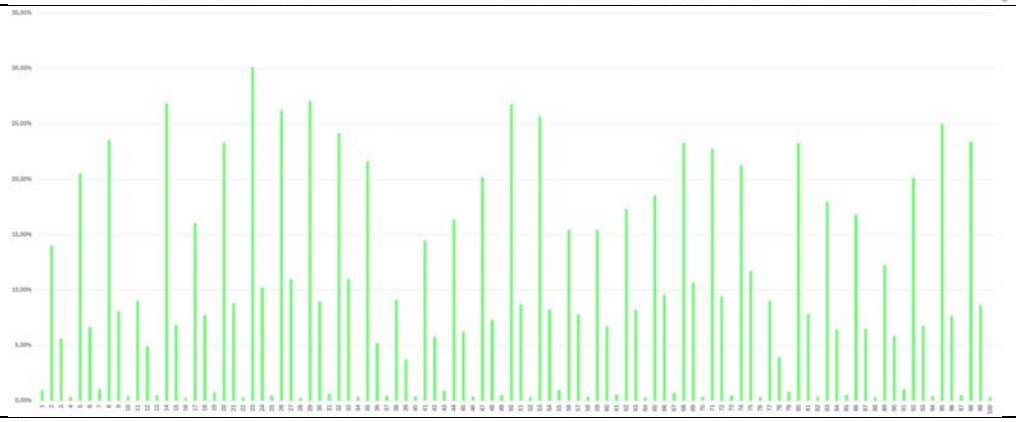
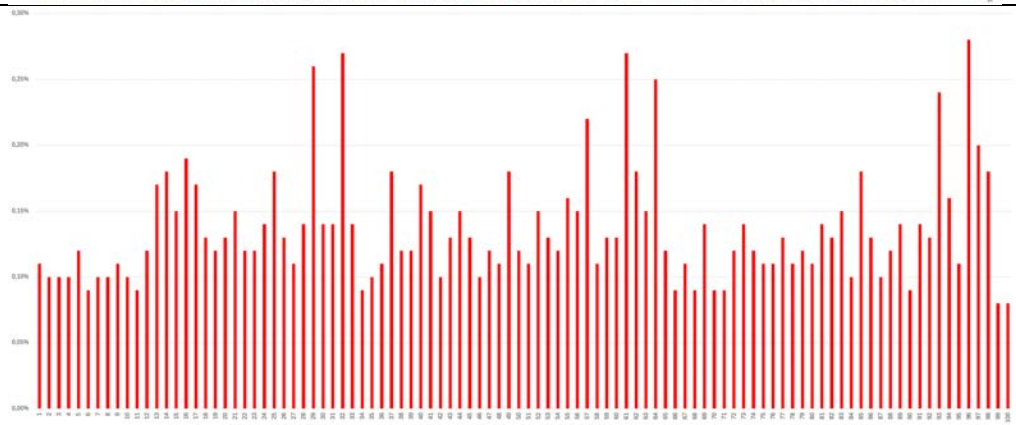
Wskaźnikiem, który pozytywnie koreluje z prawdopodobieństwem zakodowania w *steganogramie komunikatu* metodą *RAI* jest *odchylenie standardowe*[160] σ rozkładu częstotliwości *masek* kodowania. Poniżej zaprezentowano przykładowe *histogramy* częstotliwości występowania *masek kodowania* używanych w *podstawowym algorytmie kodowania c-RAI* dla *nośnika* oraz *steganogramów* zakodowanych z coraz większym *stopniem kodowania* wraz z wartością *odchylenia standardowego*. Na wykresach przedstawionych w tabelach 20 oraz 21 widać, że zwiększanie *stopnia kodowania* zmienia rozkład częstotliwości *masek kodowania* oraz zwiększa *odchylenie standardowe* obliczone

[160] **Odchylenie standardowe** - miara rozproszenia danych wokół średniej w zbiorze. Określa, jak bardzo poszczególne wartości różnią się od średniej arytmetycznej. Mała wartość odchylenia standardowego wskazuje, że większość danych jest blisko średniej, podczas gdy duża wartość oznacza większe rozproszenie wyników *(Kenney, 1954)*.

dla tych rozkładów. Kolorem niebieskim oznaczono *histogramy* dla *nośnika*, kolorem zielonym dla przykładowego *steganogramu* nie poddanego *zniekształceniom analogowym*, a kolorem czerwonym dla przykładowego *steganogramu* poddanego *zniekształceniom analogowym*.

Tabela 20 - Wyniki: histogramy *maski „-+-” oraz wartości odchylenia standardowego* dla *nośnika* i *steganogramów*
dla różnych *poziomów kodowania* steganogramu 457
Źródło: *Opracowanie własne*

Steganogram: 457 Maska: „-+-”	
Nośnik 457 ***σ=0,2%***	
Oryginalny steganogram 457_1 ***Poziom kodowania=1*** ***σ=0,3%***	
Oryginalny steganogram 457_2 ***Poziom kodowania=2*** ***σ=0,9%***	

Steganogram: 457 Maska: „-+-”	
Oryginalny steganogram 457_3 *Poziom kodowania=3* *σ=4,1%*	
Oryginalny steganogram 457_4 *Poziom kodowania=4* *σ=7,0%*	
Oryginalny steganogram 457_5 *Poziom kodowania=5* *σ=8,7%*	
Końcowy steganogram 457_1 *Poziom kodowania=1* *σ=0,1%*	

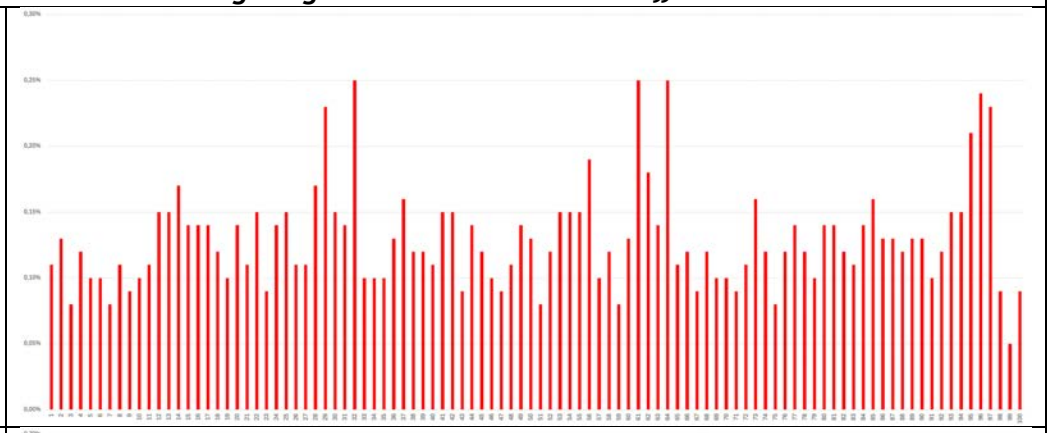
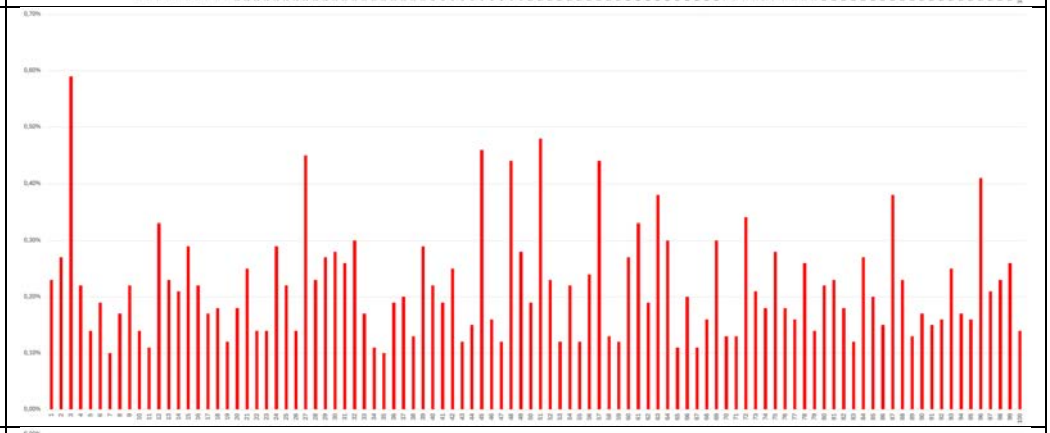
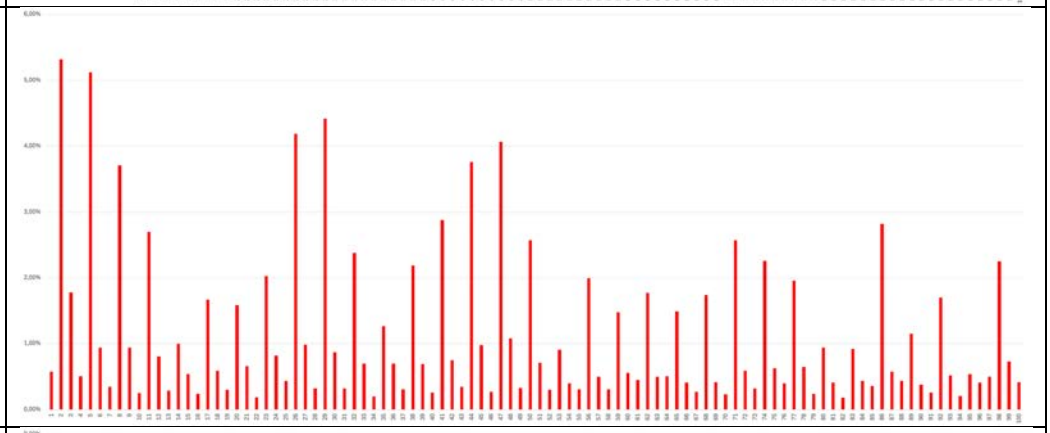
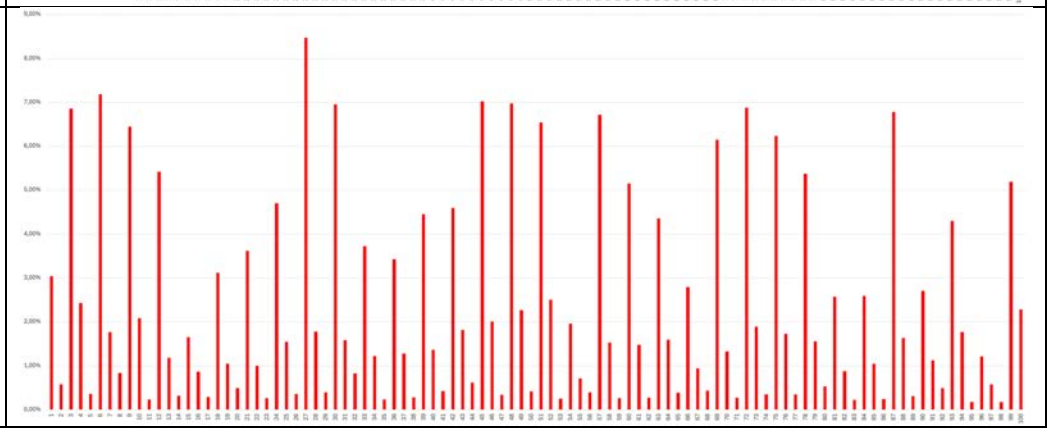
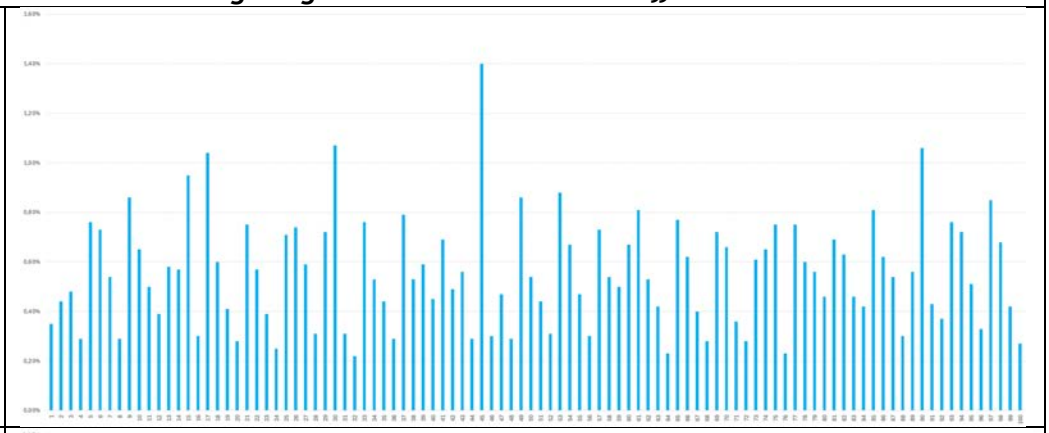
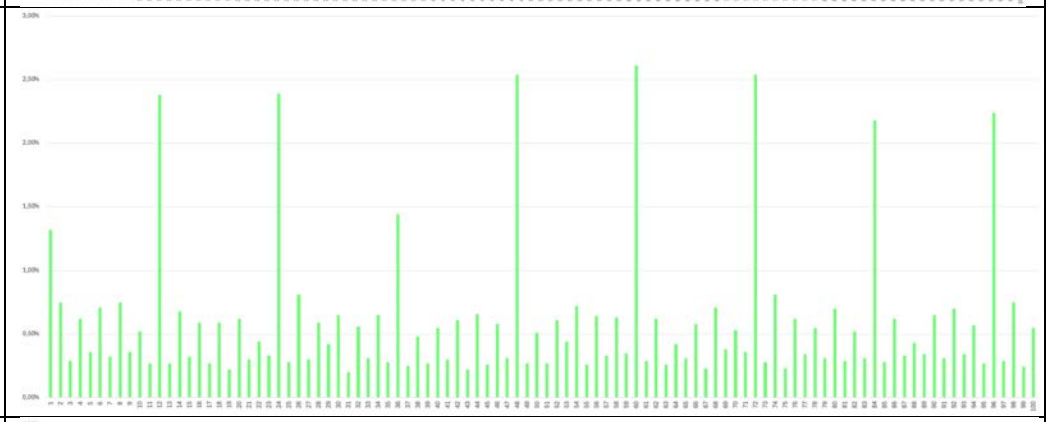
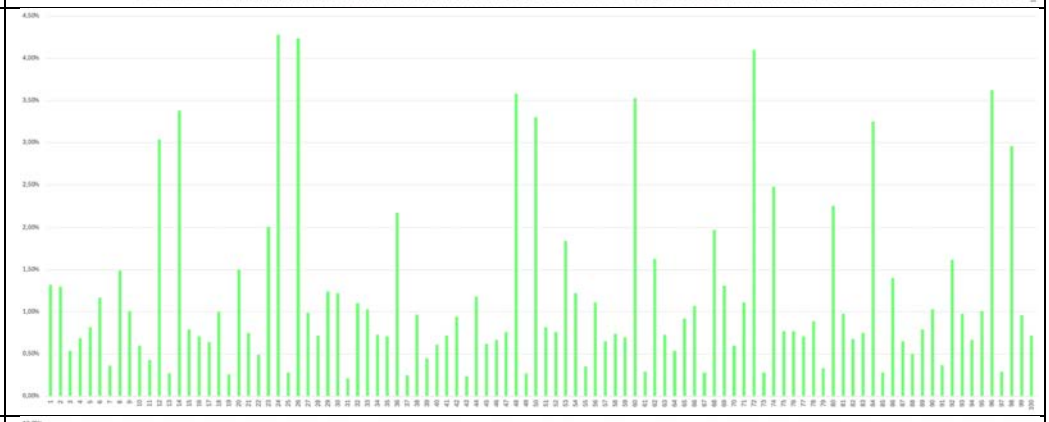
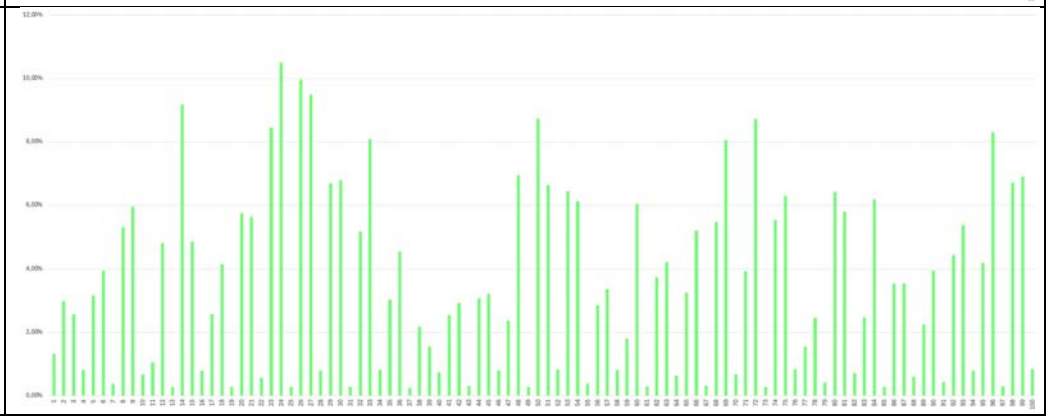
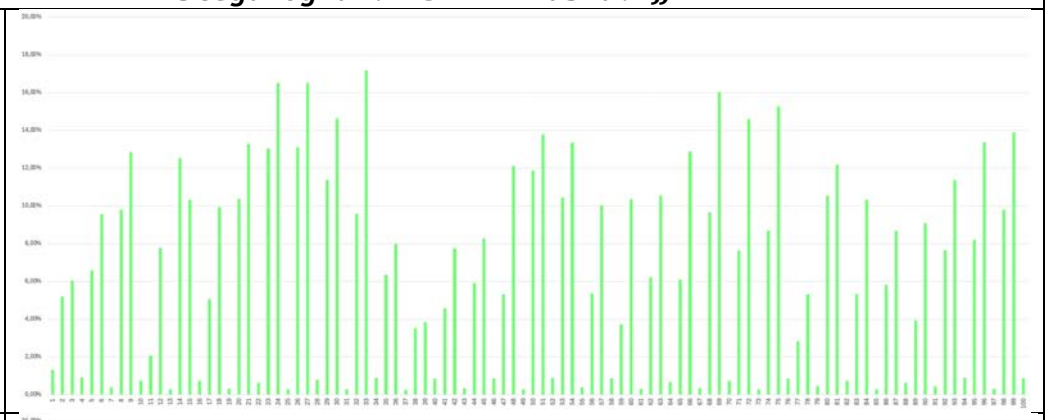
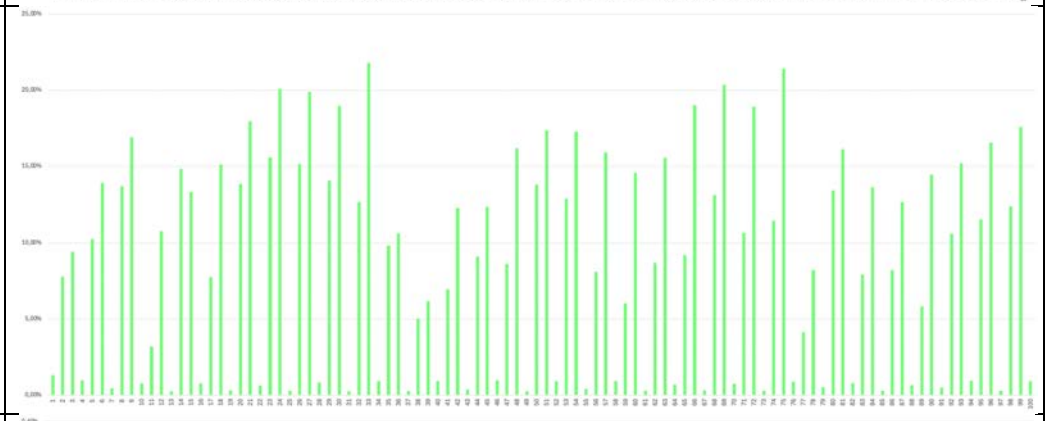
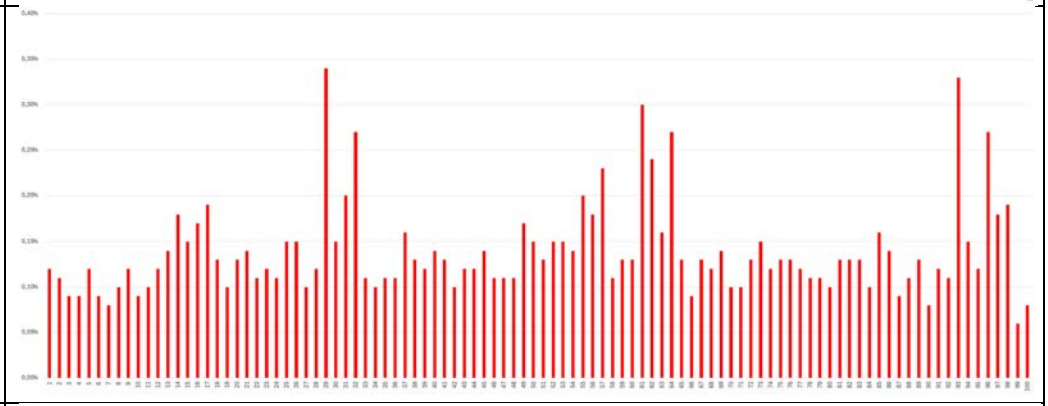
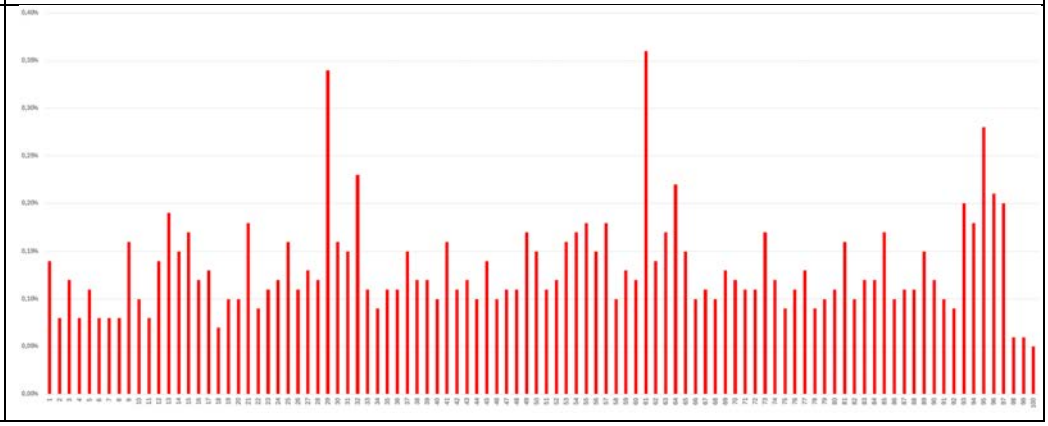
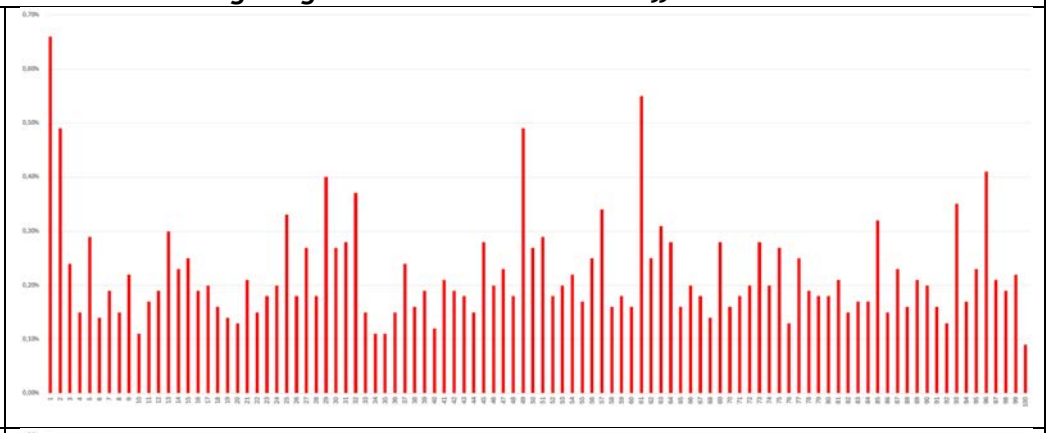
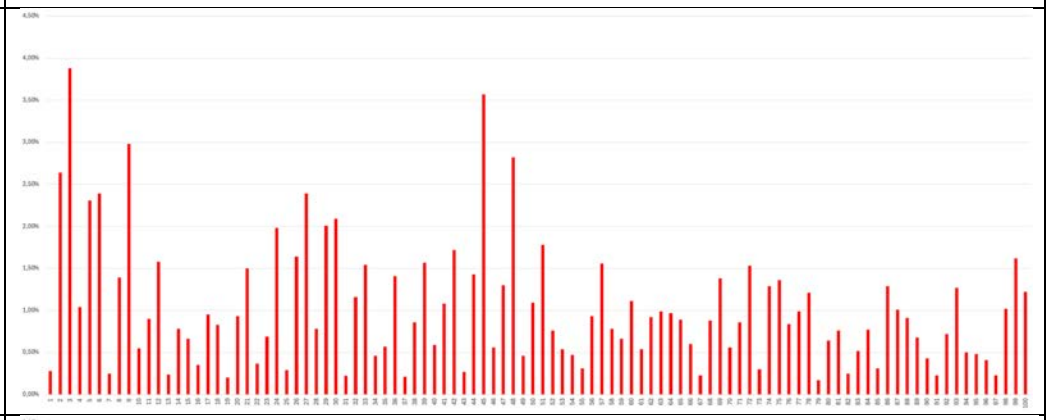
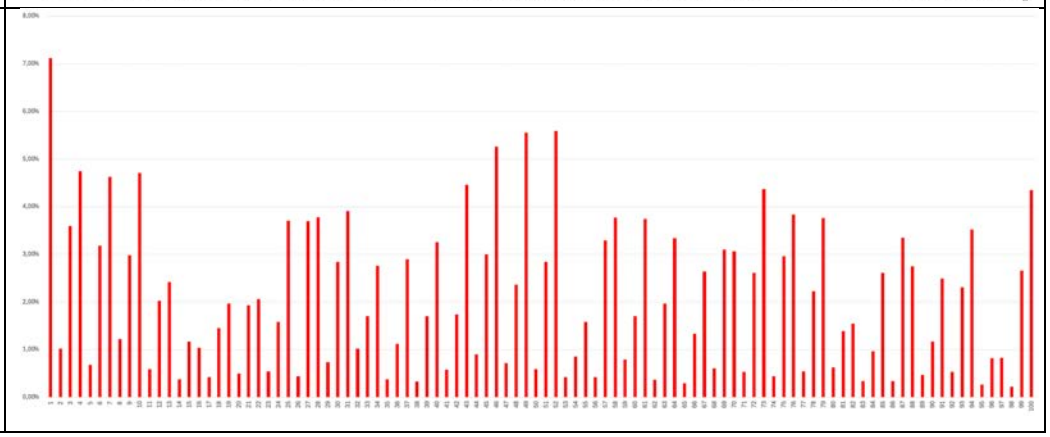
Steganogram: 457 Maska: „-+-”	
Końcowy steganogram 457_2 *Poziom kodowania=2* *σ=0,1%*	
Końcowy steganogram 457_3 *Poziom kodowania=3* *σ=0,1%*	
Końcowy steganogram 457_4 *Poziom kodowania=4* *σ=1,1%*	
Końcowy steganogram 457_5 *Poziom kodowania=5* *σ=2,2%*	

Tabela 21 - Wyniki: histogramy *maski „+-+” oraz wartości odchylenia standardowego* dla *nośnika* i *steganogramów*
dla różnych *poziomów kodowania* steganogramu 457
Źródło: *Opracowanie własne*

Steganogram: 457 Maska: „+-+”	
Nośnik 457 ***σ=0,2%***	
Oryginalny steganogram 457_1 ***Poziom kodowania=1*** ***σ=0,5%***	
Oryginalny steganogram 457_2 ***Poziom kodowania=2*** ***σ=1,0%***	
Oryginalny steganogram 457_3 ***Poziom kodowania=3*** ***σ=2,8%***	

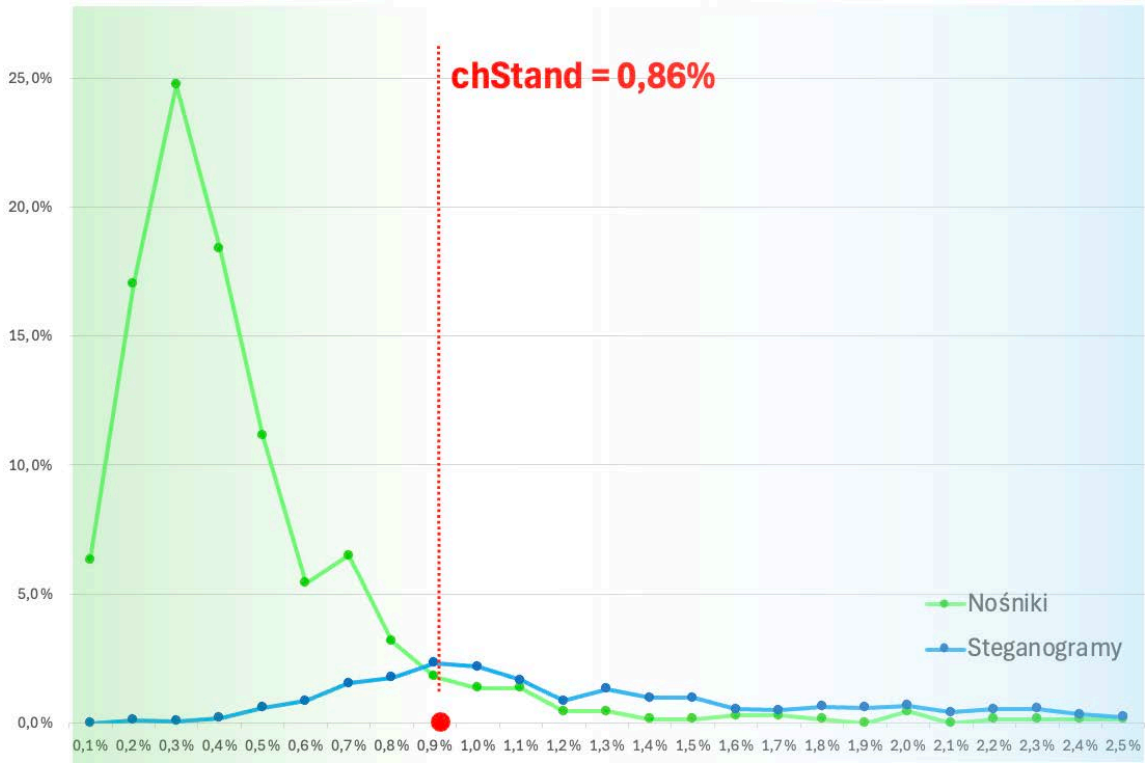
Steganogram: 457 Maska: „+-+”	
Oryginalny steganogram 457_4 *Poziom kodowania=4* *σ=5,2%*	
Oryginalny steganogram 457_5 *Poziom kodowania=5* *σ=6,8%*	
Końcowy steganogram 457_1 *Poziom kodowania=1* *σ=0,1%*	
Końcowy steganogram 457_2 *Poziom kodowania=2* *σ=0,1%*	

Steganogram: 457 Maska: „+-+”	
Końcowy steganogram 457_3 Poziom kodowania=3 σ=0,1%	
Końcowy steganogram 457_4 Poziom kodowania=4 σ=0,7%	
Końcowy steganogram 457_5 Poziom kodowania=5 σ=1,5%	

Na potrzeby proponowanej metody *steganoanalizy RAI* zbadano zależność pomiędzy wartością charakterystyczną *poziomu kodowania chLevel* dla *steganogramów* a *odchyleniem standardowym* rozkładów częstotliwości *masek* kodowania. W tym celu dla każdego *nośnika* z *bazy nośników* oraz każdego *steganogramu* zakodowanego na minimalnym *poziomie kodowania*, który był wystarczający do automatycznego zdekodowania *komunikatu*, wyznaczono *odchylenie standardowe* rozkładu częstotliwości występowania *maski kodowania „+-+”*.

Celem eksperymentu było wyznaczenie takiej wartości *odchylenia standardowego* dla rozkładu częstotliwości *maski* „+-+” w *wideo*, powyżej którego metoda

steganoanalityczna stwierdzałaby, że to *wideo* z dużym prawdopodobieństwem jest *steganogramem* utworzonym przez *podstawowy algorytm kodujący RAI*. Otrzymane wyniki zaprezentowano na rysunku 78.



Rysunek 78 - Wyniki: histogram σ częstotliwości *maski* „+-+" dla *nośników* i *steganogramów* z zaznaczoną wartością charakterystyczną *chStand*
Źródło: *Opracowanie własne*

Na powyższym wykresie na zielono oznaczono histogram *odchylenia standardowego* dla *nośników*, a na niebiesko histogram *odchylenia standardowego* dla *steganogramów*. Na czerwono oznaczono wartość charakterystyczną *odchylenia standardowego* nazwaną *chStand*[161] = 0,86%, dla którego procent *nośników*, których *odchylenie standardowe* było większe niż *chStand* wynosi 6,4% i jest równe procentowi *steganogramów*, których *odchylenie standardowe* było niższe niż *chStand*.

[161] **chStand** - wartość charakterystyczna *odchylenia standardowego* dla rozkładu częstotliwości występowania *maski kodowania* w *steganogramach wideo* zakodowanych przy użyciu *podstawowego algorytmu kodowania RAI*, równa 0,86%.

Na podstawie otrzymanych wyników oraz przeprowadzonej analizy proponuje się następującą metodę *steganoanalityczną RAI*, której celem jest identyfikacja potencjalnych *steganogramów* zakodowanych *podstawowym algorytmem kodowania c-RAI* w *wideo*:

1. Obliczyć rozkład częstotliwości występowania *maski kodującej* „+-+”.
2. Na podstawie wyznaczonego rozkładu częstotliwości obliczyć *odchylenie standardowe*.
3. Jeżeli obliczone *odchylenie standardowe* przekracza wartość progową *chStand* = 0,86%, *wideo* jest klasyfikowane jako *steganogram* utworzony przy użyciu *podstawowego algorytmu kodowania c-RAI*. W przeciwnym przypadku, *wideo* jest uznawane za niezawierające ukrytego *komunikatu*.
4. Prawdopodobieństwa wystąpienia błędu pierwszego rodzaju (fałszywie pozytywny wynik) oraz błędu drugiego rodzaju (fałszywie negatywny wynik) są równe i wynoszą 6,4%.
5. Ewentualne zwiększenie wartości progowej *chStand* prowadzi do zmniejszenia prawdopodobieństwa błędu pierwszego rodzaju kosztem zwiększenia prawdopodobieństwa błędu drugiego rodzaju. Analogicznie, ewentualne zmniejszenie wartości progowej *chStand* skutkuje zmniejszeniem prawdopodobieństwa błędu drugiego rodzaju przy jednoczesnym zwiększeniu prawdopodobieństwa błędu pierwszego rodzaju.

4.5 Ocena wyników przeprowadzonych eksperymentów badawczych

Przeprowadzone eksperymenty badawcze wykazały, że proponowana metoda steganograficzna *RAI* wykazuje się bardzo dużą *odpornością* na różnego rodzaju potencjalne przekształcenia, zniekształcenia i zakłócenia *steganogramów wideo*, w tym na *zniekształcenia analogowe* pomiędzy *nadawcą* a *odbiorcą*. Osiągnięta skuteczność metody steganograficznej *RAI* definiowana jako procent *steganogramów*, dla których przy jakimkolwiek poziomie kodowania z zakresu od 1 do 5 udało się zdekodować zakodowany *komunikat,* wahała się w poszczególnych bazach *steganogramów* od 96,7% do 99,1%, a średnio wyniosła 97,7%.

Jednocześnie przeprowadzone eksperymenty badawcze wykazały wystarczającą *niewykrywalność* metody RAI. Wyznaczone wartości *percLevel* dla badanych

steganogramów oraz obliczona mediana różnic pomiędzy *percLevel* i *chLevel* wynosząca 2 i średnia wynosząca 1,8 pokazują, że pomiędzy *poziomem kodowania* wystarczającym do automatycznego zdekodowania *komunikatu* ze *steganogramu* a *poziomem kodowania*, który jest wykrywalny przez osoby badane występuje istotna odległość wynosząca dwa *poziomy kodowania*. Oznacza to, że proponowana metoda *RAI* przy zastosowaniu odpowiednio dobranych *poziomów kodowania* jest techniką *steganografii wideo niewykrywalną* przez wzrok człowieka.

Przeprowadzone badania wykazały również dużą skuteczność zaproponowanej metody *steganoanalitycznej* wykrywająca zastosowanie *metody RAI*. Bazująca na prostym wskaźniku *chi-kwadrat* metoda *steganoanalizy RAI* wykazuje się dużą dokładnością, z błędem pierwszego i drugiego rodzaju na poziomie 6,4%, a jednocześnie jest łatwo parametryzowalna poprzez zdefiniowany wskaźnik *chStand*.

Poniżej przedstawione są wybrane najciekawsze szczegółowe spostrzeżenia na temat uzyskanych wyników:

1. Zgodnie z oczekiwaniami największą skuteczność dekodowania komunikatu uzyskano dla *bazy oryginalnych steganogramów*, a najmniejsze skuteczności dla *bazy zniekształconych steganogramów*, *bazy transkodowanych steganogramów* oraz *bazy końcowych steganogramów*.
2. 90% *oryginalnych steganogramów* miało wartości charakterystyczne *chIIF* funkcji *IIF* większe niż 0,82. W praktyce oznacza to, że z 90% prawdopodobieństwem można przyjąć, że przy odczytaniu co najmniej 82% *informacji* ze *steganogramu* udaje się poprawnie zdekodować *komunikat*.
3. Mediana uzyskanych wartości *chIIF* dla 1975 *oryginalnych steganogramów* wyniosła 0,89, a średnia arytmetyczna wartości *chIIF* dla tych samych *oryginalnych steganogramów* wyniosła 0,88. Uzyskane wartości, choć wyznaczone na konkretnej *bazie oryginalnych steganogramów*, wydają się być bardzo stabilne i mogą stanowić pewne uniwersalne stałe opisujące właściwości *steganogramów wideo* w metodzie *RAI*.
4. Średnia wartość charakterystyczna *poziomów kodowania chLevel* dla wszystkich *steganogramów* wyniosła 2,5. Jest to średni *poziom kodowania* dla metody *RAI* wystarczający do zdekodowania *komunikatu* uwzględniający różne rodzaje ewentualnych zakłóceń *kanału*.
5. W zależności od wybranego *poziomu kodowania* wartość *chIteration*, czyli minimalna liczba *iteracji dekodujących* niezbędna do zdekodowania *komunikatu* waha się od 58

dla *poziomu kodowania*=1, do 3 dla *poziomu kodowania*=5. Ponieważ *poziom kodowania* jest ujemnie skorelowany z *niewykrywalnością*, a dodatnio skorelowany z *odpornością* oraz *pojemnością*, zgodnie z oczekiwaniami - dla niskich *poziomów kodowania* konieczna jest większa liczba *iteracji dekodujących* do zdekodowania *komunikatu*, dla wyższych *poziomów kodowania* liczba koniecznych *iteracji dekodujących* jest mniejsza.

6. Mediana wartości charakterystycznej *chTime* dla wszystkich *steganogramów* wyniosła 3,5 s., co oznacza, że połowa *steganogramów*, które dawały się w ogóle zdekodować, była zdekodowana w czasie nie dłuższym niż 3,5 s.
7. Biorąc pod uwagę uzyskiwane wartości oczekiwane *chIteration* oraz *chTime* można przyjąć, że dla założonych parametrów *podstawowego algorytmu kodującego c-RAI* oraz *algorytmu dekodującego d-RAI*, jeżeli nie uda się zdekodować *komunikatu* ze *steganogramu* w ciągu 10 sekund, to *komunikat* jest w ogóle niedekodowalny.
8. Uzyskana mediana metryki *chCapacity* na poziomie 20,6 b/s oznacza, iż połowa *steganogramów* ma *pojemność* nie mniejszą niż 20,6 b/s, co obiektywnie nie jest wartością dużą, ale z drugiej strony jest to wartość wystarczająca do przekazania w krótkim czasie ważnych, krótkich *komunikatów*. Jednym ze wskazanych dalej w pracy potencjalnych kierunków badań jest rozwój metody *RAI* w kierunku zwiększenia jej *pojemności*. Można to osiągnąć np. poprzez zwiększenie efektywności kodowania przez wykorzystanie mechanizmów adaptacyjnych oraz kodowanie i dekodowanie wielu *komunikatów* sekwencyjnie jeden po drugim.
9. Dla żadnego badanego *steganogramu* wartość *percLevel* nie była równa ani mniejsza, niż *chLevel*, co pokazuje, że dla każdego steganogramu istnieje taki poziom kodowania, który jest wystarczający do automatycznego zdekodowania komunikatu i jednocześnie jest bezpieczny ze względu na ryzyko wykrycia modyfikacji przez zmysły człowieka.
10. Obliczona mediana różnic pomiędzy *percLevel* i *chLevel* wynosząca 2 oraz średnia wynosząca 1,8 oznaczają, że dla 48% badanych *steganogramów* można zwiększyć ich *pojemności* bez utraty *niewykrywalności* poprzez podwyższenie ich *poziomu kodowania* o wartość 1, a dla 17% nawet o wartość 2.

5 Podsumowanie

5.1 Wnioski

5.1.1 Osiągnięcie celu badań

W niniejszej pracy przedstawiono autorską metodę *steganografii wideo RAI*, która skutecznie odpowiada na wyzwania i problemy związane ze *scenariuszem zniekształceń analogowych*. Jak wykazały przeprowadzone badania, których wyniki zostały przedstawione w rozdziałach tej pracy, metoda *RAI* charakteryzuje się wystarczającą *odpornością* oraz akceptowalnym poziomem *niewykrywalności*.

Za pomocą zaproponowanego *podstawowego algorytmu kodującego c-RAI* można skutecznie zakodować *komunikat* w *steganogramie wideo* tak, aby odtworzony na ekranie był niezauważony dla ludzkiego oka, ale po nagraniu, pomimo dużego poziomu zakłóceń, mógł być prawidłowo odczytany przez *algorytm dekodujący d-RAI*. Jak pokazały badania na próbie osób, minimalny poziom zmian *steganograficznych* w *wideo* wystarczający do zdekodowania *komunikatu,* jest dla ludzi praktycznie niezauważalny. Oznacza to, że zaproponowana metoda *RAI* spełnia główny postulat opisany w *scenariuszu zniekształceń analogowych*, co (poza przedstawieniem samej metody *RAI*) było najważniejszym celem badań zawartych w niniejszej pracy.

Proponowana metoda steganograficzna *RAI* wyróżnia się na tle innych metod *steganografii wideo* swoim unikalnym podejściem do rozwiązania specyficznego i wąskiego problemu *zniekształceń analogowych*. Chociaż skupia się na tym szczególnym zagadnieniu, różne jej elementy, w tym konkretne techniki, proponowane algorytmy oraz zdefiniowane metryki, mogą zostać zastosowane w innych metodach i podejściach do *steganologii*, w tym zwłaszcza w *steganografii wideo*.

Podstawowa koncepcja leżąca u podstaw metody *RAI* - inkrementalne gromadzenie zakodowanej *informacji* w *domenie przestrzennej* poprzez kolejne iteracje przetwarzania *steganogramów wideo*, może być rozwijana niezależnie od przyjętych szczegółowych założeń dla metody *RAI*. Ta idea ma potencjał do dalszej eksploracji i integracji w alternatywnych podejściach, co może przyczynić się do zwiększenia *odporności* i efektywności różnych innych metod *steganograficznych*.

5.1.2 Fundamentalne własności metody RAI

Przeprowadzone badania wykazały kilka podstawowych własności proponowanej metody *RAI*.

Istnieją naturalne częstotliwości występowania różnych *masek* w *nośnikach wideo*, które mogą być nośnikiem *informacji*. Celowa modyfikacja częstotliwości wybranych *masek kodowania* pozwala skutecznie zakodować w *steganogramie wideo komunikat*.

Zakodowany poprzez zmianę częstotliwości *masek komunikat* jest przenoszony pomiędzy kolejnymi *klatkami steganogramu* w sposób stratny. Metoda *RAI* jest skuteczna wtedy i tylko wtedy, gdy ilość *informacji* utracona pomiędzy kolejnymi *klatkami* jest mniejsza niż ilość *informacji* zakodowana w pojedynczej *klatce*. Na ilość *informacji* traconej pomiędzy *klatkami* mają wpływ zakłócenia i przekształcenia *steganogramów wideo* - im większe zniekształcenia, tym większa degradacja *informacji* pomiędzy *klatkami*. Na ilość *informacji* zakodowanej w pojedynczej *klatce* mają wpływ parametry *podstawowego algorytmu kodowania c-RAI*, w tym zwłaszcza *poziom kodowania*. Im większy jest poziom kodowania - tym więcej *informacji* jest zakodowanych w *maskach kodujących* w poszczególnych *klatkach steganogramu*.

W celu zachowania efektywności metody *RAI* należy zadbać o kompromis między ilością zakodowanej *informacji, czyli pojemnością*, a *odpornością* na zakłócenia i *niewykrywalnością* dokonanych zmian.

5.2 Wyszczególnienie oryginalnych osiągnięć

Wydaje się, że oryginalnymi osiągnięciami zaprezentowanymi w niniejszej pracy są realizacje następujących przedsięwzięć badawczych:

1. Przeprowadzenie kompleksowego przeglądu aktualnego stanu wiedzy w obszarze *steganologii* ze szczególnym uwzględnieniem *steganologii wideo*.
2. Zdefiniowanie i opisanie problemu *zniekształceń analogowych* w *steganologii* oraz przedstawienie *scenariusza zniekształceń analogowych* w kontekście różnic względem *podstawowego scenariusza steganologicznego*.
3. Zaproponowanie nowej metody *steganografii wideo RAI* odpowiadającej na problem *zniekształceń analogowych* wraz z definicją formalnego modelu.
4. Zaproponowanie algorytmów kodujących c-*RAI* wykorzystujących do zapisywania *informacji* koncepcję *masek kodujących* oraz algorytmów dekodujących d-*RAI*

wykorzystujących koncepcję adaptacyjnej inkrementacji ilości *informacji* w kolejnych *iteracjach dekodujących*.

5. Zdefiniowanie dla metody *RAI* metryk miar *odporności*, *pojemności* oraz *niewykrywalności*.
6. Przeprowadzenie szczegółowych systematycznych badań wpływu różnego typu przekształceń i zniekształceń *steganogramów wideo* na zdefiniowane metryki.
7. Zastosowanie kodów *QR* do zakodowania *komunikatu* w *steganogramach wideo*, co umożliwiło przeniesienie *komunikatu* z domeny tekstowej na obrazową zapewniając *steganogramowi* dużą *odporność* oraz dało możliwość wykorzystania automatycznych zobiektywizowanych narzędzi do dekodowania *komunikatów* ze *steganogramów* i przeprowadzenie tym samym kompleksowych badań właściwości metody RAI, w tym w rzeczywistym *scenariuszu zniekształceń analogowych*.
8. Przeprowadzenie badań nad *niewykrywalnością* metody *RAI* na grupie osób wraz z ustaleniem rzeczywistych progów detekcji dla ludzkiej percepcji wzrokowej.

5.3 Proponowane kierunki dalszych badań

Interesujące są trzy główne kierunki potencjalnych dalszych badań w zakresie zagadnień omówionych w niniejszej pracy. Pierwszy obszar mógłby obejmować kontynuację badań podstawowych oraz rozwój algorytmów i technik związanych z proponowaną metodą *RAI*, w szczególności rozwój *podstawowego algorytmu kodującego c-RAI*, *algorytmu dekodującego d-RAI* oraz techniki *steganoanalizy RAI*. Drugi kierunek badań mógłby koncentrować się na rozszerzeniu *scenariusza zniekształceń analogowych* na inne obszary *steganografii*, takie jak *steganografia audio* lub *steganografia obrazowa*. Trzeci obszar mógłby dotyczyć badań rozwojowych oraz ewentualnych prac wdrożeniowych związanych z praktycznymi zastosowaniami metody *RAI*, w tym rozwoju narzędzi software'owych i sprzętowych umożliwiających implementację tej metody w rzeczywistych warunkach.

W ramach potencjalnego rozwoju algorytmów i technik związanych z proponowaną metodą *RAI* najbardziej obiecujące wydają się następujące dalsze kierunki badań, które powinny zwiększyć *odporność*, *pojemność* i *niewykrywalność steganogramów RAI*:

1. Rozwój *podstawowego algorytmu kodującego c-RAI* w kierunku adaptacyjnym np. poprzez:

a. usunięcie ograniczeń związanych z założeniem stałego *kroku kodowania* - algorytm mógłby dobierać adaptacyjnie *klatki*, w których będzie kodował *komunikat*, a kryteria wyboru mogłyby uwzględniać konieczną wielkość dokonania zmian,
b. usunięcie ograniczeń związanych z założeniem stałego *poziomu kodowania* - algorytm mógłby dobierać adaptacyjnie *poziom kodowania* dla różnych bloków kodowania lub pikseli minimalizując wielkość zmian w *klatkach steganogramu*.

2. Rozwój *adaptacyjnego algorytmu dekodującego d-RAI* tak, aby móc dekodować *komunikat* bez żadnej wiedzy o parametrach algorytmu kodującego:
   a. usunięcie ograniczeń związanych z założeniem stałego *kroku kodowania* - algorytm dekodujący nie zakładałby z góry, które *klatki* odpowiadają *iteracjom dekodującym*, ale w wyniku analizy częstotliwości występowania *masek kodujących* w kolejnych *klatkach* określałby, które iteracje są *iteracjami dekodującymi*,
   b. usunięcie ograniczeń związanych z założeniem wiedzy o wyborze *masek kodujących* - algorytm dekodujący nie zakładałby z góry konkretnych *masek kodowania*, ale w wyniku analizy częstotliwości występowania *masek* w kolejnych *klatkach*, adaptacyjnie znajdowałby zastosowane przez algorytm kodujący właściwe *maski kodowania*.
3. Zastosowanie kodowania sekwencyjnego kolejnych *komunikatów* jeden po drugim, co wymagałoby dostosowania zarówno algorytmu kodującego, jak i dekodującego.
4. Rozwój *techniki steganoanalizy RAI* w kierunku techniki *steganoanalizy ślepej*, zwłaszcza z uwzględnieniem rozwinięcia algorytmu kodującego w kierunku adaptacyjnym.
5. Uwzględnienie ewentualnego wpływu zastosowania bardziej złożonych modeli *wideo*, np. z uwzględnieniem mechanizmów *kodowania międzyklatkowego*, na przyjęte założenia oraz uzyskiwane wyniki.
6. Rozszerzenie badań nad *niewykrywalnością* poprzez analizę różnic w percepcji zmian w *wideo* spowodowanych kodowaniem *steganograficznym*, uwzględniając cechy demograficzne i inne indywidualne charakterystyki uczestników.

W ramach ewentualnego rozszerzenia *scenariusza zniekształceń analogowych* na inne obszary *steganografii* interesujące wydają się następujące potencjalne pytania badawcze:

1. Czy możliwe jest rozszerzenie metody *RAI* na *steganografię audio*? Czy w *scenariuszu zniekształceń analogowych* można znaleźć taki próg kodowania *steganogramów audio*, który z jednej strony pozwoli automatycznie zdekodować *komunikat*, a z drugiej strony pozwoli pozostać niezauważonym dla ludzkiego ucha?
2. Czy w przypadku *steganografii obrazowej*, gdzie z założenia nie istnieje możliwość implementacji algorytmu wykorzystującego zasadę inkrementalnego zwiększania ilości dekodowanej *informacji*, istnieje jakakolwiek metoda, aby w przypadku *scenariusza zniekształceń analogowych* móc zakodować i zdekodować *komunikat*?

W ramach możliwych badań rozwojowych oraz ewentualnych prac wdrożeniowych związanych z praktycznymi zastosowaniami metody *RAI* inspirującymi wyzwaniami mogłyby być:

1. Zaprojektowanie i skonstruowanie narzędzia software'owego lub sprzętowego do kodowania w czasie rzeczywistym metodą *RAI steganogramu* w postaci strumienia *wideo*. Narzędzie mogłoby działać w taki sposób, że wejściami byłby strumień *wideo* będący *nośnikiem* oraz *komunikat*, a wyjściem byłby strumień *wideo* będący *steganogramem*.
2. Zaprojektowanie i skonstruowanie narzędzia dekodującego w czasie rzeczywistym *komunikat* ze *steganogramu* wyświetlanego w postaci *wideo* na ekranie. Narzędziem mogłaby być aplikacja na smartfon dekodująca na bieżąco obraz widziany przez kamerę.

Bibliografia

1. **Abramowitz, M., & Stegun, I.** (1965). *Handbook of Mathematical Functions: with Formulas, Graphs, and Mathematical Tables.* NYC, USA: Dover Publications, ISBN 978-0486612720.
2. **Ackoff, R.** (1989). *From Data to Wisdom.* Cambridge, UK: Journal of Applied Systems Analysis, 16, 3-9, ISSN 0308-9541.
3. **Ahmed, N., Natarajan, T., & Rao, K.** (1974). *Discrete cosine transform.* Piscataway, USA: IEEE Transactions on Computers, C-23(1), 90-93, ISSN 0018-9340, DOI 10.1109/T-C.1974.223784.
4. **Ahn, L., & Hopper, N.** (2004). *Public-Key Steganography.* Berlin, Germany: Springer, Advances in Cryptology - LNCS, 3027, 323–341, ISBN 978-3-540-21935-4, DOI 10.1007/978-3-540-24676-3_20.
5. **Akansu, A., & Haddad, R.** (2000). *Multiresolution Signal Decomposition Transforms, Subbands, and Wavelets.* Amsterdam, Netherlands: Academic Press, ISBN 9780120471416.
6. **Anderson, J., & Johannesson, R.** (2006). *Understanding Information Transmission.* NYC, USA: Wiley-IEEE Press, ISBN 978-0-471-71120-9.
7. **Anderson, R.** (1996). *Information Hiding.* Cambridge, UK: First International Workshop, Springer-Verlag, ISBN 978-3-540-49589-5, DOI 10.1007/3-540-61996-8.
8. **Anderson, R., & Petitcolas, F.** (1998). *On the limits of steganography.* NYC, USA: IEEE Journal on Selected Areas in Communications, 16(4), 474-481, ISSN 0733-8716, DOI 10.1109/49.668971.
9. **Antoniou, A.** (1993). *Digital Filters: Analysis, Design and Applications.* NYC, USA: McGraw-Hill College, ISBN 978-0070021211.
10. **Artz, D.** (2001). *Digital steganography: Hiding data within data.* NYC, USA: IEEE Internet Computing, 5(3), 75-80, ISSN 1941-0131, DOI 10.1109/4236.935180.
11. **Beach, A., & Owen, A.** (2018). *Video Compression Handbook.* New Jersey, USA: Peachpit Press, ISBN 978-0134866215.
12. **Beckett, B.** (1988). *Introduction to cryptology.* Boston, USA: Blackwell Scientific Publications, ISBN 9780632018369.
13. **Bender, W., Gruhl, D., Morimoto, N., & Lu, A.** (1996). *Techniques for data hiding.* NYC, USA: IBM Systems Journal, 35(3-4), 313-336, ISSN 0018-8670, DOI 10.1147/sj.353.0313.
14. **Bennett, K.** (2004). *Linguistic Steganography: Survey, Analysis, and Robustness Concerns for Hiding Information in Text.* West Lafayette, USA: CERIAS Tech Report 2004-13, Purdue University, [Online] https://www.cerias.purdue.edu/assets/pdf/bibtex_archive/2004-13.pdf, Retrieved 24.06.2024.
15. **Bennett, W.** (1948). *Spectra of quantized signals.* NYC, USA: The Bell System Technical Journal, 27, 3, 446-472, DOI 10.1002/j.1538-7305.1948.tb01340.x.

16. **Blackledge, J.** (2011). *Cryptography and Steganography-New Algorithms and Applications.* Warsaw, Poland: Warsaw University of Technology, ISBN 978-83-61993-05-6, DOI 10.21427/D7ZS63.

17. **Bracewell, R.** (1986). *The Fourier Transform and Its Applications.* NYC, USA: McGraw-Hill, ISBN 978-0073039381.

18. **Böhme, R.** (2010). *Advanced Statistical Steganalysis.* NYC, USA: Springer, ISBN 978-3-642-14312-0, DOI 10.1007/978-3-642-14313-7.

19. **Burnham, K., & Anderson, D.** (2004). *Model Selection and Multimodel Inference.* NYC, USA: Springer, ISBN 978-0-387-95364-9, DOI 10.1007/978-0-387-22456-5_5.

20. **Cachin, C.** (1998). *An Information-Theoretic Model for Steganography.* Berlin, Germany: Springer, Information Hiding - Lecture Notes in Computer Science, 1525, 306–318, DOI 10.1007/3-540-49380-8_21.

21. **Cao, M., Tian, L., & Li, C.** (2020). *A Secure Video Steganography Based on the Intra-Prediction Mode.* Basel, Switzerland: Sensors, 20(18), ISSN 1424-8220, DOI 10.3390/s20185242.

22. **Castiglione, A., De Santis, A., & Soriente, C.** (2007). *Taking advantages of a disadvantage: Digital forensics and steganography using document metadata.* Amsterdam, Netherlands: The Journal of systems and software, 80 (5), 750-764, DOI 10.1016/j.jss.2006.07.006.

23. **Chai, H., Li, Z., Li, F., & Zhang, Z.** (2022). *End-to-End Video Steganography Network Based on a Coding Unit Mask.* Basel, Switzerland: Electronics, 11(7), 1142, ISSN 2079-9292, DOI 10.3390/electronics11071142.

24. **Chan, C., & Cheng, L.** (2004). *Hiding data in images by simple LSB substitution.* Amsterdam, Netherlands: Elsevier, Pattern Recognition, 37(3), 469-474, ISSN 0031-3203, DOI 10.1016/j.patcog.2003.08.007.

25. **Chandra, S., Paira, S., Alam, S., & Sanyal, G.** (2014). *A comparative survey of Symmetric and Asymmetric Key Cryptography.* Hosur, India: ICECCE, 83-93, DOI 10.1109/ICECCE.2014.7086640.

26. **Chang, C., Chen, T., & Chung, L.** (2002). *A steganographic method based upon JPEG and quantization table modification.* Amsterdam, Netherlands: Information Sciences, 141(1-2), 123-138, ISSN 0020-0255, DOI 10.1016/S0020-0255(01)00194-3.

27. **Cheddad, A., Condell, J., Curran, K., & Kevitt, P.** (2010). *Digital image steganography: Survey and analysis of current methods.* Amsterdam, Netherlands: Signal Processing, 90(3), 727-752, ISSN 0165-1684, DOI 10.1016/j.sigpro.2009.08.010.

28. **Chen, B., & Wornell, G.** (2001). *Quantization Index Modulation: A Class of Provably Good Methods for Digital Watermarking and Information Embedding.* NYC, USA: IEEE Transactions on Information Theory, 47(4), 1423-1443, ISSN 0018-9448, DOI 10.1109/18.923725.

29. **Cole, E.** (2003). *Hiding in Plain Sight: Steganography and the Art of Covert Communication.* Hoboken, USA: Wiley, ISBN 978-0471444497.

30. **Cover, T., & Thomas, J.** (1991). *Elements of Information Theory.* Hoboken, USA: John Wiley & Sons, ISBN 9780471241959.

31. **Cox, I., Miller, M., Bloom, J., Fridrich, J., & Kalker, T.** (2007). *Digital Watermarking and Steganography.* Burlington, USA: Morgan Kaufmann Publishers, ISBN 978-0123725851.

32. **Daubechies, I.** (1992). *Ten Lectures on Wavelets.* Philadelphia, USA: Society for Industrial and Applied Mathematics (SIAM), ISBN 978-0898712742.

33. **Dempster, A., Laird, N., & Rubin, D.** (1977). *Maximum Likelihood from Incomplete Data Via the EM Algorithm.* Oxford, UK: Journal of the Royal Statistical Society, 39, 1, 1–38, DOI 10.1111/j.2517-6161.1977.tb01600.x.

34. **DensoWave.** (2024). *QR code.* Denso Wave: [Online] https://www.qrcode.com, Retrieved 24.06.2024.

35. **Duda, J., Tahboub, K., Gadgil, N., & Delp, E.** (2015). *The use of asymmetric numeral systems as an accurate replacement for Huffman coding.* Cairns, Australia: Picture Coding Symposium (PCS), 65-69, DOI 10.1109/PCS.2015.7170048.

36. **Dynamsoft.** (2024). *Barcode Scanner.* Dynamsoft: [Online] https://www.dynamsoft.com/barcode-reader/sdk-desktop-server/, Retrieved 24.06.2024.

37. **Eid, W., Alotaibi, S., Alqahtani, H., & Saleh, S.** (2022). *Digital Image Steganalysis: Current Methodologies and Future Challenges.* Piscataway, USA: IEEE Access, 10, 78754-78775, ISSN 2169-3536, DOI 10.1109/ACCESS.2022.3202905.

38. **FFmpeg.** (2024). *FFmpeg.* FFmpeg.org: [Online] https://ffmpeg.org, Retrieved 24.06.2024.

39. **Floridi, L.** (2010). *Information: A Very Short Introduction.* Oxford, UK: Oxford University Press, ISBN 978-0199551378.

40. **Freedman, L.** (2013). *Strategy: A History.* Oxford, UK: Oxford University Press, ISBN 978-0199325153.

41. **Fridrich, J.** (2009). *Steganography in Digital Media: Principles, Algorithms, and Applications.* Cambridge, UK: Cambridge University Press, ISBN 9781139192903, DOI 10.1017/CBO9781139192903.014.

42. **Fridrich, J., & Kodovsky, J.** (2012). *Rich models for steganalysis of digital images.* Piscataway, USA: IEEE Transactions on Information Forensics and Security 7(3), 868-882, ISSN 1556-6013, DOI 10.1109/TIFS.2012.2190402.

43. **Fridrich, J., Du, R., & Meng, L.** (2000). *Steganalysis of LSB encoding in color images.* NYC, USA: IEE Multimedia and Expo 3, 1279-1282, ISBN 0-7803-6536-4, DOI 10.1109/ICME.2000.871000.

44. **Fridrich, J., Goljan, M., & Kodovský, J.** (2002). *Practical Steganalysis of Digital Images - State of the Art.* Bellingham, USA: Proceedings of SPIE - The International Society for Optical Engineering Data, 4675, ISSN 0277-786X, DOI 10.1117/12.465263.

45. **Fridrich, J., Goljan, M., & Rui, D.** (2001). *Detecting LSB steganography in color, and gray-scale images.* NYC, USA: IEEE Multimedia, 8, 4, 22–28, ISSN 1070-986X, DOI 10.1109/93.959097.

46. **Gao, W., & Ma, S.** (2015). *Advanced Video Coding Systems.* Berlin, Germany: Springer, ISBN 978-3-319-14242-5.

47. **Ghamsarian, N., Schoeffmann, K., & Khademi, M.** (2021). *Blind MV-based video steganalysis based on joint inter-frame and intra-frame statistics.* Berlin, Germany: Springer, Multimed Tools Appl 80, 9137–9159, DOI 10.1007/s11042-020-10001-9.

48. **Goodfellow, I., Bengio, Y., & Courville, A.** (2016). *Deep Learning.* Cambridge, USA: MIT Press, ISBN 9780262035613.

49. **Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., . . . Bengio, Y.** (2014). *Generative Adversarial Networks.* Montreal, Canada: Advances in Neural Information Processing Systems (NIPS) 27(11), 2672-2680, ISSN 1049-5258, DOI 10.1145/3422622.

50. **Gordon, N.** (1998). *Colour blindness.* Amsterdam, Netherlands: Elsevier, Public Health, 112, 2, 81-84, DOI 10.1038/sj.ph.1900446.

51. **Hanzo, L., Cherriman, P., & Streit, J.** (2007). *Video Compression and Communications: From Basics to H.261, H.263, H.264, MPEG4 for DVB and HSDPA-Style Adaptive Turbo-Transceivers.* Hoboken, USA: Wiley-IEEE Press, ISBN 978-0-470-51991-2.

52. **Harmsen, J., & Pearlman, W.** (2003). *Steganalysis of Additive Noise Modelable Information Hiding.* Bellingham, USA: Proceedings of SPIE - The International Society for Optical Engineering, 5020, ISSN 0277-786X, DOI 10.1117/12.476813.

53. **Hirsch, R.** (2004). *Exploring Colour Photography: A Complete Guide.* London, UK: Laurence King Publishing, ISBN 978-1856694209.

54. **Horé, A., & Ziou, D.** (2010). *Image quality metrics: PSNR vs. SSIM.* Istanbul, Turkey: ICPR, 2366-2369, DOI 10.1109/ICPR.2010.579.

55. **Howitt, D., & Cramer, D.** (2007). *Introduction to Statistics in Psychology.* London, UK: Pearson Education Limited, ISBN 978-0132051613.

56. **Hu, D., Wang, L., Jiang, W., Zheng, S., & Li, B.** (2018). *A Novel Image Steganography Method via Deep Convolutional Generative Adversarial Networks.* NYC, USA: IEEE Access, 6, 38303-38314, ISSN 2169-3536, DOI 10.1109/ACCESS.2018.2852771.

57. **Huynh-Thu, Q., & Ghanbari, M.** (2008). *Scope of validity of PSNR in image/video quality assessment.* London, UK: Electronics Letters, 44(13), 800-801, ISSN 0013-5194, DOI 10.1049/el:20080522.

58. **Ishihara, S.** (1917). *Tests for color-blindness.* Tokyo: Handaya Hongo Harukich.

59. **ISO-IEC.** (2004). *PNG.* Geneva, Switzerland: ISO-IEC 15948:2004.

60. **ITU-T.** (1992). *JPEG (T.81).* Geneva, Switzerland: International Telecommunication Union.

61. **ITU-T.** (1996). *H.263.* Geneva, Switzerland: International Telecommunication Union.

62. **ITU-T.** (2006). *MJPEG (T.802).* Geneva, Switzerland: International Telecommunication Union.

63. **ITU-T.** (2011). *H.264.* Geneva, Switzerland: International Telecommunication Union.

64. **ITU-T.** (2019). *H.265.* Geneva, Switzerland: International Telecommunication Union.

65. **ITU-T.** (2024). *H.266.* Geneva, Switzerland: International Telecommunication Union.

66. **Jain, A.** (1989). *Fundamentals of Digital Image Processing.* Englewood Cliffs, USA: Prentice Hall, ISBN 978-0133361659.

67. **Janeczko, P.** (2004). *Top Secret: A Handbook of Codes, Ciphers, and Secret Writing.* Cambridge, USA: Candlewick Press, ISBN 978-0763629723.

68. **Johnson, N., & Jajodia, S.** (1998). *Exploring Steganography: Seeing the Unseen.* Los Alamitos, USA: IEEE Computer, 31(2), 26-34, ISSN 0018-9162, DOI 10.1109/MC.1998.4655281.

69. **Johnson, N., Duric, Z., & Jajodia, S.** (2001). *Information Hiding: Steganography and Watermarking - Attacks and Countermeasures.* Boston, USA: Springer, ISBN 978-0-7923-7204-2, DOI 10.1007/978-1-4615-4375-6.

70. **Kadhim, I., Premaratne, P., Vial, P., & Halloran, B.** (2019). *Comprehensive survey of image steganography: Techniques, Evaluations, and trends in future research.* Amsterdam, Netherlands: Neurocomputing, 335, 299-326, ISSN 0925-2312, DOI 10.1016/j.neucom.2018.06.075.

71. **Kahn, D.** (1967). *The Codebreakers: The Story of Secret Writing.* NYC, USA: Macmillan, ISBN 978-0025604605.

72. **Katzenbeisser, S., & Petitcolas, F.** (2000). *Information Hiding Techniques for Steganography and Digital Watermarking.* Norwood, USA: Artech House, ISBN 978-1-58053-035-4.

73. **Kenney, J.** (1954). *Mathematics of Statistics.* Princeton, USA: Van Nostrand company, 77-80, ISBN 978-0442043520.

74. **Ker, A.** (2005). *Steganalysis of LSB matching in grayscale images.* Piscataway, USA: IEEE Signal Processing Letters, 12(6), 441-444, ISSN 1070-9908, DOI 10.1109/LSP.2005.847889.

75. **Kessler, G.** (2015). *An Overview of Steganography for the Computer Forensics Examiner.* Daytona Beach, USA: Forensic Science Communications, [Online] https://www.garykessler.net/library/fsc_stego.html, Retrieved 03.07.2024.

76. **Khan, A., Sohail, A., Zahoora, U., & Qureshi, A.** (2020). *A Survey of the Recent Architectures of Deep Convolutional Neural Networks.* Berlin, Germany: Artificial Intelligence Review, 53(1-87), 5455-5514, ISSN 0269-2821, DOI 10.1007/s10462-020-09825-6.

77. **Kharrazi, M., Sencar, H., & Memon, N.** (2006). *Performance study of common image steganography and steganalysis techniques.* Bellingham, USA: Journal of Electronic Imaging, 15(4), 041104, ISSN 1017-9909, DOI 10.1117/1.2400672.

78. **Kheddar, H., Hemis, M., Himeur, Y., Megías, D., & Amira, A.** (2024). *Deep learning for steganalysis of diverse data types: A review of methods, taxonomy, challenges and future directions.* NYC, USA: Neurocomputing, 581, 127528, ISSN 0925-2312, DOI 10.1016/j.neucom.2024.127528.

79. **Krizhevsky, A., Sutskever, I., & Hinton, G.** (2017). *ImageNet Classification with Deep Convolutional Neural Networks.* Lake Tahoe, USA: Advances in Neural Information Processing Systems (NIPS), 25, 1097-1105, ISSN 1049-5258, DOI 10.1145/3065386.

80. **Kunhoth, J., Subramanian, N., Al-Maadeed, S., & Bouridane, A.** (2023). *Video steganography: recent advances and challenges.* Berlin, Germany: Springer -

Multimedia Tools and Applications, 82(8), 11817-11868, ISSN 1380-7501, DOI 10.1007/s11042-023-14844-w.

81. **LeCun, Y., Bengio, Y., & Hinton, G.** (2015). *Deep learning.* London, UK: Nature 521, 436–444, DOI 10.1038/nature14539.

82. **Li, B., He, J., Huang, J., & Shi, Y.** (2011). *A survey on image steganography and steganalysis.* Taipei, Taiwan: Journal of Information Hiding and Multimedia Signal Processing, 2(2), ISSN 2073-4212.

83. **Lin, C., & Tsai, W.** (2004). *Secret image sharing with steganography and authentication.* Amsterdam, Netherlands: The Journal of Systems and Software, 73(3), 405-414, ISSN 0164-1212, DOI 10.1016/S0164-1212(03)00239-5.

84. **Luo, W., Huang, F., & Huang, J.** (2010). *Edge Adaptive Image Steganography Based on LSB Matching Revisited.* Piscataway, USA: IEEE Transactions on Information Forensics and Security, 5(2), 201-214, ISSN 1556-6021, DOI 10.1109/TIFS.2010.2041812.

85. **Mazurczyk, W., Wendzel, S., Zander, S., Houmansadr, S., & Szczypiorski, K.** (2016). *Information Hiding in Communication Networks: Fundamentals, Mechanisms, Applications, and Countermeasures.* Hoboken, USA: Wiley-IEEE Press, ISBN 978-1-118-86169-1.

86. **Meyn, S., & Tweedie, R.** (2009). *Markov Chains and Stochastic Stability.* Cambridge, UK: Cambridge University Press, ISBN 978-0-521-73182-9.

87. **Morkel, T., & Eloff, J. O.** (2005). *An overview of image steganography.* Pretoria, South Africa: ISSA 2005 New Knowledge Today Conference, ISBN 1-86854-625-X.

88. **Mou, C., Xu, Y., Song, J., Zhao, C., Ghanem, B., & Zhang, J.** (2023). *Large-Capacity and Flexible Video Steganography via Invertible Neural Network.* Vancouver, Canada: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 22606-22615, DOI 10.1109/CVPR52729.2023.02165.

89. **Muchahary, D., Mondal, A., Parmar, R., Borah, A., & Majumder, A.** (2015). *A Simplified Design Approach for Efficient Computation of DCT.* Gwalior, India: Fifth International Conference on Communication Systems and Network Technologies, 483-487, DOI 10.1109/CSNT.2015.134.

90. **Nelson, M., & Gailly, J.** (1995). *The Data Compression Book.* Emeryville, USA: M&T Books, ISBN 978-1558514341.

91. **Nieles, M., Dempsey, K., & Pillitteri, V.** (2017). *An Introduction to Information Security.* Gaithersburg, USA: NIST National Institute of Standards and Technology, [Online] https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-12r1.pdf, Retrieved 24.06.2024.

92. **Nyquist, H.** (1928). *Certain Topics in Telegraph Transmission Theory.* NYC, USA: Transactions of the American Institute of Electrical Engineers, 47(2), 617-644, ISSN 0096-3860, DOI 10.1109/T-AIEE.1928.5055024.

93. **Opencv.org.** (2024). *OpenCV.* OpenCV: [Online] https://opencv.org, Retrieved 24.06.2024.

94. **Pan, F., Xiang, L., Yang, X., & Guo, Y.** (2010). *Video steganography using motion vector and linear block codes.* Beijing, China: IEEE International Conference on

Software Engineering and Service Sciences, 592-595, DOI 10.1109/ICSESS.2010.5552283.

95. **Patel, J., & Read, C.** (1996). *Handbook of the Normal Distribution.* Oxford, UK: Taylor & Francis Inc, ISBN 978-0824793425.
96. **Pery, M.** (2024). *Zastosowanie steganologii w cyberbezpieczeństwie.* Warsaw, Poland: Military Uniwersity of Technology, Faculty of Cybernetics.
97. **Pery, M., & Waszkowski, R.** (2024). *A Model and Quantitative Framework for Evaluating Iterative Steganography.* Basel, Switzerland: Entropy 2024, 26, 1130, DOI 10.3390/e26121130.
98. **Pery, M., & Waszkowski, R.** (2024). *Analyzing natural pixel color variations in consecutive video frames to enhance spatial domain video steganography.* Grenada, Spain: 44th IBIMA Computer Science Conference, 27-28.11.2024, 250-259, ISBN 979-8-9867719-5-3, ISSN 2767-9640.
99. **Pery, M., & Waszkowski, R.** (2024). *Computational System for Evaluating Human Perception in Video Steganography.* Lublin, Poland: Applied Computer Science, 20, 4, 63–76, DOI 10.35784/acs-2024-40.
100. **Pery, M., & Waszkowski, R.** (2024). *Mathematical modeling in color difference analysis in consecutive video frames for steganography.* Grenada, Spain: 44th IBIMA Computer Science Conference, 27-28.11.2024, 245-249, ISBN 979-8-9867719-5-3, ISSN 2767-9640.
101. **Petitcolas, F., Anderson, R., & Kuhn, M.** (1999). *Information hiding - a survey.* NYC, USA: Proceedings of the IEEE, 87(7), 1062-1078, ISSN 0018-9219, DOI 10.1109/5.771065.
102. **Pevný, T., & Fridrich, J.** (2007). *Merging Markov and DCT features for multi-class JPEG steganalysis.* Bellingham, USA: Proceedings of SPIE - The International Society for Optical Engineering, 6505, 650503, ISSN 0277-786X, DOI 10.1117/12.696774.
103. **Pevný, T., Bas, P., & Fridrich, J.** (2010). *Steganalysis by Subtractive Pixel Adjacency Matrix.* Piscataway, USA: IEEE Transactions on Information Forensics and Security, 5(2), 215-224, ISSN 1556-6021, DOI 10.1109/TIFS.2010.2045842.
104. **Pinsky, M.** (2009). *Introduction to Fourier Analysis and Wavelets.* Providence, USA: American Mathematical Society, ISBN 978-0534376604.
105. **Pixabay.com.** (2024). *Pixabay License.* Pixabay.com: [Online] https://pixabay.com/service/license-summary/, Retrieved 24.06.2024.
106. **Polyanin, A., & Manzhirov, A.** (2008). *Handbook of Integral Equations.* Boca Raton, USA: CRC Press, ISBN 9781584885078.
107. **Pramanik, S., Ghonge, M., Ravi, R., & Cengiz, K.** (2021). *Multidisciplinary Approach to Modern Digital Steganography.* Hershey, USA: IGI Global (Information Science Reference), ISBN 9781799871620, DOI 10.4018/978-1-7998-7160-6.
108. **Provos, N., & Honeyman, P.** (2003). *Hide and seek: An introduction to steganography.* Los Alamitos, USA: IEEE Security & Privacy, 1(3), 32-44, ISSN 1540-7993, DOI 10.1109/MSECP.2003.1203220.

109. **Qian, Y., Dong, J., Wang, W., & Tan, T.** (2018). *Feature learning for steganalysis using convolutional neural networks.* Berlin, Germany: Multimedia Tools and Applications 77(2), 15, 19633 - 19657, DOI 10.1007/s11042-017-5326-1.

110. **Rahmani, K., Arora, K., & Pal, N.** (2014). *A Crypto-Steganography: A Survey.* NYC, USA: International Journal of Advanced Computer Science and Applications (IJACSA), 5(7), ISSN 2156-5570, DOI 10.14569/IJACSA.2014.050722.

111. **Richardson, I.** (2003). *H.264 and MPEG-4 Video Compression: Video Coding for Next-generation Multimedia.* NYC, USA: John Wiley & Sons, ISBN 978-0-470-84837-5.

112. **Rivaz, P., & Haughton, J.** (2018). *AV1 Bitstream & Decoding Process Specification.* Wilmington, USA: The Alliance for Open Media, https://aomediacodec.github.io/av1-spec/, Retrieved 12.12.2024.

113. **Rowley, J.** (2007). *The wisdom hierarchy: representations of the DIKW hierarchy.* Thousand Oaks, USA: Journal of Information Science, 33(2), 163-180, DOI 10.1177/0165551506070706.

114. **Ryan, W., & Lin, S.** (2009). *Channel Codes: Classical and Modern.* Cambridge, UK: Cambridge University Press, ISBN 978-0-521-84868-8.

115. **Sadek, M., Khalifa, A., & Mostafa, M.** (2014). *Video steganography: a comprehensive review.* Amsterdam, Netherlands: Multimedia Tools and Applications, 74(17), 7063–7094, ISSN 1573-7721, DOI 10.1007/s11042-014-1952-z.

116. **Sencar, H., Ramkumar, M., & Akansu, A.** (2004). *Data Hiding Fundamentals and Applications.* Burlington, USA: Elsevier Academic Press, ISBN 9780120471447, DOI 10.1016/B978-0-12-047144-7.X5000-5.

117. **Shannon, C.** (1948). *A Mathematical Theory of Communication.* NYC, USA: Bell System Technical Journal, 27(3), 379-423, ISSN 0005-8580, DOI 10.1002/j.1538-7305.1948.tb01338.x.

118. **Shi, Y., Chen, C., & Chen, W.** (2006). *A Markov Process Based Approach to Effective Attacking JPEG Steganography.* Alexandria, USA: Information Hiding, 8th International Workshop, Lecture Notes in Computer Science, 4437, 249-264, ISBN 978-3540741237, DOI 10.1007/978-3-540-74124-4_17.

119. **Shih, F.** (2017). *Digital Watermarking and Steganography.* Boca Raton, USA: CRC Press, ISBN 9781315121109, DOI 10.1201/9781315121109.

120. **Stoica, P., & Moses, R.** (2005). *Spectral Analysis of Signals.* New Jersey, USA: Prentice Hall, Inc., ISBN 9780131139565.

121. **Subhedar, M., & Mankar, V.** (2014). *Current Status and Key Issues in Image Steganography: A Survey.* Amsterdam, Netherlands: Computer Science Review, 13-14, 95-113, ISSN 1574-0137, DOI 10.1016/j.cosrev.2014.09.001.

122. **Sushma, R., & Manjula, G.** (2024). *StegVRN: Enhancing Quality of Video Steganography Using CNN-Based Techniques.* NYC, USA: SN Computer Science, 4(6), 638, ISSN 2661-8907, DOI 10.1007/s42979-023-02498-2.

123. **Tekalp, M.** (1995). *Digital Video Processing.* Upper Saddle River, USA: Prentice Hall, ISBN 978-0-13-190075-7.

124. **VP9.** (2024). *VP9.* VP9: [Online] https://www.webmproject.org/vp9/, Retrieved 24.06.2024.

125. **Wang, Y.** (2024). *Hiding Data Within Thumbnail Videos: An Adaptive Downsampling-Resilient Video Steganography Method.* NYC, USA: IEEE Access, 12, 17446-17458, ISSN 2169-3536, DOI 10.1109/ACCESS.2024.3386798.

126. **Wang, Z., Bovik, A., Sheikh, H., & Simoncelli, E.** (2004). *Image Quality Assessment: From Error Visibility to Structural Similarity.* NYC, USA: IEEE Transactions on Image Processing, 13(4), 600-612, ISSN 1057-7149, DOI 10.1109/TIP.2003.819861.

127. **Watson, A.** (1993). *DCT quantization matrices visually optimized for individual images.* Bellingham, USA: Proceedings of SPIE - The International Society for Optical Engineering, 1913–1914, ISSN 0277-786X, DOI 10.1117/12.152694.

128. **Westfeld, A., & Pfitzmann, A.** (1999). *Attacks on Steganographic Systems.* Dresden, Germany: Springer-Verlag, Lecture Notes in Computer Science, 1768, 61-75, DOI 10.1007/10719724_5.

129. **Wu, D., & Tsai, W.** (2003). *A steganographic method for digital images using pixel-value differencing.* Amsterdam, Netherlands: Pattern Recognition Letters, 24(9-10), 1613-1626, ISSN 0167-8655, DOI 10.1016/S0167-8655(02)00402-6.

130. **Xu, S., Li, Z., Zhang, Z., & Liu, J.** (2022). *An End-to-End Robust Video Steganography Model Based on a Multi-Scale Neural Network.* Basel, Switzerland: Electronics 11(24):4102, DOI 10.3390/electronics11244102.

131. **Yunxia, L., Shuyang, L., Yonghao, W., Hongguo, Z., & Si, L.** (2018). *Video Steganography: A Review.* Amsterdam, Netherlands: Neurocomputing, 281, 218-241, ISSN 0925-2312, DOI 10.1016/j.neucom.2018.09.091.

132. **Zhang, J., Li, J., & Zhang, L.** (2001). *Video watermark technique in motion vector.* Florianópolis, Brazil: IEEE Computer Society, XIV Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI 2001), 179-182, ISBN 978-0769513300, DOI 10.1109/SIBGRAPI.2001.963053.

133. **Zhang, R., Dong, S., & Liu, J.** (2018). *Invisible steganography via generative adversarial networks.* Berlin, Germany: Multimedia Tools and Applications, 78(7), 8559-8575, ISSN 1380-7501, DOI 10.1007/s11042-018-6951-z.

134. **Zöllner, J., Federrath, H., Klimant, H., Pfitzmann, R. P., Westfeld, A., Wicke, G., & Wolf, G.** (1998). *Modeling the security of steganographic systems.* Berlin, Germany: Springer, Lecture Notes in Computer Science, 1525, 344-354, ISSN 0302-9743, DOI 10.1007/3-540-49380-8_24.

Spis pojęć

Baza wylosowanych steganogramów - baza wylosowanych 150 *oryginalnych steganogramów* odpowiadających 30 wylosowanym numerom z wszystkimi pięcioma *poziomami kodowania*. *Baza wylosowanych steganogramów* stanowi bazę do utworzenia *bazy transkodowanych steganogramów* oraz *bazy końcowych steganogramów*.....189

Baza zaszumionych steganogramów - baza plików *wideo* utworzona poprzez zastosowanie do *oryginalnych steganogramów* z *bazy oryginalnych steganogramów* zaszumienia *szumem Gaussa*. 185

Baza zniekształconych steganogramów - baza plików *wideo* utworzona poprzez zastosowanie do *oryginalnych steganogramów* z *bazy oryginalnych steganogramów* wielu przekształceń i zniekształceń o różnym charakterze, tj. kompresja, rzut perspektywiczny, obrót, osadzenie na tle innego nośnika, zaszumienie. 188

BER (Bit Error Rate) - metryka oceniająca dokładność kodowania jako udział błędnie zakodowanych bitów *komunikatu* w stosunku do wielkości *komunikatu (Katzenbeisser i Petitcolas, 2000)*.....95

Bit - podstawowa jednostka *informacji* w *systemie komunikacyjnym*, która przyjmuje jedną z dwóch możliwych wartości: 0 lub 1, używana do kodowania, przesyłania i przetwarzania *informacji* w *kanałach cyfrowych (Shannon, 1948)*. 14

Bitrate - ilość danych przesyłanych lub przetwarzanych na jednostkę czasu. W kompresji *wideo* określa ilość *bitów* danych na sekundę materiału. Wartość *bitrate* wpływa na jakość i rozmiar pliku. Większy *bitrate* oznacza lepszą jakość i większy rozmiar, a mniejszy *bitrate* odwrotnie. Nowoczesne kodeki dynamicznie dostosowują *bitrate* do złożoności treści, optymalizując kompresję i jakość *wideo*. 114

BPP (bits per pixel) - miara określająca ilość *bitów* używanych do reprezentacji koloru pojedynczego piksela w obrazie cyfrowym. Większe *bpp* oznacza większą głębię koloru i lepsze odwzorowanie kolorów, np. 1 *bpp* oznacza dwa kolory, 8 *bpp* oznacza 256 kolorów, a 24 *bpp* oznacza ponad 16 milionów kolorów.....94

chCapacity - wartość charakterystyczna *pojemności* metody *steganograficznej RAI* definiowana jako iloraz długości kodowanego *komunikatu* do *chTime* - minimalnego czasu potrzebnego na zdekodowanie *komunikatu* ze *steganogramu*. 166

Chi-kwadrat - test statystyczny stosowany do oceny zależności między zmiennymi kategorycznymi umożliwiający porównanie rozkładu danych empirycznych z rozkładem teoretycznym, aby sprawdzić, czy istnieje istotna różnica między tymi rozkładami. W *steganoanalizie* wykorzystywany jest do porównywania oczekiwanego i obserwowanego rozkładu danych w *steganogramie*, który pomaga wykryć potencjalne anomalie mogące sugerować obecność zakodowanego *komunikatu (Abramowitz i Stegun, 1965)*. 71

chIIF - wartość charakterystyczna funkcji *IIF* - wartość funkcji IIF, dla której ilość *informacji* odczytana ze *steganogramu* jest wystarczająca do zdekodowania *komunikatu*. 164

chIteration - wartość charakterystyczna *iteracji dekodujących* - pierwsza *iteracja dekodująca*, dla której *algorytm dekodujący d-RAI* poprawnie dekoduje *komunikat* ze *steganogramu.* 164

chLevel - wartość charakterystyczna *poziomu kodowania* - minimalny *poziom kodowania*, który musi być użyty w *algorytmie kodującym c-RAI*, aby ze *steganogramu* dało się zdekodować *komunikat*. 165

chStand - wartość charakterystyczna *odchylenia standardowego* dla rozkładu częstotliwości występowania *maski kodowania* w *steganogramach wideo* zakodowanych przy użyciu *podstawowego algorytmu kodowania RAI*, równa 0,86%. 214

chTime - wartość charakterystyczna czasu dekodowania - minimalny czas potrzebny do zdekodowania *komunikatu* w metodzie *steganograficznej RAI* definiowany jako czas odtwarzania *steganogramu* od pierwszej *klatki* do *klatki* określonej przez wartość charakterystyczną *chIteration*. 165

CNN (Convolutional Neural Network) - konwolucyjne *sieci neuronowe*, specjalistyczne *sieci neuronowe* efektywne w przetwarzaniu danych o strukturze siatki, np. obrazów. Składają się z warstw konwolucyjnych, które filtrują obraz, warstw aktywacji wprowadzających nieliniowość oraz warstw splotowych redukujących wymiarowość mapy cech. Na końcu znajdują się warstwy w pełni połączone do klasyfikacji lub regresji. *CNN* są powszechnie stosowane w rozpoznawaniu obrazów, analizie *wideo*, przetwarzaniu języka naturalnego i generowaniu treści. Wymagają dużych zbiorów danych do skutecznego trenowania *(Krizhevsky, Sutskever i Hinton, 2017; LeCun, Bengio i Hinton, 2015)*. 47

Cyberbezpieczeństwo - ochrona informacji i systemów informacyjnych przed nieautoryzowanym dostępem, wykorzystaniem, ujawnieniem, zakłóceniem, modyfikacją lub zniszczeniem w celu zapewnienia poufności, integralności i dostępności *(Nieles, Dempsey i Pillitteri, 2017)*. 51

Daltonizm - zaburzenie widzenia barw polegające na trudności w rozróżnianiu kolorów, najczęściej czerwonego i zielonego. Jest to dziedziczna wada genetyczna związana z mutacją chromosomu X, przez co częściej dotyka mężczyzn *(Gordon, 1998)*. 201

DCT (Discrete Cosine Transform) - rodzaj *transformaty*, która przekształca sygnały w dziedzinie czasu lub przestrzeni na sygnały w dziedzinie częstotliwości przy użyciu funkcji cosinusowych. *DCT* jest szczególnie efektywna w analizie i kompresji danych, zwłaszcza obrazów i dźwięków, dzięki swojej zdolności do koncentracji energii sygnału w kilku współczynnikach. Metody wykorzystujące *DCT* są często stosowane w *steganografii obrazowej (Ahmed, Natarajan i Rao, 1974)*. 23

DIKW (Data Information Knowledge Wisdom) - klasyczna piramida: *dane, informacja, wiedza, mądrość*. Dane to surowe, nieprzetworzone fakty i liczby bez kontekstu. *Informacje* to dane zorganizowane i przetworzone, które nabierają znaczenia. Wiedza to *informacje*, które zostały zrozumiane i połączone z doświadczeniem i kontekstem. Mądrość to praktyczne zastosowanie wiedzy, prowadzące do mądrych decyzji *(Ackoff, 1989; Rowley, 2007)*. 14

Domena częstotliwościowa - sposób reprezentacji *nośnika*, dla którego najpierw wyznaczane są *transformaty* częstotliwościowe, a następnie są one poddawane analizie lub modyfikacji. 27

Domena przestrzenna - sposób reprezentacji *nośnika*, w którym operacje analizy lub modyfikacji są wykonywane poprzez bezpośrednie manipulowanie jego wartościami składowymi, np. pikselami. 27

DWT (Discrete Wavelet Transform) - metoda przekształcania sygnału z dziedziny czasu na reprezentację w dziedzinie częstotliwości i czasu przy użyciu falek. *DWT* jest używana w analizie sygnałów, kompresji danych, przetwarzaniu obrazów i innych dziedzinach inżynierii i nauki *(Daubechies, 1992; Akansu i Haddad, 2000)*....39

End-to-end - model *sieci neuronowej* przetwarzającej surowe dane wejściowe i generującej pożądane wyjścia w sposób zintegrowany, bez ręcznego projektowania cech pośrednich. *Sieć neuronowa end-to-end* uczy się istotnych reprezentacji i przekształceń podczas treningu, co upraszcza modelowanie i zwiększa jej elastyczność *(Goodfellow, Bengio i Courville, Deep Learning, 2016)*....109

Entropia - miara nieuporządkowania *informacji* - im większa *entropia*, tym większa nieprzewidywalność i niepewność *informacji (Shannon, 1948)*....13

FFmpeg - narzędzie do przetwarzania multimediów obsługujące kodowanie, dekodowanie, transkodowanie i strumieniowanie plików *audio* i *wideo (FFmpeg, 2024)*....180

FPS (frames per second) - liczba *klatek* wyświetlanych na sekundę, wpływająca na płynność, jakość i percepcję wizualną *wideo*. Mieści się w zakresie od 24 fps do 120 fps i jest dobierana tak, aby widz miał wrażenie oglądania ciągłego, płynnego *wideo*, np. 24 fps używany jest w filmach kinowych, 30 fps w internecie, a 60 fps w transmisjach sportowych. 133

FT (Fourier Transform) - Transformata Fouriera, matematyczne narzędzie używane do analizy funkcji lub sygnałów w dziedzinie częstotliwości. *FT* przekształca funkcję czasową lub przestrzenną na reprezentację w dziedzinie częstotliwości, umożliwiając analizę jej składowych harmonicznych. Jest szeroko stosowana w wielu dziedzinach, takich jak inżynieria, fizyka, przetwarzanie sygnałów, analiza obrazu *(Bracewell, 1986; Pinsky, 2009)*....39

GAN (Generative Adversarial Networks) - generatywne sieci przeciwstawne, technika głębokiego uczenia składająca się z dwóch rywalizujących *sieci neuronowych*: generatora, który tworzy dane, oraz dyskryminatora, który ocenia, czy dane są prawdziwe czy wygenerowane. Proces ten prowadzi do coraz lepszego generowania danych przez generator *(Goodfellow i inni, 2014)*....47

H.264 - znany również jako *MPEG-4 Part 10* lub *AVC* (*Advanced Video Coding*). Jest jednym z najczęściej stosowanych standardów kompresji *wideo*. *H.264* oferuje wysoką jakość obrazu przy stosunkowo małym *bitrate*. Technika ta wykorzystuje metody *kodowania międzyklatkowego* oraz *kodowania wewnątrzklatkowego*, które pozwalają na redukcję *redundancji* przestrzennej i czasowej. Dzięki temu może osiągnąć dużą kompresję, przy zachowaniu wysokiej jakości *(ITU-T, H.264, 2011)*....117

H.265 - znany również jako *HEVC (High Efficiency Video Coding)*. Jest następcą *H.264* oferując większą efektywność kompresji i mniejszy o około 50% *bitrate,* przy zachowaniu podobnej jakości obrazu. Technika ta wykorzystuje zaawansowane metody *predykcji międzyklatkowej* oraz *predykcji wewnątrzklatkowej* stosując mi.in mechanizmy dynamicznego podziału obrazu na bloki o różnych rozmiarach, co pozwala na lepsze dopasowanie do złożonych scen w *wideo (ITU-T, H.265, 2019)*. 117

H.266 - znany również jako *VVC (Versatile Video Coding)*. Jest następcą *H.265* i oferuje nawet o 50% lepszą efektywność kompresji niż *H.265*, przy zachowaniu tej samej jakości obrazu. Wykorzystuje zaawansowane metody *predykcji międzyklatkowej*, adaptacyjne podziały bloków obrazu i adaptacyjną optymalizację *wektorów ruchu (ITU-T, H.266, 2024)*. 117

HCF (Histogram Characteristic Function) - technika *steganoanalizy* wykorzystująca analizę *histogramu* do identyfikacji potencjalnych anomalii w *steganogramie* mogących wskazywać na zastosowanie *steganografii (Harmsen i Pearlman, 2003)*. 32

Histogram - graficzny wykres słupkowy przedstawiający rozkład danych, gdzie oś X pokazuje przedziały wartości, a oś Y - częstotliwość ich występowania. Jest powszechnie stosowany w statystyce i analizie danych umożliwiając wizualizację rozkładu danych, identyfikację koncentracji wartości oraz wykrywanie anomalii *(Howitt i Cramer, 2007)*. 64

IDCT (Inverse Discrete Cosine Transform) - odwrotna *transformata* do *DCT*, która przekształca dane z *domeny częstotliwościowej* do *domeny przestrzennej*. *IDCT* jest stosowana w celu rekonstrukcji oryginalnych wartości pikseli po ich przetworzeniu przez *DCT (Ahmed, Natarajan i Rao, 1974)*. 39

IIF (Information Incremental Function) - funkcja opisująca przyrosty *informacji* w kolejnych iteracjach *algorytmu dekodującego RAI*. 163

Informacja - zmniejszenie niepewności związane z wystąpieniem konkretnego zdarzenia w systemie probabilistycznym kwantyfikowana przez pojęcie *entropii* zdefiniowane w *teorii informacji* Shannona *(Shannon, 1948; Anderson i Johannesson, 2006)*. 12

Ishihara test - narzędzie do diagnozy daltonizmu, składające się z tablic z kolorowymi kropkami tworzącymi liczby lub kształty. Osoby z prawidłowym widzeniem rozpoznają je, podczas gdy osoby z zaburzeniami barw mają trudności. Test pozwala określić rodzaj i stopień deficytu widzenia barw (Ishihara, 1917)..... 201

JPEG (Joint Photographic Experts Group) - standard *kompresji stratnej* dla obrazów cyfrowych, który zmniejsza rozmiar plików poprzez odrzucenie niektórych danych minimalizując utratę jakości. Proces kompresji obejmuje konwersję kolorów, podział obrazu na bloki 8x8 pikseli, zastosowanie dla nich dyskretnej *transformaty* cosinusowej *DCT* wraz z *kwantyzacją* i *kodowaniem entropijnym* (ITU-T, JPEG (T.81), 1992)..... 39

Kanał - medium lub droga, przez którą sygnał zawierający zakodowaną *informację* jest przesyłany od *nadajnika* do *odbiornika*. 13

Kanał analogowy - występująca w *scenariuszu zniekształceń analogowych* część *kanału* pomiędzy *nadawcą* a *odbiorcą*, w ramach którego *steganogram*

przekazywany jest pomiędzy *nadajnikiem analogowym* a *odbiornikiem analogowym* w sposób analogowy, np. poprzez wyświetlenie *steganogramu wideo* na ekranie odbiornika telewizyjnego. 130

Kanał cyfrowy - medium do przesyłania pomiędzy *nadawcą* a *odbiorcą steganogramu* w postaci pliku cyfrowego, którego zadaniem jest efektywny, niezawodny i bezpieczny transfer cyfrowych danych, minimalizując potencjalne zakłócenia i straty *informacji*. 82

Klatka - podstawowa jednostka w *wideo*, która przedstawia pojedynczy statyczny obraz, wyświetlany w określonym czasie, tworząc wrażenie ruchu po zestawieniu z kolejnymi *klatkami.* 19

Klucz kryptograficzny - tajny ciąg *bitów* używany w algorytmach *kryptograficznych* do *szyfrowania* i deszyfrowania informacji, zapewniający bezpieczeństwo komunikacji poprzez kontrolowanie dostępu do zaszyfrowanych danych *(Chandra, Paira, Alam i Sanyal, 2014)*. 50

Kodek - oprogramowanie do kodowania i dekodowania, w tym do kompresji i dekompresji obrazów i *wideo*, dzięki któremu można efektywnie przechowywać i przesyłać wysokiej jakości *wideo* przy minimalnym zużyciu zasobów. 114

Kodowanie entropijne - technika *kompresji bezstratnej* na podstawie częstotliwości występowania symboli, np. kodowanie Huffmana - przypisuje krótsze kody binarne częściej występującym symbolom i dłuższe rzadziej występującym lub kodowanie arytmetyczne - koduje całe ciągi symboli jako pojedyncze liczby zmiennoprzecinkowe z przedziału [0,1) dzieląc ten przedział na podprzedziały proporcjonalne do prawdopodobieństwa wystąpienia symboli *(Nelson i Gailly, 1995; Duda, Tahboub, Gadgil i Delp, 2015)*. 115

Kodowanie międzyklatkowe (Inter-frame coding) - technika kompresji *wideo* wykorzystująca redundancję czasową między *klatkami* i mechanizm *predykcji międzyklatkowej*. Kodowane są tylko różnice między *klatkami*, co pozwala na znaczne zmniejszenie ilości danych. Kluczowe *klatki* (I-frames) są w pełni kodowane, a pozostałe (P-frames, B-frames) bazują na różnicach względem innych *klatek*. Dzięki temu uzyskuje się większą efektywność kompresji *(Tekalp, 1995; Gao i Ma, 2015)*. 115

Kodowanie transformatowe - technika stosowana w kompresji *wideo* zmniejszająca *redundancję danych* poprzez przekształcenie sygnału z *domeny przestrzennej* do *domeny częstotliwościowej*. Proces obejmuje podział obrazu na bloki, zastosowanie *transformaty*, np. *DCT* i *kwantyzację* jej współczynników poprzez ich zaokrąglenie do określonych poziomów. *(Muchahary, Mondal, Parmar, Borah i Majumder, 2015)*. 115

Kodowanie wewnątrzklatkowe (Intra-frame coding) - technika kompresji *wideo*, w której każda *klatka* jest kodowana niezależnie, podobnie jak obrazy statyczne, ale wykorzystująca mechanizm *predykcji wewnątrzklatkowej*, czyli przewidywanie wartości pikseli w danej *klatce* na podstawie wartości sąsiednich pikseli. Choć jest mniej efektywne pod względem kompresji w porównaniu do *kodowania międzyklatkowego*, jego zaletą jest większa odporność na błędy i łatwy dostęp do poszczególnych *klatek*, co jest istotne np. w edycji i szybkim przeszukiwaniu *wideo*. 115

Kody liniowe - kody korekcyjne, które zabezpieczają dane przed błędami transmisji. Przekształcają bloki danych wejściowych na bloki kodowe za pomocą operacji liniowych, umożliwiając wykrywanie i korekcję błędów poprzez dodanie *bitów* kontrolnych, np. kody Hamminga i Reeda-Solomona *(Ryan i Lin, 2009)*. 104

Kompresja bezstratna - metoda redukcji rozmiaru danych, polegająca na transformacji *informacji* do postaci o zmniejszonym zapotrzebowaniu na liczbę *bitów*, przy jednoczesnym zapewnieniu możliwości pełnego odtworzenia oryginalnej treści bez jakichkolwiek strat. 15

Kompresja stratna - nieodwracalna metoda redukcji danych, polegająca na eliminacji informacji mniej istotnych z punktu widzenia ludzkiej percepcji. Pozwala ona uzyskać znacznie większy stopień kompresji niż metody bezstratne, wprowadzając zniekształcenia dobrane tak, aby percepcja wzrokowa lub słuchowa pozostała zbliżona do oryginału. 91

Komunikat - *informacja*, którą *nadawca steganogramu* chce w sposób sekretny przekazać *odbiorcy*. 17

Końcowy steganogram - w *scenariuszu zniekształceń analogowych* utworzony przy pomocy *odbiornika analogowego* nowy plik cyfrowy, którym dysponuje *odbiorca* oraz potencjalnie *podsłuchujący kanał analogowy*. 131

Krok kodowania - parametr *podstawowego algorytmu kodującego RAI* określający, w co której *klatce* ma być kodowany *komunikat*. 147

Kryptografia - dziedzina nauki zajmująca się zabezpieczaniem *informacji* poprzez ich *szyfrowanie*, czyli przekształcanie w formę nieczytelną dla osób nieuprawnionych, z użyciem algorytmów i *kluczy kryptograficznych*, umożliwiająca poufną komunikację i ochronę danych *(Blackledge, 2011; Beckett, 1988)* 50

Kwantyzacja - proces przekształcania sygnałów ciągłych w dyskretne poprzez zaokrąglanie wartości do najbliższych poziomów w określonym zbiorze. Stosowana w przetwarzaniu sygnałów, kompresji danych i grafice komputerowej, umożliwia redukcję ilości danych kosztem wprowadzenia niewielkich zniekształceń *(Bennett W., 1948)*. 115

Liniowe i nieliniowe filtry wysokoprzepustowe - filtry stosowane do wykrywania ukrytych *informacji* przez analizę wysokoczęstotliwościowych komponentów obrazu. Przykładem filtra liniowego jest filtr Butterwortha, a nieliniowego filtr medianowy *(Antoniou, 1993)*. 80

LSB (Least Significant Bit) - najmniej znaczący *bit* o najmniejszej wartości wagowej w binarnej reprezentacji liczby lub technika *steganograficzna* polegająca na kodowaniu kolejnych *bitów komunikatu* w najmniej znaczących *bitach* wybranych parametrów *nośnika (Chan i Cheng, 2004)*. 20

Łańcuch Markowa - *proces Markowa* z dyskretną przestrzenią stanów *(Meyn i Tweedie, 2009)*. 93

Makroblok - podstawowa jednostka przetwarzania obrazu stosowana w standardach *MPEG*, *H.264* i *H.265*. Składa się zazwyczaj z 16x16 pikseli i odgrywa kluczową rolę w *predykcji ruchu*, umożliwiając efektywne kodowanie różnic między *klatkami* za pomocą *wektorów ruchu*. Przy wykorzysta Pevný niu *DCT*,

kwantyzacji i *kodowania entropijnego makrobloki* pozwalają na znaczną redukcję danych przy zachowaniu jakości obrazu. 112

Maska - trójka symboli "+- " lub liczb {1,2,4} opisująca różnicę pomiędzy składowymi kolorystycznymi dwóch pikseli A - B. Wartości " " lub 1 oznaczają, że piksele A i B mają ten sam składowy kolor, wartości "-" lub 2 oznaczają, że piksel A ma mniejszą daną składową kolorystyczną niż piksel B, a wartość "+" lub 4 oznacza, że piksel B ma mniejszą składową kolorystyczną niż piksel A. 145

Maska kodująca - używana w *podstawowym algorytmie kodującym RAI maska* kodująca bit 0 lub *maska* kodująca bit 1 *komunikatu*.....146

Metadane - dane opisujące właściwości i strukturę innych danych, takie jak kontekst, pochodzenie lub format, umożliwiające organizację, wyszukiwanie i zarządzanie *informacjami*. 132

MJPEG (Motion JPEG) - technika kompresji *wideo* wykorzystująca mechanizmy *kodowania wewnątrzklatkowego*, w której każda *klatka* jest kompresowana jako oddzielny obraz *JPEG*. *MJPEG* jest prosty w implementacji i oferuje wysoką jakość obrazu, ale jego efektywność kompresji jest niższa w porównaniu do standardów takich jak *H.264* lub *H.265* (ITU-T, MJPEG (T.802), 2006). 117

MPEG-4 Part 2 - standard kompresji *wideo* znany również jako *MPEG-4 Visual*. Wykorzystuje m.in. techniki *kodowania międzyklatkowego* oraz *kodowanie transformatowe (ITU-T, H.263, 1996).* 117

MSE (Mean Squared Error) - metryka oceniająca podobieństwo *steganogramu* do *nośnika* zdefiniowana jako średnia kwadratów różnic między nimi. Niższe wartości *MSE* wskazują na większe podobieństwo *steganogramu* do *nośnika*, co jest skorelowane z większą *niewykrywalnością*. *(Cheddad, Condell, Curran i Kevitt, 2010)*.....85

Nadajnik - element *systemu komunikacyjnego*, który przekształca zakodowaną *informację* od *nadawcy* w sygnał odpowiedni do przesłania przez *kanał.* 13

Nadajnik analogowy - urządzenie przetwarzające cyfrowy *steganogram* na postać analogową, np. ekran odbiornika telewizyjnego. 131

Nadawca - element *systemu komunikacyjnego*, który generuje, koduje i przekazuje *informacje* poprzez wybrany *kanał* do *odbiorcy*. 13

Nagłówki - *metadane* opisujące strukturę i format pliku cyfrowego lub pakietu danych, np. typ danych i rozmiar. 132

Niewykrywalność (Undetectability) - miara zdolności techniki *steganograficznej* do ukrycia *komunikatu* w *steganogramie* w sposób, który uniemożliwia wykrycie jego obecności zarówno subiektywnymi testami percepcji ludzkich zmysłów, jak też obiektywnymi metodami matematycznymi. Duża *niewykrywalność* oznacza, że zmiany w *nośniku* wprowadzone przez techniki *steganograficzne* są mniej zauważalne oraz trudniejsze do wykrycia *(Katzenbeisser i Petitcolas, 2000; Huynh-Thu i Ghanbari, 2008; Wang, Bovik, Sheikh i Simoncelli, 2004; Shannon, 1948)*.....85

Nośnik - oryginalny plik cyfrowy, w którym w wyniku zastosowania technik *steganograficznych* może być ukryty *komunikat*. 16

Odbiorca - element *systemu komunikacyjnego*, który odbiera, dekoduje i interpretuje *informacje* przekazane przez *nadawcę* poprzez wybrany *kanał*. 13

Odbiornik - element *systemu komunikacyjnego*, który odbiera sygnał z *kanału*, dekoduje go i przekształca w *informację* zrozumiałą dla *odbiorcy*..... 13

Odbiornik analogowy - urządzenie przetwarzające analogową postać *steganogramu* ponownie do postaci pliku cyfrowego, np. kamera smartfonu..... 131

Odchylenie standardowe - miara rozproszenia danych wokół średniej w zbiorze. Określa, jak bardzo poszczególne wartości różnią się od średniej arytmetycznej. Mała wartość odchylenia standardowego wskazuje, że większość danych jest blisko średniej, podczas gdy duża wartość oznacza większe rozproszenie wyników *(Kenney, 1954)*..... 207

Odporność (Robustness) - miara zdolności techniki *steganograficznej* do utrzymania *komunikatu* w *steganogramie* pomimo różnych przekształceń, takich jak np. kompresja, zmiany formatu, filtrowanie, obracanie lub skalowanie. Większa *odporność* oznacza, że *komunikat* po takich modyfikacjach z większym prawdopodobieństwem pozostanie nienaruszony i nieusuwalny bądź ilość pozostałej *informacji* w *komunikacie* będzie większa. *(Provos i Honeyman, 2003; Cox, Miller, Bloom, Fridrich i Kalker, 2007; Fridrich, Steganography in Digital Media: Principles, Algorithms, and Applications, 2009; Katzenbeisser i Petitcolas, 2000)*..... 95

OpenCV (Open Source Computer Vision Library) - biblioteka programistyczna zaprojektowana do realizacji zadań związanych z komputerowym przetwarzaniem obrazów i widzeniem maszynowym. Oferuje szeroki zestaw algorytmów do analizy obrazów i *wideo*, takich jak detekcja i rozpoznawanie obiektów, segmentacja, śledzenie ruchu, rekonstrukcja 3D, oraz przetwarzanie obrazu w czasie rzeczywistym *(Opencv.org, 2024)*..... 169

Oryginalny steganogram - w *scenariuszu zniekształceń analogowych* utworzony przez *nadawcę* plik cyfrowy będący pierwotnym *steganogramem*, który *nadawca* wysyła do *odbiorcy*..... 131

Patchwork - technika *steganograficzna* polegająca na wprowadzaniu zmian w jasności wybranych w sposób pseudolosowy par pikseli obrazu tak, aby zakodować kolejne *bity komunikatu (Bender, Gruhl, Morimoto i Lu, 1996)*..... 33

percLevel - najmniejszy *poziom kodowania steganogramu wideo* utworzonego za pomocą techniki *RAI*, przy którym więcej niż 10% badanych osób widzi wprowadzone w *wideo* modyfikacje *steganograficzne*..... 202

PNG (Portable Network Graphics) - rastrowy format plików graficznych i system *kompresji bezstratnej* danych graficznych *(ISO-IEC, 2004)*..... 20

Podsłuchujący - osoba postronna, która podsłuchuje transmisję pomiędzy *nadawcą* i *odbiorcą*, a która nie powinna dowiedzieć się, że oprócz oficjalnych *informacji nadawca* przekazuje *odbiorcy komunikat* zakodowany w *steganogramie*..... 83

Podstawowy algorytm dekodujący d-RAI - algorytm dekodujący *komunikat* ze *steganogramu* utworzonego przy pomocy *podstawowego algorytmu kodującego c-RAI*, zakładający pewną wiedzę o zastosowanych parametrach algorytmu kodującego..... 153

Podstawowy algorytm kodujący c-RAI - algorytm kodujący *komunikat* w *steganogramie wideo* metodą *RAI*..... 147

Podstawowy scenariusz steganologiczny - scenariusz przesyłania pomiędzy *nadawcą* a *odbiorcą steganogramu* w postaci pliku cyfrowego poprzez nieprzerwany, ciągły *kanał cyfrowy.* 82

Pojemność (Capacity) - miara ilości *informacji*, którą można ukryć w *nośniku* cyfrowym bez zauważalnych zmian jego jakości. Zależy m.in. od właściwości *nośnika* i użytej techniki *steganografii*. Duża *pojemność* umożliwia ukrycie większej ilości *informacji*, ale może jednocześnie zwiększać ryzyko wykrycia zmniejszając *niewykrywalność (Fridrich, Steganography in Digital Media: Principles, Algorithms, and Applications, 2009; Cox, Miller, Bloom, Fridrich i Kalker, 2007)*..... 93

Pojemność kanału - maksymalna ilość *informacji*, którą *kanał* może przesłać w jednostce czasu, określająca wartość przepustowości *systemu komunikacyjnego (Shannon, 1948; Cover i Thomas, 1991)*. 14

Poziom kodowania - parametr *podstawowego algorytmu kodującego c-RAI* określający wielkość zmian dokonywanych w pikselach kodowanych *klatek steganogramu*. Im większy *poziom kodowania*, tym zmiany są bardziej znaczące. 144

Predykcja międzyklatkowa - technika wykorzystywana przez *kodeki* w procesie *kodowania międzyklatkowego*, która umożliwia przewidywanie zawartości kolejnych *klatek* na podstawie różnic względem poprzednich *klatek*, redukując ilość danych potrzebnych do przechowywania i transmisji sekwencji *(Tekalp, 1995; Gao i Ma, 2015; Richardson, 2003)*. 115

Predykcja wewnątrzklatkowa - metoda przewidywania wartości pikseli w bieżącej *klatce* na podstawie sąsiednich pikseli stosowana w procesie *kodowania wewnątrzklatkowego*. 115

Proces Markowa - proces stochastyczny, w którym przy decyzji o przejściu do kolejnego stanu systemu uwzględnia się wyłącznie obecny stan, co jest określane jako własność braku pamięci *(Meyn i Tweedie, 2009)*. 93

Przekształcony steganogram - otrzymany w *podstawowym scenariuszu steganologicznym* przez *odbiorcę* przekształcony lub zniekształcony plik cyfrowy wysłany pierwotnie przez *nadawcę* jako *oryginalny steganogram*. 83

Przeuczenie sieci neuronowej - zjawisko, w którym *sieć neuronowa* uczy się zbyt dokładnie na danych treningowych, tracąc zdolność do generalizacji i skutecznego działania na nowych, nieznanych danych, co prowadzi do spadku jej efektywności *(Burnham i Anderson, 2004)*..... 79

PSNR (Peak Signal-to-Noise Ratio) - metryka oceniająca podobieństwo *steganogramu* do *nośnika* (szczytowy stosunek sygnału do szumu). Im większe wartości *PSNR,* tym mniejsze są zniekształcenia *nośnika* i większe podobieństwo *steganogramu* do *nośnika*, co oznacza większą *niewykrywalność*, czyli skuteczniejsze ukrycie *komunikatu (Katzenbeisser i Petitcolas, 2000; Huynh-Thu i Ghanbari, 2008)*..... 87

PVD (Pixel Value Differencing) - adaptacyjna technika *steganograficzna*, która ukrywa *komunikat* w *steganogramie* poprzez modyfikację różnic jasności między sąsiadującymi pikselami *(Wu i Tsai, 2003).* 36

QR code (Quick Response code) - dwuwymiarowy kod kreskowy opracowany przez japońską firmę Denso Wave. Dzięki swojej matrycowej strukturze kody *QR* mogą przechowywać znacznie więcej danych niż jednowymiarowe kody kreskowe. Cechują się szybkością skanowania, możliwością przechowywania różnorodnych informacji i odpornością na częściowe uszkodzenia dzięki mechanizmom korekcji błędów *(DensoWave, 2024)*. 168

RAI (Robust Adaptive Incremental) - zaproponowana w niniejszej pracy nowa technika *steganografii wideo*. 137

Redundancja - nadmiarowość *informacji* w systemie, która zwiększa niezawodność przekazu, umożliwiając jego odtworzenie w przypadku błędów lub utraty danych. W *steganografii* wykorzystuje się ją do ukrywania *komunikatu*, np. zmieniając nadmiarowe *bity* bez zauważalnej utraty jakości *steganogramu*. 114

Reszta kompresji (residual) - różnica między rzeczywistymi a przewidywanymi wartościami pikseli w *klatkach* kodowana i przesyłana w *wideo*, umożliwiająca rekonstrukcję obrazu podczas procesu dekompresji. 112

RGB (Red, Green, Blue) - model kolorów używany w elektronice i technologii cyfrowej do reprezentacji i wyświetlania kolorów. Model *RGB* bazuje na mieszaniu trzech podstawowych kolorów światła: czerwonego (Red), zielonego (Green) i niebieskiego (Blue). Kombinacja tych trzech kolorów w różnych proporcjach pozwala na stworzenie szerokiej gamy kolorów *(Hirsch, 2004)*. 20

Rozdzielczość wideo - liczba pikseli w poziomie i pionie wpływająca na jakość i ostrość obrazu, wyrażana jako szerokość x wysokość, np. 1920x1080 - Full HD lub 3840x2160 - 4K. Im większa *rozdzielczość wideo* tym bardziej szczegółowy jest obraz *wideo*, ale tym większe jest równocześnie zapotrzebowanie na przepustowość *kanału* oraz moc obliczeniową do przetwarzania takiego *wideo*..... 132

Scenariusz zniekształceń analogowych - scenariusz przesyłania pomiędzy *nadawcą* a *odbiorcą steganogramu* początkowo przez *kanał cyfrowy*, a od pewnego momentu poprzez *kanał analogowy*. W tym scenariuszu *końcowy steganogram* jest nowo utworzonym przez odbiorcę plikiem cyfrowym różniącym się istotnie od *oryginalnego steganogramu* utworzonego pierwotnie przez *nadawcę*. 130

Serwis społecznościowy - internetowa platforma umożliwiająca użytkownikom tworzenie profili, nawiązywanie kontaktów oraz dzielenie się treściami, takimi jak posty, zdjęcia i filmy. Przykładami serwisów społecznościowych są YouTube, Facebook i Twitter (X). 111

Sieć neuronowa - model obliczeniowy inspirowany mózgiem, stosowany w obszarach sztucznej inteligencji. Składa się z warstw neuronów, które uczą się rozpoznawać wzorce poprzez dostosowywanie wag połączeń. *Sieć neuronowa* jest używana m.in. do rozpoznawania obrazów ucząc się generalizować na dużych zbiorach obrazów *(Goodfellow, Bengio i Courville, Deep Learning, 2016)*. 47

SSIM (Structural Similarity Index) - metryka używana do oceny *niewykrywalności* poprzez określanie podobieństwa strukturalnego *steganogramu* do oryginalnego *nośnika* uwzględniającego jasność, kontrast i strukturę w sposób bardziej zbliżony do percepcji ludzkich zmysłów *(Wang, Bovik, Sheikh i Simoncelli, 2004)*. 88

Szum - mogące pochodzić z różnych źródeł niepożądane zakłócenia w przekazie *informacji*, które obniżając jej jakość i dokładność, utrudniają lub zniekształcają jej odbiór oraz interpretację *(Shannon, 1948)*. 13

Szum Gaussa - model zakłóceń oparty na *rozkładzie normalnym (Patel i Read, 1996)*, gdzie wartości losowo rozkładają się wokół średniej. Charakteryzuje się symetrycznym rozkładem i jest powszechnie stosowany w analizie sygnałów i przetwarzaniu obrazów. Dzięki swoim właściwościom dobrze opisuje naturalne zjawiska losowe, co czyni go kluczowym w modelowaniu zakłóceń w systemach *(Jain, 1989)*. 127

Szyfrowanie - proces przekształcania *informacji* w sposób, który uniemożliwia jej odczytanie bez odpowiedniego *klucza kryptograficznego*, stosowany w *kryptografii* w celu ochrony danych przed nieautoryzowanym dostępem *(Beckett, 1988)*. 50

Teoria informacji - dziedzina nauki zajmująca się kwantyfikacją, przechowywaniem, przesyłaniem i przetwarzaniem *informacji*, wykorzystująca matematyczne modele i narzędzia do analizy transmisji sygnałów, kompresji danych, przetwarzania i ukrywania *informacji (Shannon, 1948; Anderson i Johannesson, 2006)*. 12

Transformata - operacja matematyczna przekształcająca jedną funkcję w inną funkcję upraszczając problem i ułatwiając jego analizę i obliczenia *(Polyanin i Manzhirov, 2008)*. 27

Uczenie maszynowe - dziedzina sztucznej inteligencji, w której systemy uczą się na podstawie danych, zamiast być bezpośrednio programowane. Modele przetwarzają dane i dostosowują swoje parametry, aby poprawić wydajność w zadaniach takich jak klasyfikacja, prognozowanie lub rozpoznawanie wzorców *(Goodfellow, Bengio i Courville, Deep Learning, 2016)*. 47

VOD (Video on demand) - usługa umożliwiająca użytkownikom wybieranie i oglądanie *wideo* na żądanie niezależnie od harmonogramu nadawania i rodzaju urządzenia. Przykładami serwisów VOD są YouTube i Netflix. 116

VP9 - otwarty standard kompresji *wideo*, prekursor *kodeka AV1*. Wykorzystywany i wspierany m.in. przez liczne przeglądarki internetowe i platformy *VOD (VP9, 2024)*. 117

Wektor ruchu - podstawowy element wykorzystywany przez *kodeki* w *predykcji międzyklatkowej*. *Wektor ruchu* opisuje przesunięcie bloku pikseli pomiędzy kolejnymi *klatkami*, umożliwiając efektywne kodowanie różnic między *klatkami* i redukcję ilości danych potrzebnych do reprezentacji ruchu *(Gao i Ma, 2015)*. 116

Wideo - zapis ruchomych obrazów i dźwięku w formacie cyfrowym, skompresowany przy użyciu *kodeka*, np. *H.264* i zapisany w kontenerze, np. MP4, zarządzającym synchronizacją między obrazem, dźwiękiem i napisami *(Hanzo, Cherriman i Streit, 2007)*. 26

Znakowanie wodne - technika zabezpieczania danych polegająca na osadzaniu ukrytej *informacji (znaku wodnego)* w pliku multimedialnym, np. obrazie, *wideo* lub dźwięku w sposób zauważalny lub niezauważalny, służąca ochronie praw autorskich, zapewnieniu autentyczności lub śledzeniu nieautoryzowanego użycia *(Shih, 2017)*. 36

Spis rysunków

Spis tabel

Spis kodów