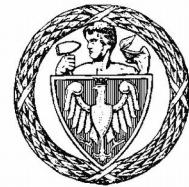


Politechnika Warszawska

WYDZIAŁ ELEKTRONIKI
I TECHNIK INFORMACYJNYCH



Instytut Informatyki

Praca dyplomowa inżynierska

na kierunku Informatyka
w specjalności Inżynieria Systemów Informatycznych

Wykrywanie fałszerstw w obrazach cyfrowych

Andrzej Dackiewicz

Numer albumu 253070

promotor
prof. dr hab. inż. Jan Zabrodzki

Warszawa 2017

STRESZCZENIE

Niniejsza praca opisuje problematykę związaną z wykrywaniem fałszerstw w obrazach cyfrowych. Dokument ten jest jednocześnie opisem oprogramowania stworzonego do znajdywania zmian wykonanych na obrazach naturalnych przechowywanych w plikach o rozszerzeniu jpg.

Na początku omówione zostały podstawy teoretyczne, wymagane do zrozumienia metod użytych w programie. Kolejnymi częściami pracy są opis działania wykorzystanych zabezpieczeń, instrukcja obsługi i prezentacja działania programu dla przygotowanych przypadków testowych.

Słowa kluczowe: JPEG, zdjęcie cyfrowe, exif, wykrywanie fałszerstw w obrazach cyfrowych

Detection of digital image forgery

This thesis addresses the problems associated with digital image forgery detection. This document also describes a program that was created for detecting changes in natural images that are stored in jpg files.

In the beginning basic theory that is necessary for understanding of methods used in program is being discussed. Next parts consist of descriptions of how each of methods used in program work, user's manual and results of work for some test cases.

Keywords: JPEG compression, digital image, exif, digital image forgery detection

Spis treści

| | |
|--|----|
| 1 Wprowadzenie..... | 9 |
| 2 Podstawowe informacje..... | 10 |
| 2.1 Fałszerstwo cyfrowe..... | 10 |
| 2.2 Kompresja JPEG..... | 10 |
| 2.3 Metadane exif..... | 12 |
| 2.4 Histogram obrazu..... | 13 |
| 2.5 Podział metod wykrywania fałszerstw ze względu na działanie..... | 15 |
| 2.5.1 Metody statyczne..... | 15 |
| 2.5.2 Metody dynamiczne..... | 15 |
| 2.5.3 Metody steganograficzne..... | 15 |
| 2.5.4 Metody kryptograficzne..... | 16 |
| 2.6 Najpopularniejsze metody wykrywania zmian w obrazach cyfrowych.... | 16 |
| 2.6.1 Metody działające na pikselach obrazu..... | 16 |
| 2.6.2 Metody opierające swe działanie na formacie zapisu..... | 17 |
| 2.6.3 Metody opierające swe działanie na właściwościach fizycznych... | 18 |
| 3 Koncepcja rozwiązania..... | 19 |
| 3.1 Wymagania funkcjonalne..... | 19 |
| 3.2 Wymagania niefunkcjonalne..... | 20 |
| 3.3 Plan realizacji wymagań..... | 20 |
| 3.4 Założenia związane z korzystaniem ze stworzonego programu..... | 21 |
| 4 Struktura aplikacji..... | 22 |
| 5 Zabezpieczenia wykorzystane w programie..... | 24 |
| 5.1 Wykorzystanie pól metadanych EXIF..... | 24 |
| 5.2 Wykorzystanie histogramów obrazu..... | 27 |
| 5.3 Zależności między pikselami..... | 30 |
| 5.4 Znak zabezpieczający..... | 33 |
| 5.5 Wykrywanie rotacji obrazu..... | 35 |

| | |
|--|----|
| 5.6 Zabezpieczenie wielkości obrazu..... | 36 |
| 5.7 Wpływ zabezpieczeń na wygląd obrazu..... | 38 |
| 6 Instrukcja obsługi programu..... | 40 |
| 6.1 Wybór pliku obrazu do badań..... | 41 |
| 6.2 Informacje exif i histogramy obrazu..... | 43 |
| 6.3 Zabezpieczanie wybranego obrazu..... | 45 |
| 6.4 Weryfikacja oryginalności obrazu..... | 46 |
| 7 Przykłady działania programu..... | 47 |
| 7.1 Przykład 1..... | 47 |
| 7.2 Przykład 2..... | 52 |
| 7.3 Przykład 3..... | 56 |
| 7.4 Przykład 4..... | 60 |
| 8. Plany dalszego rozwoju programu..... | 66 |
| 9. Podsumowanie..... | 67 |
| Bibliografia..... | 68 |
| Spis rysunków..... | 69 |
| Opis dołączonej płyty CD..... | 71 |

1 Wprowadzenie

Fałszowanie zdjęć jest pojęciem istniejącym od ponad 150 lat. Wraz z upływem czasu metody edycji zdjęć uległy zmianie. W dzisiejszych czasach zaawansowane edytory graficzne pozwalają na wprowadzenie zaawansowanych zmian na obraz cyfrowy. Część tych zmian może być trudna do zauważenia, a czasami w ogóle niewidoczna, zależnie od umiejętności edytującego.

Są sytuacje, kiedy istnieje konieczność zbadania, czy zdjęcie jest oryginalne. Zdjęcia są często wykorzystywane jako dowody rzeczowe w sprawach sądowych, w policji, oraz w służbach bezpieczeństwa. Media często korzystają ze zdjęć, które wcześniej zostały zedytowane w sposób zmieniający kontekst obrazu, zatajający informacje, lub nadający im nowe znaczenie. Może to skutkować oszukaniem obserwujących, wprowadzaniem ich w błąd oraz ukazaniem jakiejś osoby w złym świetle. Są też sytuacje, gdy podejrzewamy, że udostępnione przez nas zdjęcia mogły ulec edycji, ale nie jesteśmy tego pewni. Wtedy możemy skorzystać z narzędzi, których zadaniem jest weryfikacja, czy badany obraz rzeczywiście uległ edycji.

Autor pracy zdecydował się nad pracą w tym zakresie, czego wynikiem jest program umożliwiający zabezpieczanie obrazów JPG, tak aby po ich edycji można było wskazać miejsca zmiany w obrazie. Oprogramowanie wykorzystuje informacje exif, tworzy zależności między pikselami obrazu, rysuje histogramy stworzone na podstawie zawartości obrazu a także pozwala na zabezpieczenie oryginalnej wielkości obrazu.

2 Podstawowe informacje

Ważnym krokiem w zrozumieniu działania aplikacji jest zapoznanie się z pojęciem fałszerstwa cyfrowego. Stworzony program został poświęcony pracy nad obrazami pochodząymi ze zdjęć wykonanych za pomocą aparatów cyfrowych, zapisanych w plikach JPG. W programie badana jest zawartość pól metadanych exif. Użytkownik aplikacji powinien również wiedzieć jakie informacje można wywnioskować patrząc na histogram obrazu. Bardzo istotnym jest, aby czytelnik zapoznał się z działaniem tych technologii. Autor tej pracy postanowił wytłumaczyć, na czym one polegają a następnie omówić podział metod zabezpieczania obrazu ze względu na niektóre ich cechy.

2.1 Fałszerstwo cyfrowe

Fałszerstwo cyfrowe jest to wprowadzanie zmian na obiektach postaci cyfrowej, mające na celu ukrycie jakichś informacji lub zmylenie potencjalnych użytkowników. Ocena, czy obraz został sfałszowany może być subiektywna, zależna od osoby, która podejmuje taką decyzję. To osoba znajdująca zmiany w obrazie decyduje, co według niej jest fałszerstwem a co jest zmianą, która ma mały wpływ na informacje zawarte w obrazie.

2.2 Kompresja JPEG

Kompresja JPEG jest to metoda kompresji obrazów rastrowych, przeznaczona głównie do stronnego zapisu obrazów naturalnych, charakteryzujących się płynnymi przejściami barw oraz brakiem lub małą ilością ostrzych krawędzi i drobnych detali. Dzięki kompresji JPEG możemy sprawić, że obrazy będą zajmowały mniej miejsca pamięci. Odbywa się to kosztem jakości obrazu. Im mocniejszą kompresję wykorzystamy, tym obraz będzie zajmował mniej pamięci i tym mniej czytelny będzie obraz.

Źródłowy obraz, który chcemy skompresować jest zapisany w postaci RGB24, co oznacza, że na zapis wartości każdej z barw przeznaczone jest po 8 bitów. Pierwszym krokiem jest wykonanie przejścia z modelu barw RGB na model YCbCr (YUV).

Wartość Y odpowiada za jasność obrazu. Wartości Cb i Cr mówią o 2 kanałach chrominancji. Po wykonaniu przejścia z modelu barw RGB na model YCbCr, wykonywany jest podział obrazu na bloki o rozmiarze 8x8 pikseli.

Jeśli wymiary obrazu w poziomie lub pionie nie są wielokrotnością liczby 8 to ostatnie bloki obrazu są zapełniane odpowiednio kopiami wartościami z ostatniej kolumny lub wiersza aż nowa liczba wierszy i kolumn będzie podzielna przez 8. Następnym krokiem jest wykonanie dyskretnej transformaty kosinusowej. Kolejnym krokiem kompresji JPEG jest kwantyzacja obrazu, czyli odrzucenie części pikseli kanałów barwy. Wykonując tę operację wykorzystuje się fakt, że ludzkie oko jest bardziej wrażliwe na niskie niż wysokie częstotliwości. Do wykonania kwantyzacji obrazu można posłużyć się tablicami kwantyzacyjnymi. Dla każdego obrazu JPEG istnieje odpowiednia tablica kwantyzacyjna która zawiera w sobie liczby, przez które należy podzielić odpowiednie wartości wynikowe dyskretnej transformacji kosinusowej. Efektem wykonania kwantyzacji jest zamiana danych zmiennoprzecinkowych na liczby w postaci całkowitoliczbowej. W procesie kwantyzacji obrazu tracona jest część danych o obrazie. Ostatnimi etapami kompresji JPEG jest wykonanie uporządkowania Zig-zag dla każdego z otrzymanych skwantyzowanych bloków obrazu, kodowanie długości ciągów RLE i kodowanie algorymem Huffmana.

Dzięki zastosowaniu porządkowania Zig-zag otrzymane wartości możemy przechowywać w postaci wektora. Dzięki wykorzystaniu kodowania RLE otrzymujemy słowa kodowe dla serii takich samych symboli. Po zastosowaniu kodowania Huffmana otrzymujemy skompresowany plik.

Więcej o komprekcji JPEG można dowiedzieć się korzystając z [9].

2.3 Metadane exif

Metadane exif to standard metadanych dla plików z obrazami, wydany przez JEITA (Japan Electronic and Information Technology Industries Association). Pola exif przechowują wybrane informacje o obrazie. Zawartość pól może zostać nadpisana lub zmieniona za pomocą niektórych edytorów graficznych, więc nie może to być główne zabezpieczenie obrazu. Dane przechowywane w polach exif mogą okazać się pomocne dla osoby badającej obraz. Informacje zawarte w obrazie mogą obejmować datę i czas ostatniej edycji obrazu, program wykorzystany do tego celu, informacje o okolicznościach w których powstał obraz, czy wielkość obrazu licząc w pikselach. Na rysunku 2 widoczne są przykładowe zawartości pól metadanych EXIF.

```
EXIF tags in 'mazda.1.jpg' ('Intel' byte order):
-----
Tag          |Value
-----+-----+
Image Width   |2816
Image Length  |2112
Bits per Sample|8, 8, 8
Photometric Interpretation|RGB
Image Description |OLYMPUS DIGITAL CAMERA
Manufacturer    |OLYMPUS IMAGING CORP.
Model          |FE180/X745
Orientation     |Top-left
Samples per Pixel|3
X-Resolution   |72.0000
Y-Resolution   |72.0000
Resolution Unit|Inch
Software        |Adobe Photoshop CS5 Windows
Date and Time   |2012:01:24 17:03:35
YCbCr Positioning|Co-sited
PRINT Image Matching|690 bytes undefined data
Compression     |JPEG compression
X-Resolution   |72
Y-Resolution   |72
Resolution Unit|Inch
Exposure Time  |1/250 sec.
F-Number        |f/3.1
Exposure Program|Creative program (biased toward depth of field)
ISO Speed Ratings|349
Exif Version    |Exif Version 2.21
Date and Time (Original)|2011:11:21 15:28:04
Components Configuration|Y Cb Cr -
Compressed Bits per Pixel|5
Shutter Speed   |7.97 EV (1/249 sec.)
Aperture         |3.26 EV (f/3.1)
Exposure Bias   |0.00 EV
Maximum Aperture Value|3.26 EV (f/3.1)
Metering Mode   |Pattern
Light Source     |Unknown
Flash            |Flash did not fire, compulsory flash mode
Focal Length    |6.3 mm
User Comment     |
FlashPixVersion|FlashPix Version 1.0
Color Space      |sRGB
Pixel X Dimension|1408
Pixel Y Dimension|1056
File Source      |DSC
Scene Type       |Directly photographed
Custom Rendered  |Normal process
Exposure Mode    |Auto exposure
White Balance    |Auto white balance
Digital Zoom Ratio|0.00
Scene Capture Type|Standard
Gain Control     |High gain up
Contrast          |Normal
Saturation        |Normal
Sharpness          |Normal
Interoperability Ind|R98
Interoperability Ver|0100
-----
EXIF data contains a thumbnail (5927 bytes).
```

Rys. 1: Przykład pól exif wraz z ich zawartością

Więcej informacji o metadanych exif na stronach [5] i [7].

2.4 Histogram obrazu

Histogram obrazu to jeden z graficznych sposobów przedstawiania rozkładu danych numerycznych dotyczących jakiejś jego cechy. Histogram może składać się z szeregu prostokątów umieszczonych na osi współrzędnych. Nie muszą to być prostokąty. Histogram równie dobrze może być stworzony za pomocą sekwencji punktów lub odcinków. Dane numeryczne, na podstawie których rysowany jest histogram obrazu wyznaczane są poprzez zliczenie występowania pikseli o konkretnych wartościach barwy. Przykładem może być histogram obrazu oparty na liczebności 256 grup szarości. Autor pracy przyjął rozwiązanie polegające na podziale pikseli na 153 grupy. Pierwszym krokiem przy tworzeniu histogramu obrazu na podstawie danych numerycznych jest wyznaczenie długości i wysokości całego histogramu a także poziomej odległości pomiędzy dwiema sąsiadującymi reprezentacjami danych numerycznych. Następnie dla zadanych danych numerycznych wyznaczamy „najwyższą” wartość. Gdy już to zrobimy, wiemy jakiej wysokości powinien być prostokąt lub na jakiej wysokości powinniśmy umieścić punkt albo odcinek opisujący daną wartość liczbową. Osoba badająca obraz na podstawie histogramu jakiejś jego cechy może uzyskać o nim wiele cennych informacji.



Rys. 2: Wpływ zmiany jasności obrazu na jego histogram szarości [1]



Rys. 3: Wpływ zmiany kontrastu obrazu na jego histogram szarości [1]

Na rysunku 2 widać wpływ zmiany jasności obrazu na jego histogram szarości. Po lewej stronie widoczny jest obraz oryginalny i odpowiadający mu histogram szarości. Po prawej znajduje się obraz, który uległ edycji i jego histogram szarości. Obserwator jest w stanie dostrzec, że dla dużej liczby grup szarości reprezentujących piksele o niskiej wartości barw RGB liczebność grupy wynosi 0, co może być powodem do uznania obrazu za edytowany.

Na rysunku 3 widoczny jest wpływ zmiany kontrastu obrazu na histogram szarości. Podobnie jak to było w przypadku rysunku 2 „widoczne są” grupy szarości, dla których liczebność pikseli przynależących wynosi 0. Taki stan histogramu szarości również może świadczyć o edycji obrazu.

Więcej informacji o histogramach obrazu na [1], [6] i [10].

2.5 Podział metod wykrywania fałszerstw ze względu na działanie

Obecnie znanych jest bardzo wiele metod wykrywania zmian w obrazach cyfrowych. Wielu producentów oprogramowania zajmującego się tą dziedziną prowadzi badania, których celem jest opracowanie nowych, nieznanych do tej pory metod wykrywania fałszerstw, dzięki którym ich oprogramowanie będzie bardziej efektywne. Metody wykrywania fałszerstw w obrazach cyfrowych można podzielić na kilka kategorii biorąc pod uwagę ich sposób działania.

2.5.1 Metody statyczne

Metody statyczne, są to metody, które nie wymagają dostępu do obrazu przed jego edycją. Próbowią znaleźć podejrzane fragmenty obrazu mając do dyspozycji jedynie obraz który uległ już zmianom. Nie mają dostępu do oryginalnego obrazu.

2.5.2 Metody dynamiczne

Metody dynamiczne, są to metody wykrywania fałszerstw cyfrowych, polegające na wprowadzeniu zabezpieczeń na obraz oryginalny i późniejszej weryfikacji zabezpieczonych obrazów. Zabezpieczenie obrazu może polegać na wprowadzeniu zmian na obrazie oryginalnym, lub zapisaniu informacji uzyskanych dzięki analizie obrazu oryginalnego. Weryfikacja obrazu może polegać na sprawdzeniu, czy wprowadzone zmiany nadal są obecne lub czy zachowane dane nadal są zgodne z obrazem.

2.5.3 Metody steganograficzne

Metody steganograficzne są podgrupą metod dynamicznych wykrywania fałszerstw [3]. Polegają one na prowadzeniu zmian w zabezpieczanym obrazie w sposób niewidoczny dla obserwatorów. Niektórzy uważają, że steganograficzne metody wykrywania fałszerstw mają przewagę nad metodami kryptograficznymi

wykrywania fałszerstw. Dzięki trudnej wykrywalności zabezpieczenia sprawiamy, że informacje, które chcemy ukryć nie będą podlegały analizie przez obserwatorów.

2.5.4 Metody kryptograficzne

Metody kryptograficzne podobnie jak metody steganograficzne są podgrupą metod dynamicznych wykrywania fałszerstw w obrazach cyfrowych [2]. Ich działanie także opiera się na wprowadzeniu zmian w obrazie. Nie polegają one na ukrywaniu zabezpieczenia, ale polegają na tym by widoczne zabezpieczenie nie było zrozumiałe dla obserwatora. Narzędzia kryptograficzne służą do zmiany postaci zabezpieczanych informacji (szyfrowanie) w sposób możliwy do odwrócenia (deszyfrowanie).

2.6 Najpopularniejsze metody wykrywania zmian w obrazach cyfrowych

2.6.1 Metody działające na pikselach obrazu

Programy przeznaczone do wykrywania zmian w obrazach cyfrowych bardzo często polegają na analizie sąsiedztwa pikseli na obszarze całego obrazu. Istnieją zabezpieczenia, których działanie polega na uzależnieniu poszczególnych pikseli obrazu od jakichś parametrów. Przykładem tego typu zabezpieczenia jest nakładanie znaków wodnych na obraz.

Kopiowanie i umieszczanie kopii części obrazu w innym miejscu jest jedną z bardziej popularnych metod edycji obrazu. Często zmiany tego typu są widoczne gołym okiem, ale nie zawsze. Dlatego powstały metody polegające na wyszukiwaniu identycznych fragmentów obrazu. Przykładowa metoda tego typu polega na wytworzeniu bloków pikseli tej samej wielkości. Następnie dla każdego z otrzymanych zestawów pikseli możemy wyliczyć wartość według określonej funkcji. Kolejnym krokiem może być posortowanie wszystkich bloków według ich wartości dla funkcji. Jeśli dla dwóch bloków wartości funkcji są identyczne, to może to oznaczać, że jeden fragment jest kopią drugiego.

Jeśli fałszerz zdjęcia chciałby umieścić w obrazie fragment innego obrazu to musiałby ten fragment uprzednio przygotować. Można go powiększyć lub zmniejszyć tak, aby wizualnie pasował do reszty obrazu i nie wyróżniał się.

Wykonanie takich zmian sprawia, że na edytowany fragment narzucone są właściwości, które nie występują w obrazach naturalnych, co może wskazywać na możliwość edycji obrazu.

Badanie autentyczności obrazu może wymagać zebrania odpowiedniej liczby zdjęć wykonanych za pomocą aparatu tej samej marki i modelu. Zebrane zdjęcia mogą posłużyć do badań polegających na zbadaniu prawdopodobieństwa występowania pikseli o różnych barwach RGB w sąsiedztwie ze sobą. Uzyskane w ten sposób dane mogą posłużyć do weryfikacji oryginalności zdjęcia wykonanego danym modelem aparatu. Im większy jest zbiór zdjęć wykorzystany do zebrania danych o możliwości sąsiedztwa pikseli, tym badanie będzie bardziej godne zaufania. Wynikiem badania nie musi być stwierdzenie, że obraz jest prawdziwy lub nie. Rezultatem może być wynik procentowy. Im więcej znaleziono sąsiedztw pikseli niepasujących do przeprowadzonych badań, tym większa szansa, że obraz edytowano.

2.6.2 Metody opierające swoje działanie na formacie zapisu

Ta grupa metod skupia się na badaniach obrazów, które uległy kompresji JPEG. Kompresja ta nakłada na obraz właściwości, których zakłócenie może być oznaką prób edycji obrazu. Do metod opartych na formacie zapisu zalicza się metody badające kwantyzację wykonaną w trakcie kompresji JPEG, metody badające skutki wielokrotnej kompresji JPEG obrazu oraz metody skupiające się na badaniu podziałów obrazu na bloki.

Istnieje możliwość zbadania, jakie urządzenie posłużyło do stworzenia badanego obrazu. Do tego celu należy wykorzystać tabele kwantyzacji obrazu, które zostały wykorzystane do wygenerowania obrazu. Tabele kwantyzacji można pozyskać bezpośrednio z obrazu lub mogą zostać określone na podstawie tego obrazu. Więcej o tabelach kwantyzacji obrazu można dowiedzieć się czytając artykuł [11].

Zakładając, że obraz edytowany musiał ulec kompresji JPEG co najmniej dwa razy, możemy go zbadać w poszukiwaniu znaków szczególnych, charakterystycznych dla obrazów, na których przeprowadzono wielokrotną kompresję. Przykładem takiego znaku może być wygląd histogramu obrazu. Jeśli uda nam się wykryć cechę obrazu, świadczącą o użyciu wielokrotnej kompresji, to możemy wtedy przypuścić, że obraz mógł zostać sfałszowany. Sam fakt istnienia takich cech nie daje jednak pewności, że obraz został zedytowany. Ich występowanie może być skutkiem wczytania i zapisania obrazu w edytorze graficznym.

W trakcie kompresji JPEG kompresowany obraz dzielony jest na równej wielkości bloki. Operacja kompresji obrazu wykonywana jest na każdym bloku oddzielnie. Powoduje to, że na krawędziach poszczególnych bloków mogą pojawić się artefakty. Edycja obrazu, może spowodować, że zostaną one naruszone, dzięki czemu będziemy w stanie stwierdzić które bloki mogły ulec edycji. Przykładami zmian w obrazie powodującymi naruszenie artefaktów nałożonych na krawędziach bloków jest dorysowanie lub wklejenie fragmentu obcego. Jeśli edytowany obszar będzie przekraczać granice jednego bloku, to część artefaktów zniknie.

2.6.3 Metody opierające swe działanie na właściwościach fizycznych

Istnieją metody wykrywania zmian w obrazach opierające swoje działanie na badaniu właściwości fizycznych dla obiektów widocznych na obrazie. Możemy sprawdzić, czy ta sama zależność fizyczna jest zgodna dla wszystkich obiektów widocznych na zdjęciu.

Najczęściej badaną właściwością fizyczną obrazu jest oświetlenie obiektów. Metody te polegają na znalezieniu i ustaleniu położenia źródeł światła mających wpływ na obraz a następnie zweryfikowaniu, czy te same źródła światła w odpowiedni sposób wpływają na wszystkie obiekty widoczne na obrazie.

Więcej o metodach wykrywania zmian w obrazie można dowiedzieć się czytając artykuł [4].

3 Koncepcja rozwiązania

Autor tej pracy zdecydował się na stworzenie aplikacji komputerowej pozwalającej wykryć niektóre typy zmian w obrazie JPG. Zdecydowano, że program powinien umożliwić pozyskanie informacji o obrazie, jego zabezpieczenie, oraz późniejszą weryfikację autentyczności obrazu.

Przed rozpoczęciem procesu tworzenia aplikacji zgromadzono informacje o popularnych rodzajach zabezpieczeń, wykorzystywanych w nowoczesnym oprogramowaniu stworzonym do tego typu celów. W trakcie prac nad programem autor pracy wymyślał kolejne techniki radzenia sobie z problemami wykrywania zmian w obrazie, które następnie były akceptowane przez promotora pracy i wprowadzane do programu.

Autor zdecydował się napisać program w języku programowania C++, oraz skorzystać z biblioteki exiv2 oraz biblioteki QT. Dzięki wykorzystaniu biblioteki exiv2 umożliwiono sprawdzanie zawartości pól metadanych EXIF a także pozwoliło to na stworzenie dodatkowych zabezpieczeń, wykorzystując część tych pól. Biblioteka QT natomiast została wykorzystana do stworzenia interfejsu oraz jest wykorzystywana do nakładania zabezpieczeń i późniejszej weryfikacji obrazu. Więcej o użytych bibliotekach można dowiedzieć się przeglądając strony internetowe [7] i [8].

3.1 Wymagania funkcjonalne

- Aplikacja będzie w stanie obsługiwać pliki z obrazami zapisanymi z rozszerzeniem JPG,
- Aplikacja będzie przedstawiała histogramy, wyliczone na podstawie zadанego obrazu,
- Aplikacja będzie przedstawiała metadane, znalezione w zadanym obrazie,
- Aplikacja pozwoli na stworzenie nowego pliku z zabezpieczonym obrazem,
- Aplikacja pozwali na zbadanie obrazu w celu znalezienia ewentualnych śladów jego edycji,
- Program będzie przedstawiać raport stworzony na podstawie przebiegu procesu weryfikacji obrazu,
- Program będzie przedstawiać różnicę histogramów sum barw opartej na aktualnym stanie obrazu i stanie historycznym.

3.2 Wymagania niefunkcjonalne

- Aplikacja zostanie napisana w języku programowania C++ z wykorzystaniem bibliotek EXIV2 oraz QT w środowisku programistycznym Eclipse,
- Program będzie łatwy w obsłudze, a interfejs użytkownika będzie intuicyjny.

3.3 Plan realizacji wymagań

Korzystając z biblioteki QT będzie można stworzyć wygodny w obsłudze interfejs graficzny użytkownika programu. Dodatkowo biblioteka QT umożliwia wykonywanie działać na obrazach, co sprawia, że można ją wykorzystać w części programu odpowiedzialnej za zabezpieczanie obrazu i późniejszą jego analizę. Dzięki zastosowaniu klasy QFileDialog umożliwiony zostanie wybór pliku obrazu, nad którym chcemy działać. Klasa QT umożliwia także wykonywanie operacji rysowania na obrazie, dzięki czemu będziemy w stanie wyznaczyć liczebność poszczególnych grup a następnie narysować odpowiednie histogramy sum barw i barw RGB dla wybranego obrazu. Biblioteka EXIV2 służy do przeglądania i edycji zawartości metadanych zamieszczonych w obrazie JPG, dzięki czemu tworzony program będzie w stanie wyświetlić zawartość metadanych exif. Klasa QT daje możliwość działania na plikach, co umożliwi stworzenie nowego pliku i zamieszczenia w nim zabezpieczonego obrazu. Stworzone przez autora metody wykrywania fałszerstw zostaną zaimplementowane w programie, co pozwoli na wykrycie zmian w zabezpieczonym obrazie.

Metody wykrywania zmian w obrazie, które zostaną wykorzystane w ramach projektu to:

- Metoda wykorzystania pól metadanych exif,
- Analiza histogramów obrazu,
- Tworzenie zależności między pikselami,
- Tworzenie znaku zabezpieczającego,
- Wykrywanie rotacji obrazu,
- Tworzenie zabezpieczenia wielkości obrazu,
- W trakcie analizy obrazu przygotowywany będzie raport z wynikami poszczególnych etapów weryfikacji obrazu.

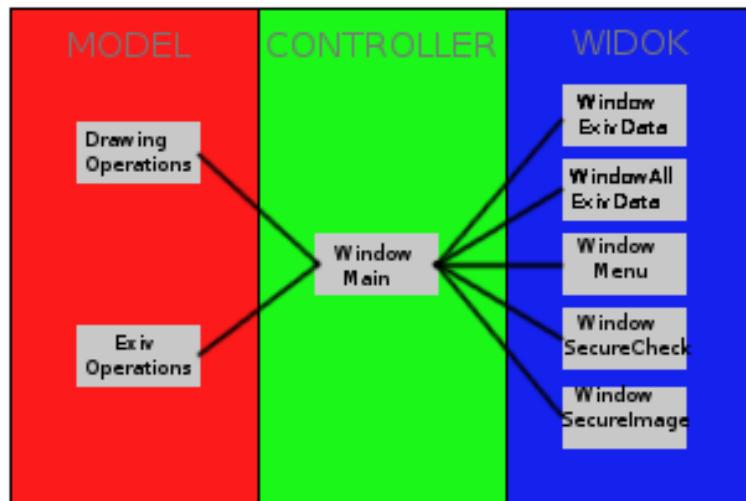
3.4 Założenia związane z korzystaniem ze stworzonego programu

Siła oprogramowania służącego do wykrywania fałszerstw w obrazach cyfrowych jest oparta na utrzymywaniu w tajemnicy użytych metod i ograniczeniu do niego dostępu. Im więcej ludzi wie na czym polegają algorytmy wykrywania fałszerstw, tym większa jest szansa, że znajdzie się osoba będąca w stanie zedytować obraz tak, aby zmiany nie zostały wykryte. Ograniczanie dostępu do oprogramowania przyczynia się do utrzymywania w tajemnicy sposobu jego działania. Część metod zabezpieczania i późniejszej weryfikacji obrazu wykorzystanych w stworzonym w ramach tego projektu oprogramowaniu jest prosta do ominięcia, jeśli znane jest znaczenie tych zabezpieczeń. Program mógłby być udostępniany dla osób należących do konkretnej instytucji albo organizacja posiadająca program mogłaby świadczyć usługi związane z zabezpieczaniem i weryfikacją autentyczności obrazów cyfrowych. Dodatkowym powodem, dla którego program nie powinien być rozpowszechniany jest możliwość wykonania zabezpieczenia obrazu na już wcześniej zabezpieczonym i zedytowanym obrazie.

Nie każda edycja obrazu musi być utożsamiana z fałszerstwem cyfrowym. Jest możliwe wprowadzenie zmian, które nie zmieniają kontekstu obrazu i nie mają na celu zmylenia obserwatorów. Dlatego to osoba weryfikująca obraz powinna zadecydować, czy obraz uległ fałszerstwu. Zależnie od sytuacji w niektórych przypadkach najmniejsza edycja zdjęcia może sprawić, że obraz nie będzie wiarygodny. Są też sytuacje, gdzie obserwator może uznać niektóre rodzaje edycji za akceptowalne.

4 Struktura aplikacji

Stworzony program składa się z 8 klas. Aplikacja została stworzona w oparciu o wzorzec MVC. Na rysunku 4 widoczny jest schemat podziału klas na poszczególne moduły.



Rys. 4: Schemat podziału klas na moduły wzorca MVC

DrawingOperations – Klasa ta odpowiada za realizację wprowadzenia zabezpieczeń w treść obrazu a także ich weryfikację. W klasie tej realizowane są także operacje mające na celu wyznaczenie liczebności poszczególnych grup sum barw i barw RGB a następnie stworzenie odpowiednich histogramów obrazu. W trakcie analizy obrazu tworzona jest w niej część raportu dotycząca rezultatów weryfikacji obrazu pod kątem wprowadzonych wcześniej zabezpieczeń.

ExivOperations – Dzięki tej klasie możliwe jest wydobycie metadanych EXIF z obrazu. Dodatkowo jest ona wykorzystywana do wprowadzania zabezpieczenia obrazu opartego na wykorzystaniu metadanych EXIF. Podczas analizy obrazu wytwarzany jest tu fragment raportu opisującego przebieg weryfikacji obrazu.

WindowAllExivData – Jest to klasa służąca do pokazania użytkownikom zawartości wszystkich pól metadanych EXIF w obrazie.

WindowExivData – Klasa wykorzystywana do przedstawienia najbardziej istotnych metadanych w kontekście analizy autentyczności obrazu a także histogramów sum barw i barw RGB.

WindowMain – Jest to klasa reagująca na większość zdarzeń zachodzących podczas użytkowania programu. Kontroluje przebiegiem działań i odpowiada za zmianę widoków aplikacji.

WindowMenu – Jest to klasa przedstawiająca menu główne programu. Pozwala wybrać plik obrazu, który chcemy przeanalizować lub zabezpieczyć. Umożliwia przejście do innych widoków programu tylko jeśli wybrany zostanie odpowiedni plik z obrazem.

WindowSecureCheck – Klasa ta służy do wizualizacji wyników sprawdzenia autentyczności obrazu. Przedstawiony jest tam obraz z zaznaczonymi miejscami możliwej edycji. Wykres porównujący histogramy sum barw obrazu i raport tworzony w trakcie weryfikacji obrazu. Umożliwia powtórzenie badań obrazu pod względem zabezpieczenia wprowadzanych zależności między pikselami dla ustalonej wartości dokładności weryfikacji.

WindowSecureImage – Jest to klasa przedstawiająca okno zabezpieczania obrazu. Umożliwia wybór lokalizacji i nazwy pliku zabezpieczonego.

5 Zabezpieczenia wykorzystane w programie

Autor oprogramowania chciał, aby wszystkie zabezpieczenia jednocześnie były mało skomplikowane oraz by umożliwiały wykrywanie jak największej liczby zmian w obrazie.

Właśnie dlatego zdecydowano się na stworzenie własnych zabezpieczeń. Wraz z rozwojem projektu dodawana była implementacja nowych zabezpieczeń, które dawały możliwość radzenia sobie z nowymi problemami.

5.1 Wykorzystanie pól metadanych EXIF

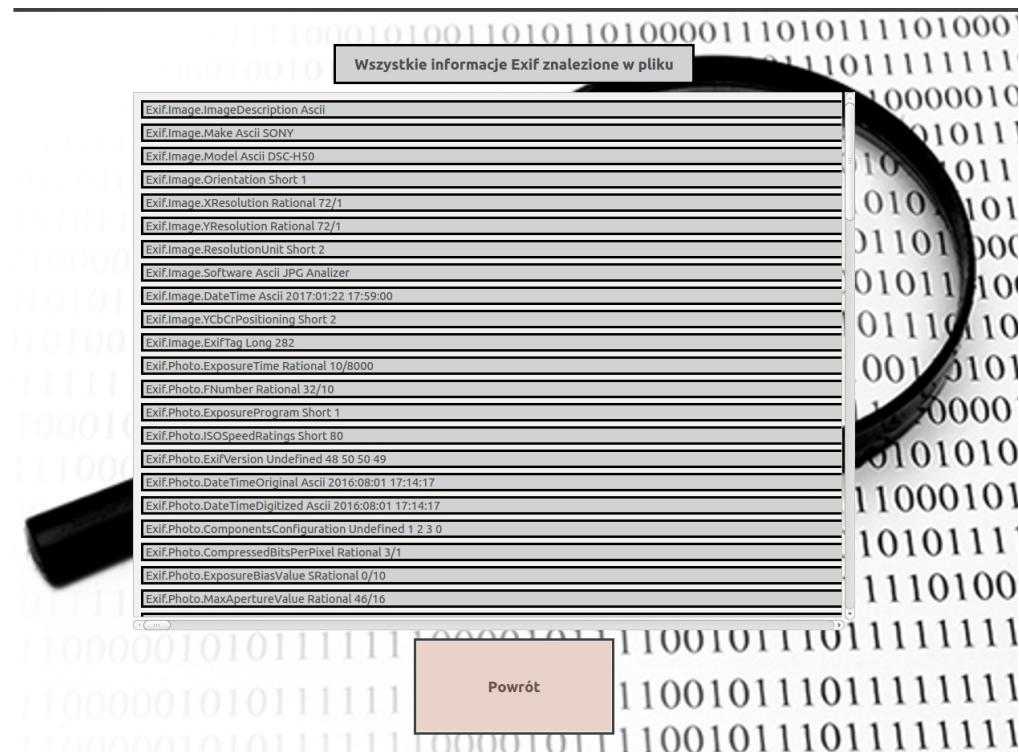
Program umożliwia sprawdzenie zawartości pól metadanych EXIF w wybranym obrazie. Na rysunku 5 przedstawiona została zawartość wybranych pól EXIF. Przedstawione dane zostały uznane za najistotniejsze w kontekście weryfikacji autentyczności obrazu cyfrowego. Pola Exif.Image.Make i Exif.Image.Model przechowują informacje o marce i modelu aparatu wykorzystanego do wygenerowania zdjęcia. Exif.Image.Software mówi jaki program graficzny został użyty do ostatniej edycji obrazu. Exif.Image.DateTime przechowuje datę i czas ostatniej edycji obrazu. Exif.Photo.ExifVersion zawiera informację o wersji EXIF z którą zgodne są przechowywane metadane. Metadane Exif.Photo.DateTimeOriginal i Exif.Photo.DateTimeDigitized mówią o czasie wytworzenia zdjęcia i czasie kiedy wygenerowano plik cyfrowy przechowujący to zdjęcie. Pole Exif.Photo.UserComment jest polem przechowującym informacje użytkownika o obrazie. Pola Exif.Photo.PixelXDimension i Exif.Photo.PixelYDimension zawierają wielkość obrazu. Są to odpowiednio liczba pikseli w poziomie i liczba pikseli w pionie.

Jest też możliwość obejrzenia wszystkich metadanych EXIF zawartych w pliku badanego obrazu. Zawartość wszystkich pól metadanych EXIF jest pokazana na rysunku 6.

Zmiana zawartości pól EXIF jest jednym z kroków zabezpieczenia obrazu. Zabezpieczenie oparte na analizie pól EXIF nie powinno być główną metodą zabezpieczania obrazu w żadnym oprogramowaniu, ze względu na łatwość zmiany ich zawartości. Istnieje wiele programów umożliwiających zarówno przeglądanie jak i edycję wszystkich pól metadanych EXIF. Osoba znajdująca się na programowaniu jest w stanie szybko stworzyć własny program umożliwiający przeglądanie i edycję wszystkich pól metadanych EXIF.

| |
|--|
| Exif.Image.Make Ascii SONY |
| Exif.Image.Model Ascii DSC-H50 |
| Exif.Image.Software Ascii JPG Analyzer |
| Exif.Image.DateTime Ascii 2017:01:22 17:59:00 |
| Exif.Photo.ExifVersion Undefined 48 50 50 49 |
| Exif.Photo.DateTimeOriginal Ascii 2016:08:01 17:14:17 |
| Exif.Photo.DateTimeDigitized Ascii 2016:08:01 17:14:17 |
| Exif.Photo.UserComment Undefined I;9KH;;)*+, (+/((&'-0&'0(('-0+/0&& *.&)&*(('+&) *(.'+ |
| Exif.Photo.PixelXDimension Long 3456 |
| Exif.Photo.PixelYDimension Long 2592 |

Rys. 5: Fragment widoku programu przedstawiający zawartość wybranych pól EXIF



Rys. 6: Widok programu przedstawiający zawartość wszystkich pól EXIF znalezionych w obrazie

W etapie zabezpieczania obrazu edytowane są zawartości pól EXIF:

- Pole nazwy programu, ostatnio edytującego obraz,
- Pole daty ostatniej zmiany obrazu,
- Pole komentarza użytkownika.

Pole nazwy programu zmieniane jest na JPGAnalyzer, czyli na nazwę programu stworzonego w ramach tego projektu. W polu daty ostatniej zmiany obrazu wpisujemy aktualną datę i czas. Pole komentarza jest miejscem, gdzie wpisujemy informacje o aktualnym rozmiarze obrazu, dacie i czasie zabezpieczenia obrazu, oraz dane wymagane do odtworzenia histogramu sum barw zabezpieczanego obrazu. Edytor graficzny użyty do zmiany obrazu w trakcie zapisu może zmienić zawartość pola nazwy ostatniego oprogramowania użytego do zmiany obrazu, może także nadpisać datę i czas ostatniej zmiany dokonanej na obrazie. Domyślnie oprogramowanie takie nie wprowadza natomiast zmian w polu komentarza użytkownika. Dzięki temu w trakcie weryfikacji możemy sprawdzić, czy zarówno nazwa programu, data i czas ostatniej zmiany są zgodne z informacjami zapisanymi w polu komentarza użytkownika.

Dane zapisywane do pola komentarza użytkownika (Exif.Photo.UserComment) ulegają szyfrowaniu. Jest to proste szyfrowanie polegające na dodaniu 10 do wartości Ascii dla każdego znaku prócz znaków spacji. Deszyfrowanie polega na odjęciu 10 od wartości Ascii dla odczytanych znaków. Na rysunku 5 widoczna jest zawartość pola komentarza użytkownika dla zabezpieczonego obrazu cyfrowego.

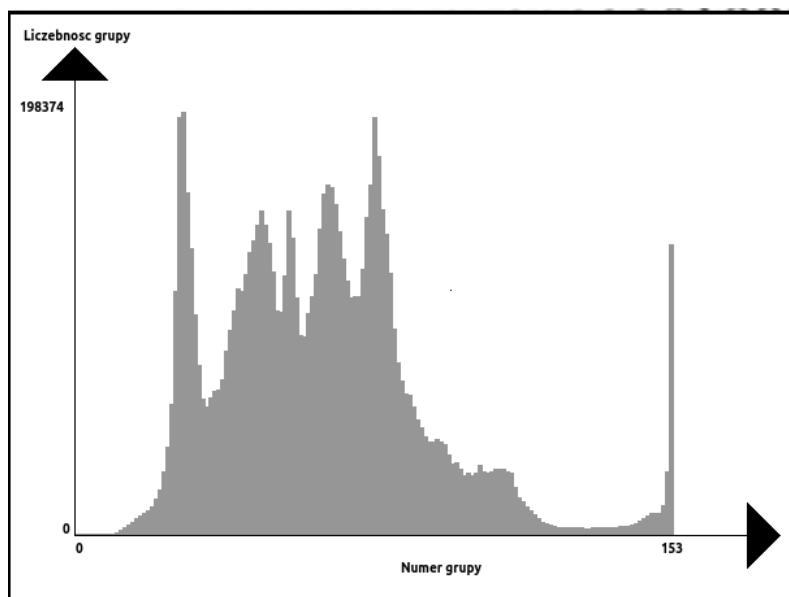
Pierwszym słowem występującym w ciągu znaków, który chcemy zapisać do pola komentarza użytkownika jest słowo SECURED, które jest słowem oznaczającym, że kolejne słowa będą stanowić informacje dotyczące obrazu. Słowo to po dodaniu 10 do każdego znaku uzyskuje postać „I;9KH;:”. Następnym krokiem jest zapisanie aktualnej wielkości obrazu. W przypadku przedstawionym na rysunku 5 są to odpowiednio 3456 i 2592. Po zastosowaniu szyfrowania uzyskujemy „)*+,” i „(+(/”. Po zapisaniu rozmiaru obrazu program zapisuje aktualną datę i czas. Jest to ta sama data i czas, która została zapisana do pola „Exif.Photo.DateTime”. Zaszyfrowana data ma postać „(&-0&'0(('-0+/0&&”. Ostatnim krokiem jest zaszyfrowanie i zapisanie liczebności 153 grup histogramu sum barw obrazu.

Weryfikacja zabezpieczenia EXIF polega na sprawdzeniu, czy nazwa ostatnio użytego programu do edycji obrazu jest zgodna z nazwą naszego programu i sprawdzeniu, czy data i czas zabezpieczenia, znajdujące się w komentarzu użytkownika nadal są zgodne z zawartością pola daty i czasu ostatniej zmiany. Dodatkowo liczebność grup histogramu sum barw posłuży się do stworzenia

diagramu porównującego histogramy sum barw w aktualnym i pierwotnym obrazie, o czym napiszę w następnym podpunkcie.

5.2 Wykorzystanie histogramów obrazu

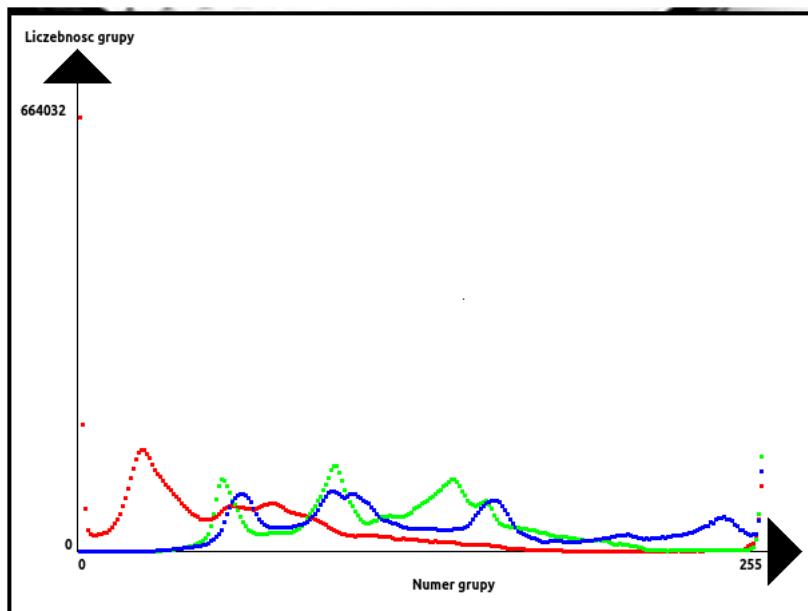
Program jest w stanie tworzyć histogramy dla wybranego obrazu. Histogram sum barw obrazu jest tworzony na podstawie liczb pikseli przydzielonych do 153 grup. Przydział do grup wykonywany jest na podstawie sumy składowych RGB dla pikseli. Histogram jest tworzony poprzez narysowanie 153 prostokątów, których wysokość zależna jest od liczby poszczególnych grup. Bardziej intuicyjnym rozwiązaniem jest użycie 256 grup, jednak ze względu na wielkość zabezpieczenia znakiem zabezpieczającym autor zdecydował się ograniczyć liczbę grup do 153. Opis znaku zabezpieczającego znajduje się w jednym z kolejnych podpunktów. Na rysunku 7 widać przykładowy histogram sum barw stworzony przez program.



Rys. 7: Przykładowy histogram sum barw stworzony przez program

Dodatkowo program może tworzyć histogram barw RGB obrazu. Istnieje 256 grup dla każdej z barw RGB. Dla każdej z 256 wartości barwy rysowany jest punkt, którego położenie pionowe uzależnione jest od liczby grup. Po wyliczeniu i narysowaniu histogramu RGB program przedstawia go użytkownikowi. Nie stworzono algorytmu mającego weryfikować obraz na podstawie histogramu RGB. Zakłada się, że praca nad projektem może być w przyszłości

kontynuowana i wtedy można będzie zaimplementować zabezpieczenie obrazu w oparciu o ten typ histogramu. W ramach pracy inżynierskiej histogram RGB został wprowadzony jako dodatkowe źródło informacji, które użytkownik może wykorzystać do analizy obrazu. Przykładowy histogram barw RGB umieszczono na rysunku 8.

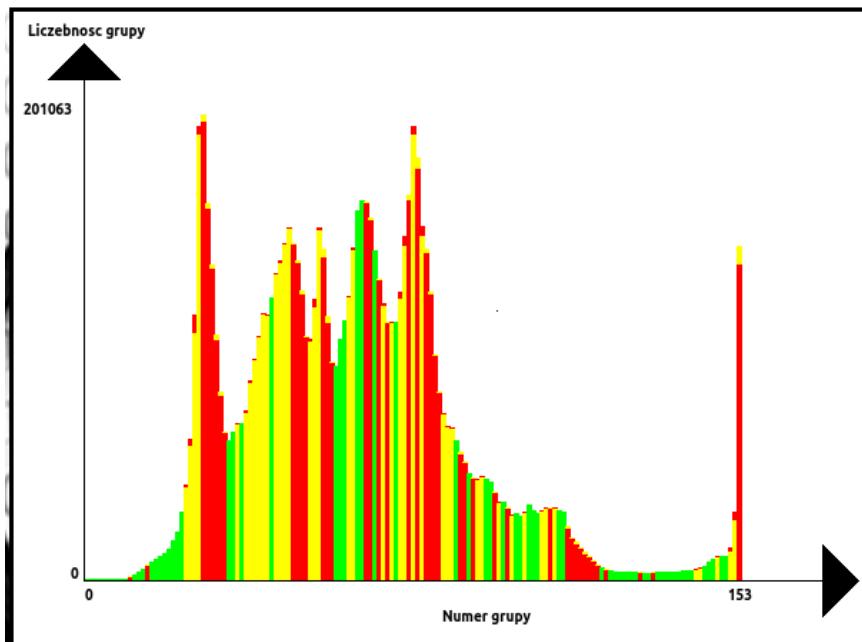


Rys. 8: Przykładowy histogram barw RGB stworzony przez program

W trakcie zabezpieczenia informacja o liczce każdej ze 153 grup sum barw jest zapisywana w dwóch miejscach:

- W polu komentarza użytkownika,
- W znaku zabezpieczającym.

Ostatnim histogramem, tworzonym w aplikacji jest histogram porównujący histogram sum barw oparty na aktualnym stanie obrazu z histogramem sum barw opartym na wartościach wczytanych z komentarza użytkownika lub ze znaku zabezpieczającego. Przykład takiego histogramu pokazano na rysunku 9.



Rys. 9: Histogram przedstawiający różnice histogramów sum barw

Na żółto malowane są prostokąty, które przedstawiają wielkości wczytane z zabezpieczenia. Na czerwono malowane są prostokąty, które przedstawiają wielkości wyliczone na podstawie aktualnego stanu obrazu. Na zielono malowane są prostokąty, dla których obie wartości są sobie równe. W razie braku wymaganych danych w polu metadanych EXIF, są one wczytywane ze znaku zabezpieczającego którego opis znajduje się w jednym z kolejnych podpunktów.

Na rysunku 9 widoczny jest histogram porównujący histogram sum barw pochodzący z aktualnego stanu obrazu z histogramem stworzonym w oparciu o dane dotyczące obrazu zabezpieczanego wyczytane z pola komentarza użytkownika lub znaku zabezpieczającego. Jak można zauważyć dane liczbowe dla wielu grup sum barw nie są identyczne. Jest tak pomimo braku edycji obrazu. Efekt ten jest spowodowany niedoskonałością procesu zapisywania obrazu do pliku. W jego trakcie lekko zmieniane są wartości barwy RGB niektórych pikseli. Autor zauważył, że im większy jest kontrast pomiędzy pikselem a jego sąsiadami, tym większa dla niego będzie zmiana barwy RGB. Opisane tutaj zmiany zachodzą pomimo braku kompresji obrazu. Zmiany barw RGB pikseli powodują, że w trakcie weryfikacji są one przyporządkowywane do innych grup, co jest powodem różnic liczebności grup sum barw. Przy dalszej pracy nad projektem można postarać się

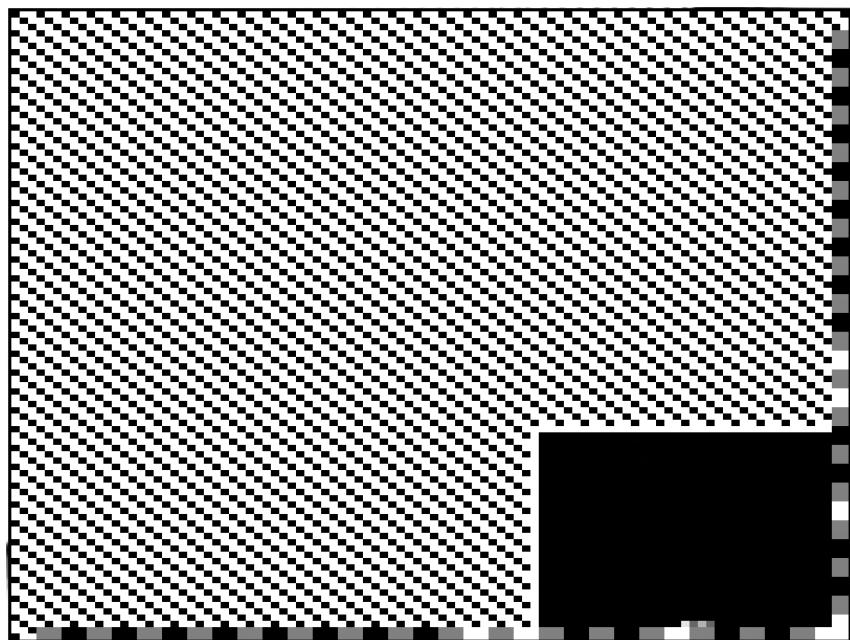
znać metodę na zapis obrazu do pliku bez zmian barw RGB pikseli. Użytkownik aplikacji powinien uwzględnić ten fakt badając obraz. Można założyć, że drobne odchylenia liczebności grup sum barw powinny być tolerowane. Tylko duże różnice wysokości „słupków” powinny być powodem do podejrzewania, że obraz uległ manipulacji.

5.3 Zależności między pikselami

Zabezpieczenia opisane do tej pory nie pozwalają na wskazanie zmienionych fragmentów obrazu, a jedynie mogą dać użytkownikowi powód do podejrzewania, że obraz mógł ulec edycji. Jak zostało to już napisane, zawartość wszystkich pól metadanych można zmienić. Dane te można tak zmienić, aby użytkownik nie zauważał w nich niczego, co mogłoby wskazywać na edycję obrazu. Wykorzystanie szyfrowania zawartości pola komentarza nie rozwiązuje problemu. W najgorszym przypadku wszystkie dane EXIF mogą ulec usunięciu. Usunięcie metadanych z obrazu nie musi być równoważne fałszerstwu.

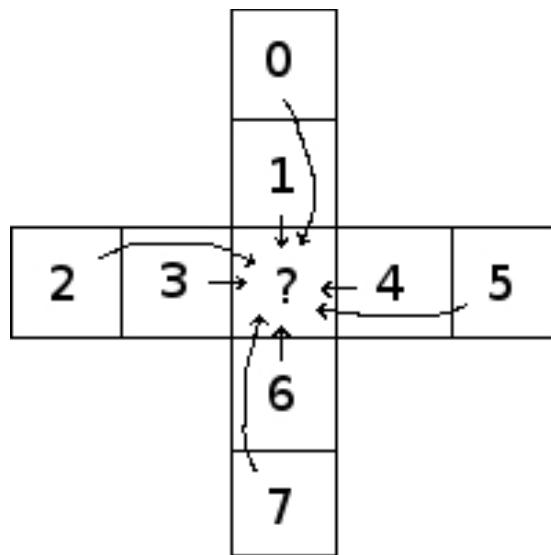
Właśnie dlatego autor programu zdecydował się wprowadzić kolejne zabezpieczenie, które stało się głównym zabezpieczeniem w tworzonym programie. Zabezpieczenie polega na zmianie koloru RGB dla określonych pikseli. Piksele, na których wykonywana jest edycja położone są zgodnie z siatką przedstawioną na rysunku 10. Piksele te przedstawione zostały jako czarne kwadraty ułożone na ukos.

Kolejne zabezpieczane piksele występują w odległości 3 kolumn od siebie. Numer kolumny będącej kolumną startową zabezpieczenia w kolejnych rzędach jest resztą z dzielenia numeru rzędu przez 3. Oznacza to, że dla pierwszej kolumny pierwszy piksel zabezpieczany znajduje się w „zerowym rzędzie”. Kolejne dwa zabezpieczone piksele dla pierwszej kolumny położone są w drugim i piątym rzędzie. Kolejne zabezpieczane piksele oddalone są od siebie o 3. Zabezpieczenie to nie dotyczy obszarów będących częścią znaku zabezpieczającego i zabezpieczeń wielkości obrazu o których dowiemy się w następnych podpunktach.



Rys. 10: Obraz przedstawiający rozmieszczenie pikseli, których kolor jest zmieniany

Nowy kolor piksela uzyskujemy licząc średnią arytmetyczną barw RGB jego sąsiadów. Branych pod uwagę jest maksymalnie osiem pikseli sąsiadujących w sposób przedstawiony na rysunku 11.



Rys. 11: Schemat przedstawiający sąsiadów na podstawie kolorów których wyliczany jest nowy kolor dla piksela

Weryfikacja oryginalności obrazu polega na ponownym przejściu przez zabezpieczone piksele i sprawdzenie, czy zależności pomiędzy nimi a ich sąsiadami nadal jest spełniona. Ze względu na niedoskonałości w metodzie zapisu obrazu i kompresję obrazu do której może dojść podczas otwarcia i nadpisania obrazu zakłada się, że zależność jest spełniona, gdy wartość barw RGB sprawdzanego piksela nie przekracza zadanego odchylenia od wyliczonej średniej barw RGB z sąsiednich pikseli. Zauważono, że już w trakcie zabezpieczania obrazu może dojść do nieplanowanych zmian kolorów pikseli obrazu.

Dzięki wykorzystaniu odchylenia, z jednej strony umożliwiamy wprowadzanie bardzo delikatnych zmian na pojedynczych pikselach. Z drugiej strony znacznie ogranicza to wykrywanie tak zwanych „False positives” czyli obszarów obrazu uznanych jako zmienione, gdy tak naprawdę nie uległy one zmianie. Obszary te mogły zostać zmienione na skutek kompresji wykonanej przy wczytaniu i nadpisaniu obrazu lub nieprzewidywalnych lekkich zmian barw pikseli narzuconych w trakcie zabezpieczania obrazu. Zmiany barw RGB pikseli w trakcie zapisywania zabezpieczonego obrazu opisano w poprzednim podpunkcie.

Gdy znaleziony zostaje piksel, którego barwa przekracza maksymalne akceptowalne odchylenie od prawidłowej wartości wyliczonej na podstawie sąsiadów, w miejscu jego wystąpienia zamalowywany jest obszar wielkości 5 na 5 pikseli na kolor czerwony.

Postanowiono zamalować obszar większy od pojedynczego piksela ze względu na słabą widoczność zamalowania pojedynczego piksela na obrazie którego wielkość może wynosić tysiące pikseli. W przypadku, gdy zamalowywany obszar przekracza granice obrazu, nachodzi na obszar znaku zabezpieczającego lub obszaru zabezpieczenia wielkości oryginalnej obrazu, to obszar ten jest ograniczany tak, aby nie wpływać na te fragmenty obrazu. Po zamalowaniu obszaru wokół niepasującego piksela następuje weryfikacja 4 sąsiadujących pikseli. Sprawdzane jest, czy nie posiadają one sąsiadów o kolorze identycznym z ich kolorem. Jeśli tak, to zamalowywany jest obszar wokół sąsiadów spełniających ten warunek i następnie podobnej weryfikacji podlegają sąsiedzi tych pikseli. Dzięki takiemu rozwiązaniu, gdy znaleziony zostanie niepasujący fragment obiektu o stałym kolorze, to zamalowana będzie nie tylko jego krawędź ale także jego wnętrze. Bez tej operacji wnętrze takich obiektów nie byłoby zaznaczane, ponieważ weryfikowane piksele i sąsiedzi według których sprawdzamy poprawność, miałyby identyczny kolor.

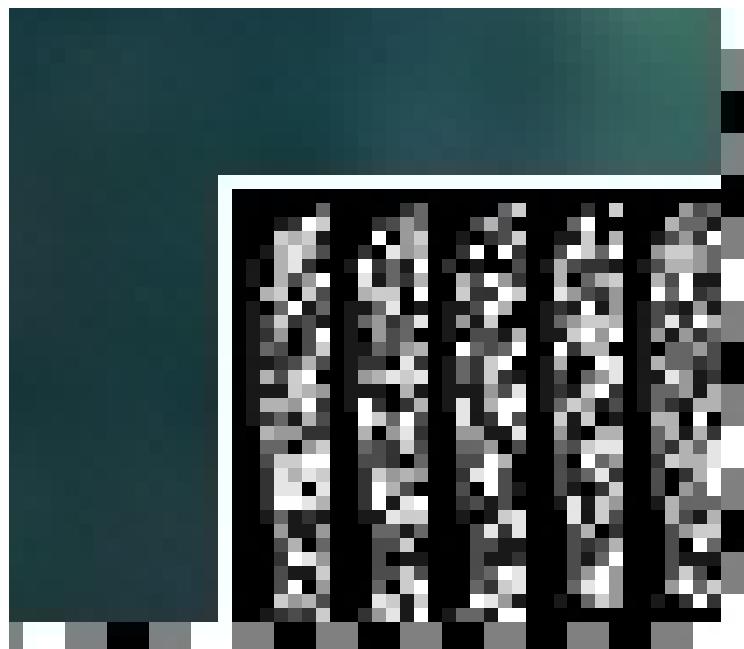
5.4 Znak zabezpieczający

Dane o histogramie sum barw zabezpieczonego obrazu mogą być źródłem podejrzeń o edycji obrazu. Dotychczas jedyną metodą zapisania liczebności grup sum barw było użycie pola komentarza użytkownika. Metadane EXIF są jednak bardzo narażone na ich edycję lub usunięcie. Było to powodem dla którego autor programu zdecydował się na stworzenie nowego rodzaju zabezpieczenia, które pozwala na zapisanie informacji o histogramie sum barw wewnątrz obrazu.

W trakcie zabezpieczania obrazu w prawym dolnym rogu obrazu tworzony jest prostokąt o wymiarach 38 na 34 pikseli, w którym zapisywane są informacje o stanie liczebnym pikseli przyporządkowanych do wszystkich grup. Założono, że maksymalna liczba pikseli dla grupy może wynosić 9.999.999. Liczba pikseli przynależna do grupy zapisywana jest w postaci 7 kolejnych pikseli, gdzie każdy piksel jest kolorowany na kolor odpowiadający jednej cyfrze całej liczby. Należy zapisać 153 liczb reprezentujących liczebność pikseli dla wszystkich grup sum barw. W jednym rzędzie zapisywanych jest do 5 liczb. Jest 31 rzędów z liczbami. Ostatni rząd zawiera tylko 3 liczby. Liczby zapisywane i czytane są od lewej do prawej, z góry na dół.

- Kolor czarny (0, 0, 0) oznacza cyfrę 0,
- Wartość koloru (25, 25, 25) oznacza cyfrę 1,
- Wartość koloru (51, 51, 51) oznacza cyfrę 2,
- Wartość koloru (76, 76, 76) oznacza cyfrę 3,
- Wartość koloru (102, 102, 102) oznacza cyfrę 4,
- Wartość koloru (153, 153, 153) oznacza cyfrę 5,
- Wartość koloru (178, 178, 178) oznacza cyfrę 6,
- Wartość koloru (204, 204, 204) oznacza cyfrę 7,
- Wartość koloru (229, 229, 229) oznacza cyfrę 8,
- Kolor biały (255, 255, 255) oznacza cyfrę 9.

Na początku próbowało malować piksele na „zwykłe” kolory. Niestety w trakcie zapisu piksele mocno zmieniały swój kolor. Dlatego postanowiono wykorzystać odcienie szarości. Dzięki temu rozwiązaniu piksele, które służą do przechowywania danych o histogramie nie ulegają poważnym zmianom. Problem polegał na zmianach kolorów RGB pikseli przez co liczby zapisane i odczytane ze znaku nie były ze sobą zgodne. Wykorzystanie odcieni szarości umożliwiło poprawny zapis i odczyt danych o histogramie zabezpieczonego obrazu. Dzięki temu zabezpieczeniu jesteśmy w stanie narysować histogram porównujący stan oryginalny z aktualnym stanem obrazu. Na rysunku 12 widoczny jest znak zabezpieczający.



Rys. 12: Znak zabezpieczający

5.5 Wykrywanie rotacji obrazu

Program jest w stanie wykryć rotację obrazu wynoszącą 0, 90, 180 lub 270, stopni. Wykrywanie rotacji obrazu umożliwione jest dzięki istnieniu znaku zabezpieczającego. Pierwotnie znak zabezpieczający zawsze znajduje się w prawym dolnym rogu obrazu. Sprawia to, że znajdująca zabezpieczenie jesteśmy w stanie wywnioskować o jaki kąt obraz został obrócony. Znalezienie znaku zabezpieczającego polega na zbadaniu rogu obrazu. Jeśli wszystkie piksele które sprawdzamy są koloru białego (255, 255, 255), to oznacza to, że w tym rogu znajduje się znak zabezpieczający. Obrócenie obrazu o inny kąt niż wielokrotność 90 stopni oznacza konieczność wypełnienia rogów obrazu jakimiś pikselami. Ich wypełnianie zależne jest od oprogramowania wykorzystanego do wykonania rotacji. Dodatkowo zależności między pikselami nałożone na obraz w trakcie jego zabezpieczenia i dane przechowywane w znaku zabezpieczającym zostaną naruszone. Autor uznał powyższe argumenty za wystarczające aby ograniczyć się do weryfikacji obrazów obróconych o wielokrotność 90 stopni.

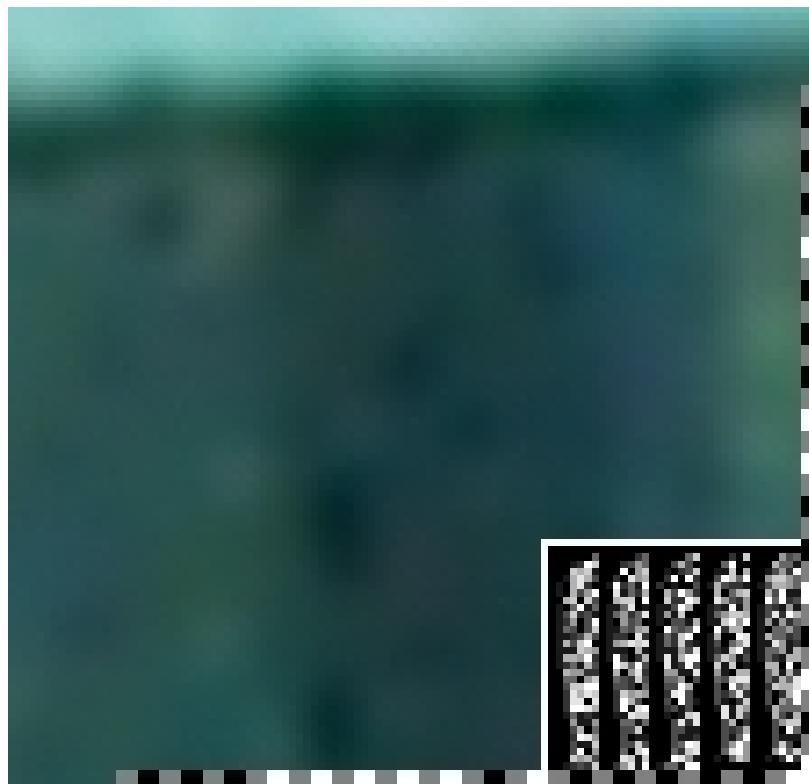
Zaproponowane rozwiązanie ma swoje problemy. Jeśli zabezpieczany obraz posiada rogi inne niż prawy dolny, które składają się z białych pikseli i zabezpieczony obraz ulegnie rotacji to istnieje możliwość błędnego wykrycia rotacji. Szansa na wystąpienie takiego ułożenia białych pikseli w obrazach naturalnych jest bardzo niska.

Zależnie od wykrytej rotacji obrazu sprawdzanie zależności między pikselami będzie przebiegało inaczej. W razie nieznalezienia pikseli koloru białego w żadnym z rogów obrazu, dalsze operacje wykonywane są tak, jakby wykryto rotację 0 stopni. Jeśli obrócono obraz i dokonano edycji pikseli mających pomóc wykryć rotację, to siatka sprawdzania poprawności zależności pomiędzy pikselami nie będzie adekwatna do obrazu, co spowoduje wskazanie ogromnych ilości miejsc gdzie mogło dojść do zmiany obrazu.

5.6 Zabezpieczenie wielkości obrazu

Podobnie jak w przypadku zabezpieczania informacji o histogramie obrazu, tutaj też występuje możliwość manipulowania danymi EXIF. Postanowiono zapisać oryginalną wielkość obrazu w obrazie. Możliwe jest odzyskanie informacji o rozmiarze obrazu w poziomie i pionie przy dowolnych powiększeniach obrazu a także zmniejszeniu obrazu maksymalnie do 50%.

Zabezpieczenie polega na zmianie kolorów pikseli w dolnej i prawej krawędzi obrazu. W dolnej krawędzi zapisujemy liczbę pikseli obrazu w poziomie. W prawej krawędzi zapisujemy liczbę pikseli w pionie. Wielkość oryginalna obrazu jest przekształcana na liczbę w systemie binarnym a następnie wpisywana w obraz wykorzystując 3 kolory pikseli. Kolor biały oznacza jedynkę, kolor czarny oznacza zero a kolor (128, 128, 128) jest stosowany jako separator 2 kolejnych ważnych bitów. Każda wartość pojedynczego bitu zapisywana jest w postaci bloku 2x3 piksele tej samej barwy. Dzięki temu przy zmianach wielkości obrazu zawsze co najmniej jeden piksel przy krawędzi ma oryginalny kolor. Po wczytaniu piksela oznaczającego ważny bit oczekujemy na wystąpienie separatora. Dopiero po znalezieniu separatora możliwe będzie wczytanie kolejnego ważnego bitu. W przypadku gdy aktualnie znajdujemy się na białych pikselach to dopóki nie natknemy się co najmniej jednego piksela oznaczającego separator, kolejne piksele nie zostaną uznane za kolejne ważne bity. Rozwiążanie takie jest konieczne ze względu na możliwość wystąpienia po sobie wielu pikseli reprezentujących ten sam bit. Dzięki temu nie liczymy tego samego bitu 2 razy.



Rys. 13: Zabezpieczenia wielkości obrazu i znak zabezpieczający

Na rysunku 13 widoczne są zabezpieczenia oryginalnej wielkości obrazu. Zabezpieczenie oryginalnej wysokości znajduje się na prawej krawędzi obrazu. Zabezpieczenie oryginalnej długości znajduje się na dolnej krawędzi obrazu. Wykorzystanie tej metody do poprawnego czytania oryginalnej wielkości obrazu dla większego zmniejszenia obrazu niż 50% odbyłoby się kosztem większych zmian w obrazie. Autor programu uznał ograniczenie zmniejszenia do 50% za wystarczające.

5.7 Wpływ zabezpieczeń na wygląd obrazu

Przedstawione zabezpieczenia mogą budzić zastrzeżenia dotyczące zbyt dużej ingerencji naszych metod w obraz. Wprowadzenie zależności między pikselami jest wykonywane na znaczącym obszarze obrazu, natomiast znak zabezpieczający i zabezpieczenia wielkości oryginalnej całkowicie zmieniają zawartość części pikseli. Jak się jednak okazuje wątpliwości te są niepotrzebne. W naszych badaniach posługujemy się obrazami o dużej wielkości co sprawia, że trudno jest zauważyć wpływ zabezpieczeń na obraz.



Rys. 14: Oryginalne zdjęcie przed dokonaniem na nim zabezpieczeń



Rys. 15: To samo zdjęcie po dokonaniu zabezpieczeń

Jak widać na rysunkach 14 i 15 zmiany wykonane na pikselach podczas zabezpieczania uśredniającego barwę RGB piksela nie są widoczne gołym okiem i nie zaburzają treści obrazu. Dopiero gdy wystarczająco mocno przybliżymy się do obrazu jesteśmy w stanie zauważać delikatne różnice w barwach pojedynczych pikseli.

Nie ma to natomiast większego wpływu na treść obrazu. Postrzegamy obraz tak samo zarówno przed jak i po jego zabezpieczeniu. W prawym dolnym rogu obrazu pojawił się znak zabezpieczający oraz zabezpieczenia wielkości które bez odpowiedniego przybliżenia są praktycznie niewidoczne. Nie są to zmiany mające duży wpływ na obraz. Zabezpieczenia są małe i dla wystarczająco dużych obrazów nie mają dużego wpływu na zamieszczone w nim treści. Możemy zatem uznać, że wykorzystane zabezpieczenia rzeczywiście ingerują w treść obrazu w bardzo ograniczony sposób i nie powinny one stanowić problemu lub niedogodności dla użytkowników programu.

6 Instrukcja obsługi programu

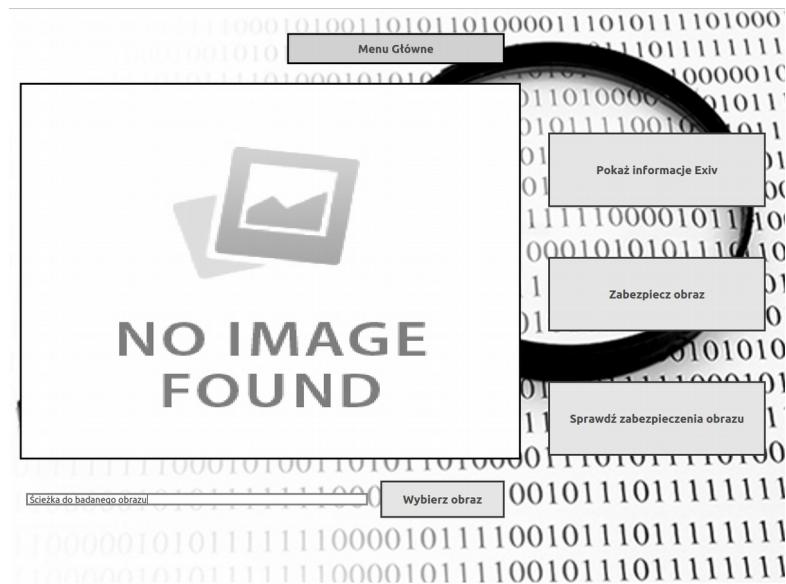
W trakcie tworzenia autor starał się, aby wszystkie kroki były jak najbardziej intuicyjne, ale w wypadku gdyby użytkownik nie wiedział od czego zacząć, w tym rozdziale dowiemy się w jaki sposób należy korzystać z oferowanego oprogramowania.

Interfejs programu składa się z 5 widoków:

- Widok okna głównego,
- Widok okna informacji exif i histogramów obrazu,
- Widok wszystkich informacji exif umieszczonych w pliku,
- Widok zabezpieczania wybranego obrazu,
- Widok weryfikacji zabezpieczeń.

W następnych podpunktach omówimy co można zrobić w każdym z nich.

6.1 Wybór pliku obrazu do badań



Rys. 16: Menu główne programu

Na rysunku 16 widać menu główne programu stworzonego w ramach tego projektu. Aby skorzystać z możliwości jakie daje to oprogramowanie należy wybrać plik z obrazem na którym możemy działać. Aby to uczynić można wpisać ścieżkę do pliku JPG lub wcisnąć przycisk „Wybierz obraz”. Program obsługuje tylko pliki JPG więc podanie ścieżki do innego rodzaju pliku nie umożliwi nam operacji na nim.



Rys. 17: Okno wyboru pliku

W oknie wyboru pliku pokazanym na rysunku 17, możemy przechodzić po katalogach komputera i wybrać jeden plik o rozszerzeniu .jpg poprzez dwukrotne kliknięcie lub pojedyncze kliknięcie i naciśnięcie przycisku „Open”. Po wpisaniu ścieżki do pliku na ekranie menu głównego pojawi się obraz zawarty w tym pliku. Jeśli korzystamy z okna wyboru pliku to dodatkowo w oknie menu ustawiona zostanie ścieżka do pliku. Na rysunku 18 przedstawiono widok menu głównego po wyborze obrazu.

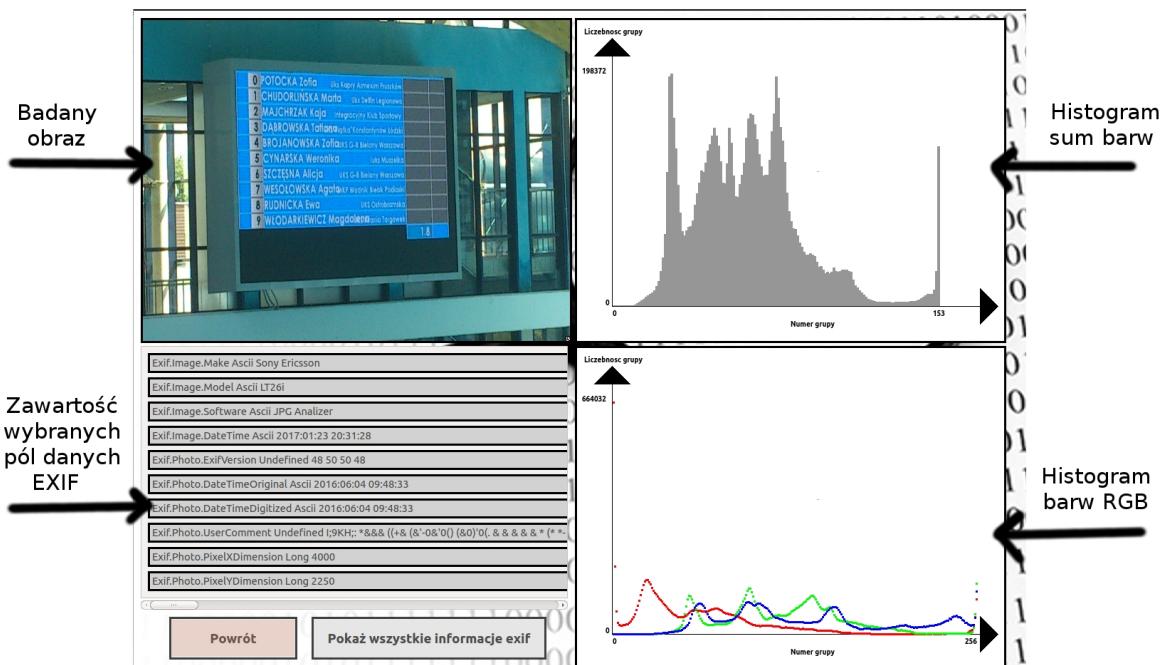


Rys. 18: Menu główne po wyborze pliku obrazu

Po wyborze pliku na których chcemy działać musimy zdecydować się co chcemy zrobić. Program daje nam trzy możliwości. Możemy wybrać opcję pokazania informacji exif i histogramów obrazu, możemy stworzyć plik w którym znajdzie się zabezpieczona wersja wybranego obrazu oraz możemy wybrać opcję weryfikacji obrazu pod względem zabezpieczeń oferowanych przez program. Po prawej stronie widoczne są 3 przyciski których wciśnięcie powoduje wykonanie się odpowiednich operacji.

6.2 Informacje exif i histogramy obrazu

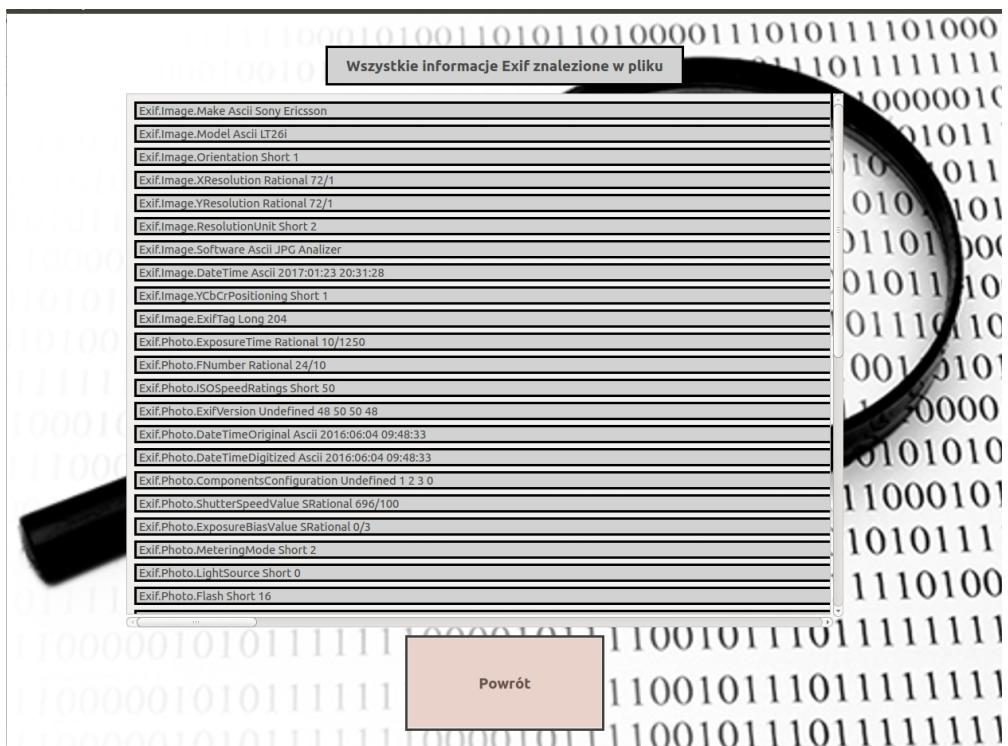
Jak już wiemy wiedza o metadanych Exif może dać nam powód do podejrzewania, że obraz mógł ulec edycji. To samo można powiedzieć o histogramach obrazu. Po wyborze opcji „Pokaż informacje Exif” pokaże nam się okno, na którym pokazana zostanie zawartość kilku wybranych pól metadanych wraz z histogramem sum barw oraz histogramem barw RGB.



Rys. 19: Okno zawierające informacje exif i histogramy obrazu

Na rysunku 19 widać przykład tego co może pokazać okno „Informacje exif oraz histogramy obrazu”. W lewej-górnej części widoczny jest obraz na podstawie którego wyciągane są informacje. W lewej-dolnej części widać zawartość pól exif wybranych przez autora za istotne pod względem weryfikacji oryginalności zdjęcia. Obejmują one informacje takie jak marka i model aparatu, który posłużył do wykonania obrazu, nazwa programu, użyta do ostatniej edycji zdjęcia czy data i czas ostatniej edycji. Dzięki polu komentarza możemy też łatwo sprawdzić, czy obraz ten został zabezpieczony za pomocą naszego komentarza. Przykładem może być sytuacja gdzie mamy do czynienia z wieloma plikami i nie wiemy czy zabezpieczaliśmy już wybrane zdjęcie. Mamy możliwość obejrzenia wszystkich metadanych exif umieszczonych w pliku, co możemy zrobić poprzez kliknięcie przycisku „Pokaż wszystkie informacje exif”.

W prawej-górnej części widoczny jest histogram sum barw wyliczony na podstawie naszego obrazu. W prawej-dolnej części widzimy histogram barw RGB. Poprzez kliknięcie przycisku „Powrót” możemy powrócić do okna menu głównego.



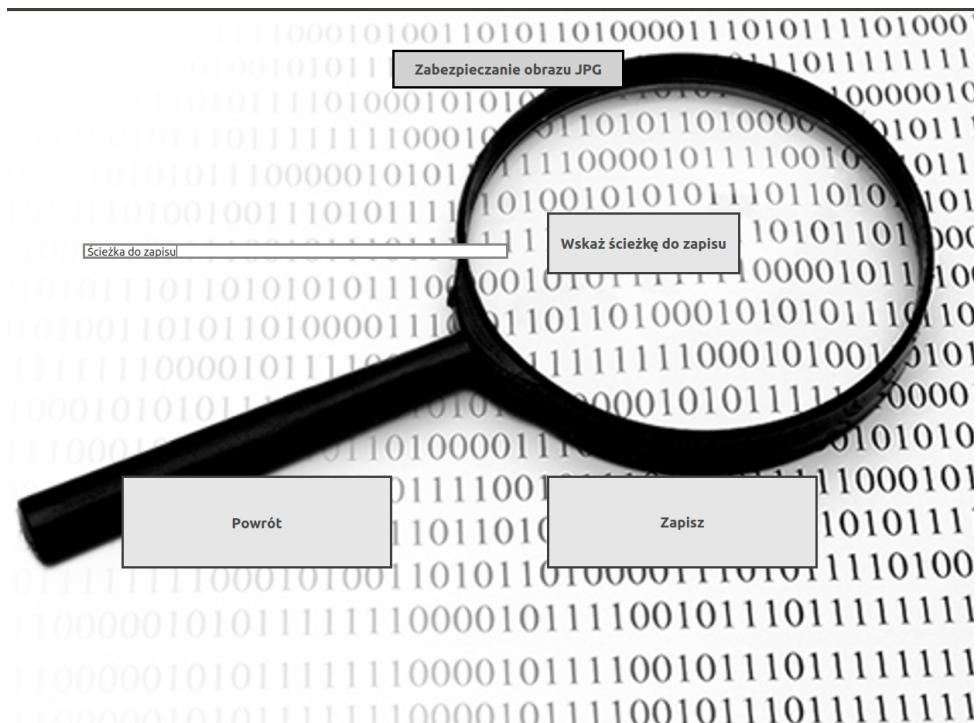
Rys. 20: Okno przedstawiające wszystkie informacje exif umieszczone w pliku z obrazem

Czasami możemy chcieć zobaczyć zawartość innych pól exif aniżeli te, umieszczone w oknie informacji exif i histogramów. Dlatego powstał widok przedstawiający wszystkie metadane exif umieszczone w obrazie. Możemy uzyskać takie informacje jak rozdzielcość obrazu, ustawienia aparatu wykorzystanego do stworzenia obrazu, czas naświetlenia elementu światłoczułego i wiele innych. Wygląd tego okna pokazano na rysunku 20.

Zawartość niektórych pól takich jak komentarz wytwórcy („MakerNote”) może być niezrozumiała dla czytelnika ze względu na niesprecyzowany w standardzie sposób przechowywanych informacji. Niektóre pola mają niezdefiniowany typ przechowywanych danych, co sprawia, że różni wytwórcy mogą inaczej wprowadzać do nich dane przez co sposób zapisu informacji w części z tych pól jest ścisłe uzależnione od marki aparatu, który posłużył do wykonania zdjęcia. Więcej informacji o standardzie EXIF na [7].

6.3 Zabezpieczanie wybranego obrazu

Po wyborze opcji „Zabezpiecz obraz” w oknie menu głównego programu pokaże nam się okno widoczne na rysunku 21.



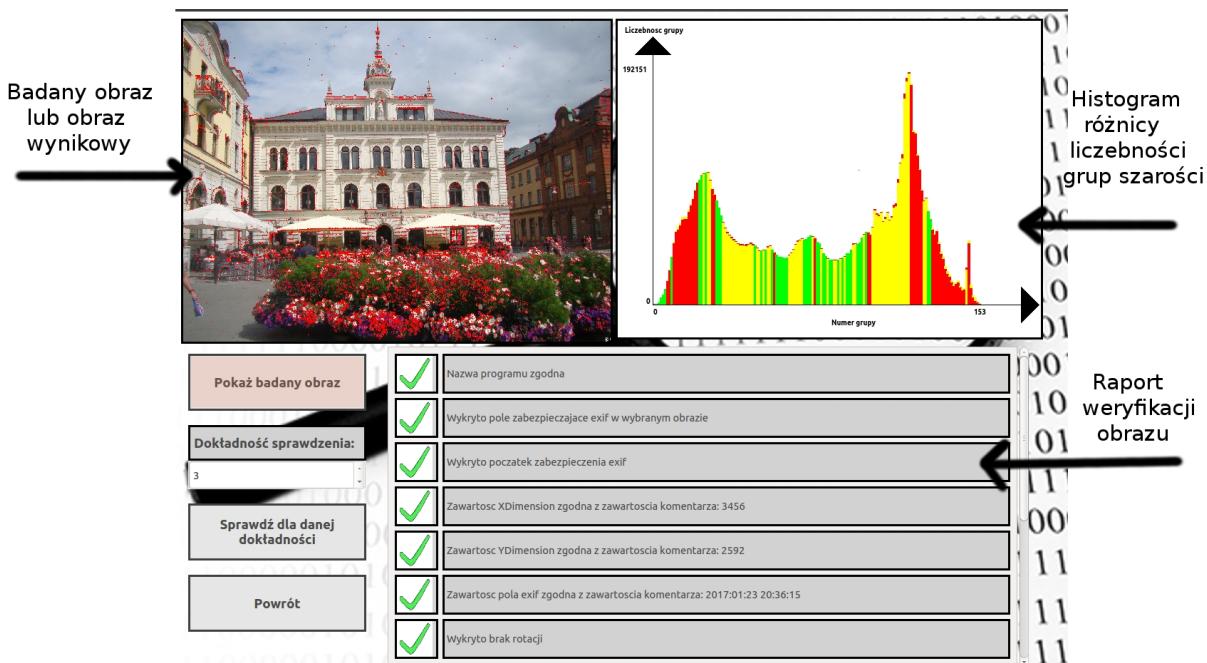
Rys. 21: Widok zabezpieczenia obrazu

Czynności jakie trzeba wykonać, aby zabezpieczyć obraz polegają na wybraniu miejsca i nazwy pliku z zabezpieczoną wersją obrazu i wciśnięcia przycisku „Zapisz”.

Podobnie jak to było w przypadku menu głównego, aby wskazać miejsce i nazwę pliku możemy wpisać pełną ścieżkę albo wcisnąć przycisk „Wskaż ścieżkę do zapisu”, co spowoduje pojawienie się podobnie wyglądającego okienka do tego co widzieliśmy w wyborze pliku w menu głównym. Po wybraniu miejsca zapisu, nazwy pliku oraz wciśnięcia przycisku „Zapisz” tworzony lub nadpisywany jest zabezpieczony obraz w pliku o podanej nazwie i wybranej lokalizacji. Jeśli jednak zdecydujemy się nie zabezpieczać obrazu możemy wcisnąć przycisk powrotu do menu głównego.

6.4 Weryfikacja oryginalności obrazu

W tym podpunkcie znajduje się opis najważniejszego widoku całego programu. Jest to okno przedstawiające wyniki badań autentyczności obrazu. Na rysunku 22 przedstawiono przykładowe okno wynikowe. W prawej-górnej części rysunku widać porównanie histogramów sum barw, o którym pisano wcześniej. Na dole znajduje się raport wyników weryfikacji obrazu. Część rekordów tego raportu dotyczy operacji wykonanych na metadanych exif z badanego pliku. Inne dotyczą operacji mających na celu analizę i wydobycie danych z samego obrazu. Rysunek zielonego ptaszka położony po lewej stronie od opisu rekordu raportu oznacza, że proces przebiegał zgodnie z założeniami i konkretny aspekt analizy nie daje podejrzeń, że obraz uległ edycji. Jeśli napotkano problem w tym samym miejscu pojawi się rysunek czerwonego krzyżyka wraz z odpowiadającym opisem problemu. Wciśnięcie przycisku „Powrót” spowoduje powrót do widoku menu głównego aplikacji.



Rys. 22: Widok weryfikacji obrazu

W lewej-górnej części widoku jest miejsce, gdzie pokazany może być obraz weryfikowany lub obraz będący wynikiem weryfikacji. Jeśli aktualnie wyświetlany jest badany obraz i chcemy obejrzeć obraz wynikowy, należy wcisnąć przycisk „Pokaż wynikowy obraz”. Natomiast jeśli aktualnie pokazywany jest obraz wynikowy, to wciśnięcie przycisku „Pokaż badany obraz” spowoduje zastąpienie obrazu wynikowego obrazem badanym.

7 Przykłady działania programu

W tym punkcie zobaczymy jak stworzony program radzi sobie z wykrywaniem naniesionych zmian na obraz. Omówione zostaną zmiany jakie dokonano na plikach obrazu oraz dowiemy się na podstawie czego można stwierdzić, że obraz został zedytowany. Do dokonania edycji na obrazach posłużyono się programem GIMP.

7.1 Przykład 1



Rys. 23: Zabezpieczone zdjęcie testowe numer 1

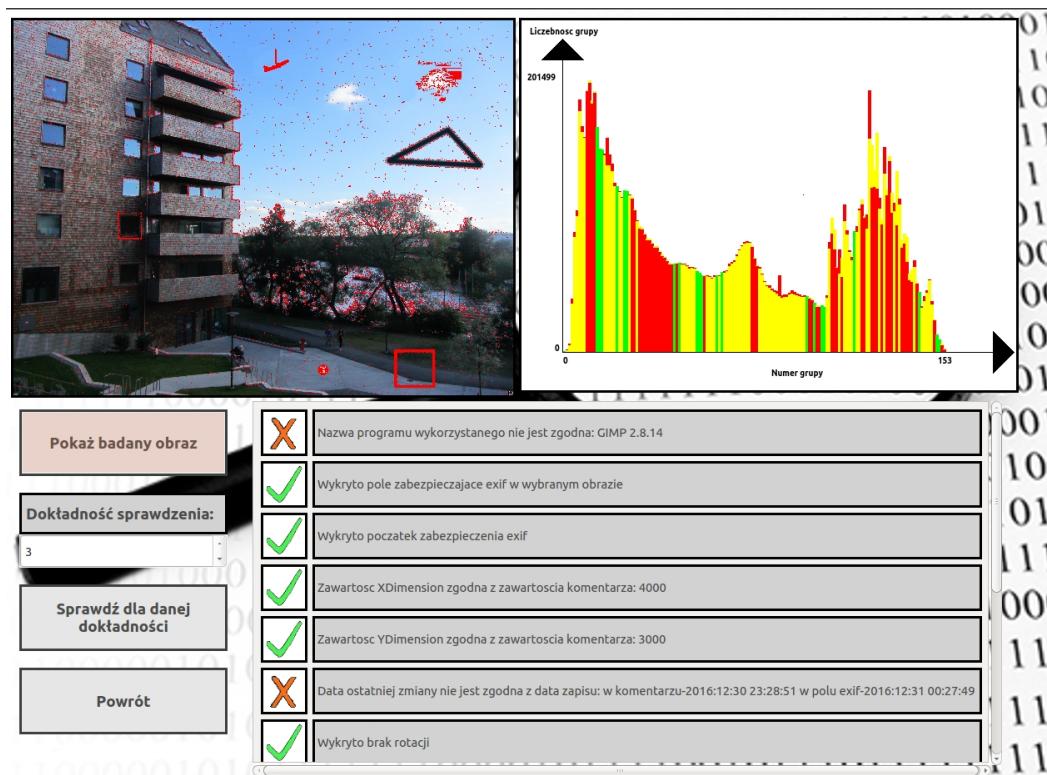


Rys. 24: Zabezpieczone zdjęcie które uległo edycji

Na rysunku 23 widoczne jest zdjęcie zabezpieczone, które postanowiono edytować. W efekcie zmian uzyskano zdjęcie widoczne na rysunku 24.

Zmiany wprowadzone na obraz:

- Skopiowano chmurę i wklejono obok, edytowano otoczenie chmury, tak by utrudnić dostrzeżenie kopii obserwatorowi,
- Wklejono obcy obiekt – samolot i podobnie jak poprzednio edytowano jego otoczenie,
- Skopiowano okno obok najniższego balkonu i wklejono w miejsce okna na drugim piętrze,
- Narysowano trójkąt za pomocą trzech prostych. Krawędzie zewnętrzne nie są „ostre”, jest łagodne przejście pomiędzy kolorem trójkąta, a tłem,
- Wklejono obcy obiekt – piłkę,
- Krawędzie prostokąta pomalowano stałym kolorem.



Rys. 25: Widok przedstawiający wyniki badań edytowanego zdjęcia testowego numer 1

Jak widać na rysunku 25 program całkiem dobrze poradził sobie z wykrywaniem wprowadzonych zmian. Program zaznaczył na czerwono obszary, które mogły ulec edycji. Pomimo tego, że program nie uznał okolic samolotu za obszar edytowany, to sam samolot został uznany został za element niepasujący do swojego otoczenia. Otoczenie samolotu nie zostało zamalowane na czerwono, co oznacza, że program nie uznał go za obszar niepasujący do obrazu. Okolice wklejonej kopii chmury zostały zamazane wystarczająco mocno, by obserwator zwrócił na to uwagę. Wokół wklejonej kopii okna została zamalowana czerwona linia co oznacza, że krawędzie wokół wklejonej części obrazu nie pasują do otoczenia w nowym miejscu. Wklejona piłka została zamalowana prawie całkowicie, co również jest rezultatem zadowalającym. Zamalowane stałym kolorem krawędzie prostokąta zostały zamalowane, co oznacza, że nie pasuje on do otoczenia.

Największym problem okazuje się trójkąt z krawędziami w kolorze przejściowym od wnętrza do otoczenia. Wewnętrzna część nie została uznana za naruszenie ze względu na to, że kolor wewnętrz jest stały, więc sprawdzane piksele oraz ich sąsiedzi mają identyczny kolor. Można jednak zauważać, że część otoczenia trójkąta została zamalowana na czerwono. Trójkąt został stworzony poprzez użycie narzędzia „Pędzel”, którego użycie powoduje, że krawędzie rysowanych obiektów mają przejściowy kolor pomiędzy wnętrzem a otoczeniem. Sprawia to, że sprawdzane piksele, które należą do części przejściowej jako sąsiadów mają piksele zewnętrzne oraz wnętrze. Same natomiast mają kolor znajdujący się pomiędzy kolorami zewnętrznymi i kolorem wewnętrznym trójkąta, co sprawia, że sąsiedzi sprawdzanego piksela mogą posiadać barwy RGB takie, że weryfikowany piksel będzie posiadał kolor będący średnią arytmetyczną ich barw RGB, czego następstwem będzie uznanie piksela za pasującego do obrazu. Jest to oznaka nieidealnego zabezpieczenia użytego w programie.

Dodatkową opcją jest weryfikacja obrazu z wykorzystaniem podanej dokładności sprawdzania. Wartość dokładności sprawdzania mówi jakie jest maksymalne odchylenie wartości barw RGB dla piksela przy sprawdzaniu zależności między pikselami. Jeśli wartość każdej z barw dla piksela zabezpieczenia różni się maksymalnie o dokładność weryfikacji, to piksel jest uznawany za zgodny z zabezpieczeniem. Im mniejsza jest wartość dokładności weryfikacji tym odchylenie może być mniejsze. Oznacza to, że program jest wtedy bardziej wyczulony na zmiany barw pikseli. Wartość dokładności sprawdzania ograniczono do przedziału od 0 do 20. Autor oprogramowania uznał, że takie ograniczenie jest wystarczające. Domyślną dokładnością weryfikacji wykorzystywana w programie jest 3. Aby skorzystać z możliwości ponownej weryfikacji obrazu z uwzględnieniem tego parametru należy wprowadzić nową wartość dokładności weryfikacji i nacisnąć przycisk „Sprawdź dla danej dokładności”.

Po wykonaniu obliczeń aktualny obraz z zaznaczonymi miejscami zmian zostanie zamieniony na nowy, uzyskany w efekcie obliczeń. Wykorzystanie tej możliwości może pozwolić uwidoczyć miejsca zmian obrazu w miejscach, gdzie wykrywana jest duża liczba „False positives”, czyli miejsc uznanych przez program za podejrzane pomimo braku ich edycji. Pomimo zastosowania takiej operacji użytkownik nadal może mieć wątpliwości.

Jeśli użytkownik uzna, że zamalowany obszar nie jest wystarczającym dowodem na uznanie obrazu za sfałszowanego, można posłużyć się analizą różnic histogramów sum barw obrazu. Jak widać są grupy sum barw, dla których liczebność przynależących pikseli znacząco różni się w danych historycznych obrazu oryginalnego wczytanych z zabezpieczenia exif. Jak już wcześniej pisałem kolor czerwony oznacza liczebność grup według aktualnego stanu obrazu, a kolor żółty oznacza liczebność pochodząą z zapisanych danych. W dwóch miejscach prostokąt czerwony jest znacznie wyższy od prostokąta żółtego. Obserwując obraz i znając zmiany, które nastąpiły w obrazie możemy stwierdzić, że grupy te dotyczą pikseli o barwach krawędzi dorysowanego trójkąta i prostokąta. Sam fakt dużej różnicy liczebności grup można uznać za wystarczający do uznania obrazu za sfałszowany.

W dolnej części rysunku widać raport tworzony w trakcie weryfikacji obrazu. Treść raportu brzmi tak:

- Nazwa programu wykorzystanego nie jest zgodna: GIMP 2.8.14,
- Wykryto pole zabezpieczające exif w wybranym obrazie,
- Wykryto początek zabezpieczenia exif,
- Zawartość XDimension zgodna: 4000,
- Zawartość YDimension zgodna: 3000,
- Data ostatniej zmiany nie jest zgodna z data zapisu: oryginalna-2016:12:30 23:28:51 aktualna-2016:12:31 00:27:49,
- Wykryto brak rotacji,
- Oryginalna wielkość obrazu X wczytana z powodzeniem: 4000,
- Oryginalna wielkość obrazu Y wczytana z powodzeniem: 3000,
- Histogram stworzony na podstawie danych zapisanych w polu exif,
- Nie było potrzeby sprawdzenia poprawności znaku zabezpieczającego.

Nazwa programu wykorzystanego nie jest zgodna. Nazwa programu jest zgodna tylko, jeśli jest to JPG Analyzer. Do ostatniej operacji na pliku został wykorzystany program GIMP, co jest powodem, dla którego można przypuszczać, że obraz mógł ulec edycji.

„Wykryto pole zabezpieczające exif w wybranym obrazie” oznacza, że w badanym pliku znajduje się pole metadanych UserComment.

„Wykryto początek zabezpieczenia exif” oznacza, że znaleziono słowo kluczowe SECURED, które rozpoczyna zapisywany ciąg informacji o oryginalnym obrazie.

„Zawartość XDimension zgodna: 4000” oznacza, że aktualna zawartość pola Xdimension jest zgodna z wartością historyczną zapisaną w polu komentarza exif.

„Zawartość YDimension zgodna: 3000” oznacza, to samo ale tym razem dotyczy liczby pikseli w pionie.

„Data ostatniej zmiany nie jest zgodna z data zapisu: oryginalna-2016:12:30 23:28:51 aktualna-2016:12:31 00:27:49” oznacza, że data i czas nie są sobie równe, co jest kolejnym powodem do podejrzewania, że obraz mógł ulec edycji. „Wykryto brak rotacji” oznacza, że program znalazł odpowiednio pomalowane piksele w dolnym prawym rogu obrazu, więc dalsza weryfikacja przebiega tak, jakby nie było rotacji obrazu.

„Oryginalna wielkość obrazu X wczytana z powodzeniem: 4000” i „Oryginalna wielkość obrazu Y wczytana z powodzeniem: 3000” oznacza, że zarówno zabezpieczenie oryginalnej wielkości w pionie jak i poziomie zostało odczytane bez problemów i wynoszą odpowiednio 4000 i 3000 pikseli. Użytkownik może dzięki temu sprawdzić, czy wartości te są równe danym zapisanym w metadanych exif. W tym przypadku wartości te są równe, więc pod tym względem obraz nie budzi zastrzeżeń. Jeśli edycji ulegną zabezpieczenia wielkości, to pojawić się mogą problemy w czytaniu z zabezpieczenia, co może być powodem do przypuszczeń, że obraz uległ edycji.

„Histogram stworzony na podstawie danych zapisanych w polu exif” oznacza, że skoro pole komentarza istnieje i ma zawartość, to dane historyczne o obrazie zabezpieczanym mogą być wczytane z tego źródła. Nie ma wtedy potrzeby czytania tych samych danych ze znaku zabezpieczającego, dlatego następną informacją napisaną w raporcie jest „Nie było potrzeby sprawdzenia poprawności znaku zabezpieczającego”.

Raport ten może dać wiele informacji dla użytkownika programu, dlatego warto czytać jego zawartość.

7.2 Przykład 2

W tym przykładzie testowym zobaczymy, jak program radzi sobie z małowidocznymi zmianami obrazu.

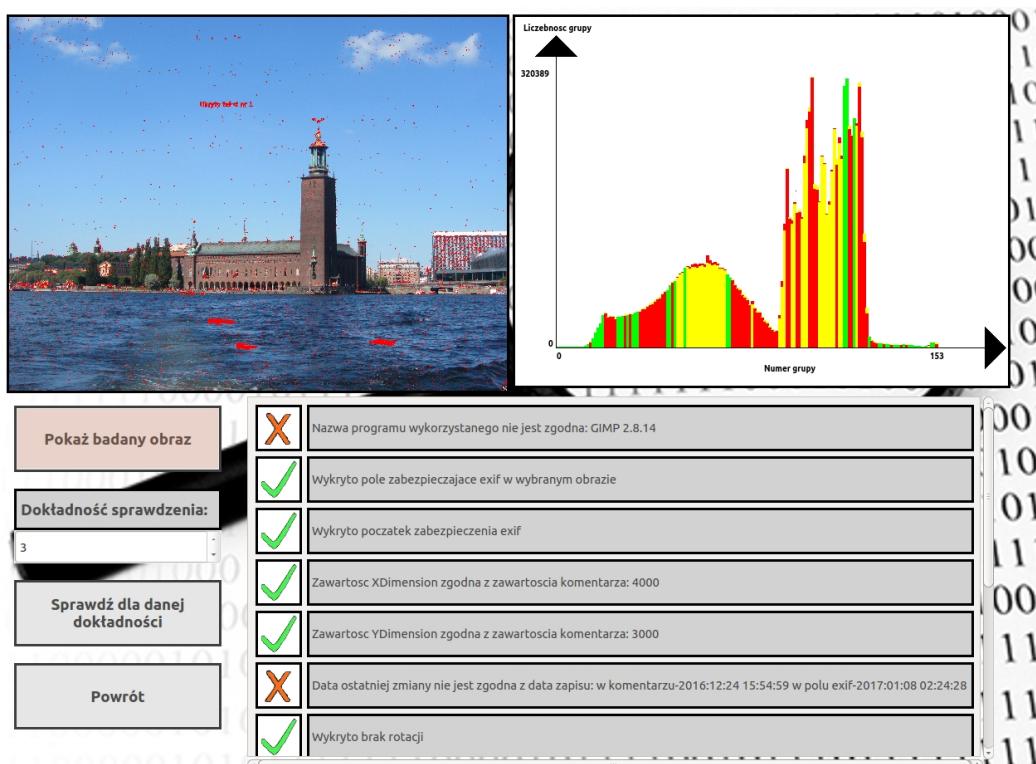


Rys. 26: Zabezpieczone zdjęcie testowe numer 2



Rys. 27: Zabezpieczone zdjęcie testowe które uległo edycji

W tym przypadku zdjęcie, które uległo edycji jest niemalże identyczne ze zdjęciem oryginalnym. Jednak zdjęcie umieszczone na rysunku 27 jest wynikiem nałożenia kilku bardzo mało widocznych zmian na zabezpieczony obraz widoczny na rysunku 26. Aby dowiedzieć się, czy zdjęcie uległo edycji możemy wykorzystać przygotowany program.



Rys. 28: Okno weryfikacji obrazu dla domyślnej wartości dokładności równej 3

Na rysunku 28 widać wynik działania programu. Program wykrył 4 zmiany nałożone na obraz. 3 z nich polegały na pobraniu koloru tła i zamalowaniu części obrazu tym kolorem. Czwarta zmiana polegała na stworzeniu napisu za pomocą koloru tła. Pozostałe punktowe obszary zaznaczone na czerwono mogą być spowodowane lekkimi zmianami wartości RGB pojedynczych pikseli. Powiększony obraz będący wynikiem weryfikacji pokazany został na rysunku 29.

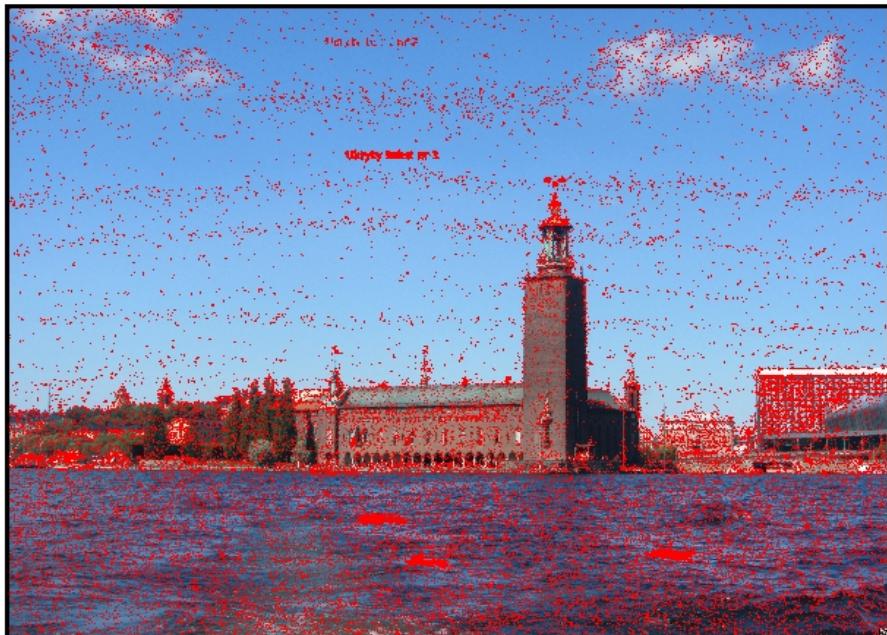
Autor programu zdecydował się umożliwić weryfikację obrazu w zależności od parametru maksymalnego akceptowalnego wychylenia barw RGB od średniej z sąsiadów. Im mniejsza wartość dokładności sprawdzania, tym bardziej rygorystyczny będzie proces weryfikacji siatki pikseli w obrazie. Aby wykorzystać tę możliwość należy wpisać nową wartość w polu pod napisem „Dokładność sprawdzenia” i nacisnąć przycisk „Sprawdź dla danej dokładności”.



Rys. 29: Wynik weryfikacji obrazu dla domyślnej wartości dokładności wynoszącej 3



Rys. 30: Wynik weryfikacji obrazu dla wartości dokładności wynoszącej 4



Rys. 31: Wynik weryfikacji obrazu dla wartości dokładności wynoszącej 2

Jak widzimy na rysunku 30, zwiększenie dokładności spowodowało zmniejszenie liczebności czerwonych miejsc. Na rysunku 31 widać efekt zmniejszenia dokładności. Dzięki zastosowaniu tej funkcjonalności użytkownik jest w stanie zauważać miejsce w którym doszło do ostatniej edycji zdjęcia. W górnej części obrazu zaznaczony został obszar, gdzie stworzono drugi napis z wykorzystaniem koloru tła. Zmiany takie są trudne do zauważenia przez użytkownika ale dzięki zmianie wrażliwości są do wykrycia. Trzeba jednak podkreślić, że na rysunku 31 powstało bardzo wiele miejsc uznanych za podejrzane, co sprawia, że wykrycie ostatniej zmiany jest mało widoczne i użytkownik programu mógłby uznać ją za jedno z bardzo wielu miejsc oznaczających lekkie zmiany w obrazie. Jeśli zastosujemy wyższą wartość parametru dokładności pokazanych zostanie mniej fałszywych wykryć zmian („False positives”), przez co wykryte zmiany będą lepiej widoczne, natomiast spowoduje to też, zmniejszenie wrażliwości programu na zmiany w obrazie, co może być przyczyną niewykrycia niektórych typów fałszerstw.

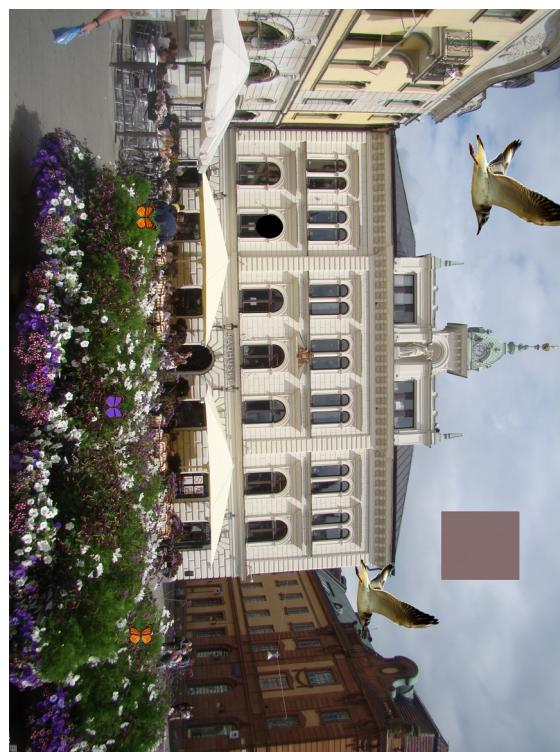
Jest to przykład świadczący o tym, że są zmiany obrazu, które bardzo trudno jest wykryć. Przygotowane zabezpieczenie obrazu nie jest przystosowane do radzenia sobie z tego typu zmianami. Pocieszeniem jest tu fakt, że zmiana ta ma niewielkie znaczenie dla treści obrazu. W takiej sytuacji jeśli analiza metadanych i histogramu obrazu nie da powodów do podejrzeń, to fałszerstwo cyfrowe może zostać niewykryte za pomocą stworzonego oprogramowania. Jest to dowód, że program nie jest narzędziem idealnym, ale powinien on wystarczyć do wykrycia większości typów fałszerstw.

7.3 Przykład 3

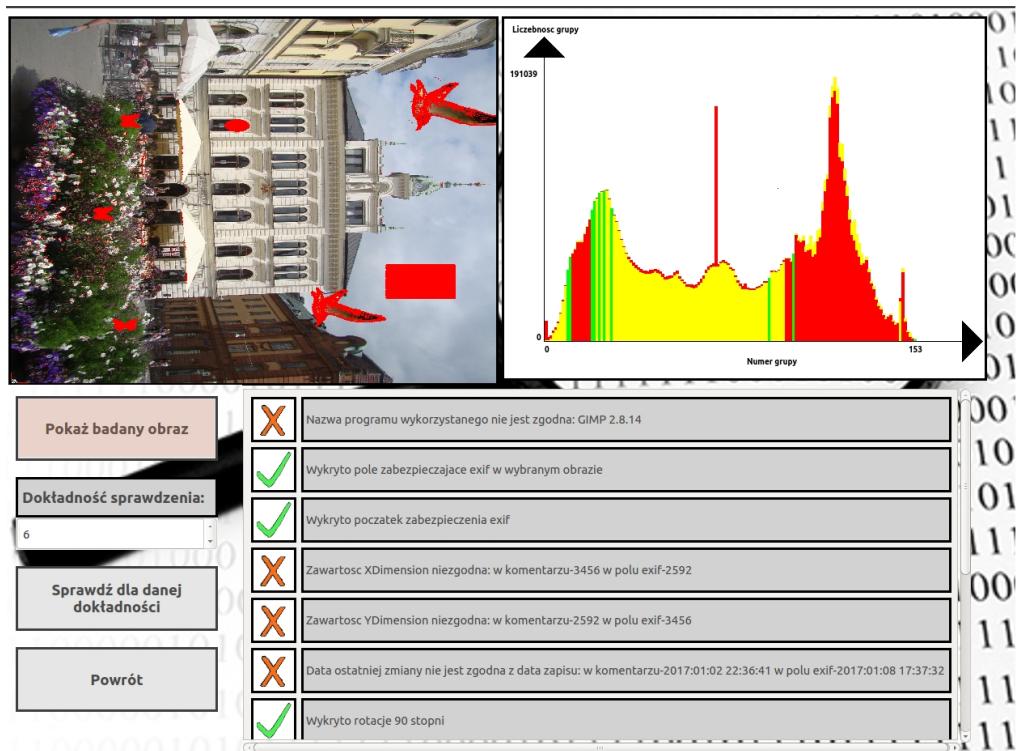
W tym przykładzie testowym zobaczymy, jak program radzi sobie z rotacją obrazu.



Rys. 32: Zabezpieczone zdjęcie testowe numer 3



Rys. 33: Zabezpieczone zdjęcie które uległo edycji



Rys. 34: Wynik weryfikacji przy dokładności sprawdzania równej 6

Rysunek 32 przedstawia zabezpieczony obraz. Rysunek 33 przedstawia obraz będący wynikiem edycji. Rysunek 34 pokazuje okno wynikowe weryfikacji.

Pierwszą rzeczą rzucającą się w oczy jest to, że obraz oryginalny został obrócony o 90 stopni. Po ustaleniu wartości dokładności sprawdzania możemy dokładnie określić, co uległo edycji w zdjęciu.

Zmiany polegały na umieszczeniu ciał obcych w postaci 2 ptaków i 3 motyli. Pozostałe zmiany polegały na zamalowaniu dwóch figur na wybrany kolor. Dodatkowo obraz obrócono o 90 stopni. Różnice wykryte w histogramach sum barw mogą być kolejnym dowodem manipulacji obrazem. Są grupy pikseli których liczebność jest znacznie większa aniżeli w oryginalnym obrazie.

Raport wykonany w trakcie weryfikacji obrazu brzmi w następujący sposób:

- Nazwa programu wykorzystanego nie jest zgodna: GIMP 2.8.14,
- Wykryto pole zabezpieczające exif w wybranym obrazie,
- Wykryto początek zabezpieczenia exif,
- Zawartość XDimension niezgodna: w komentarzu-3456 w polu exif-2592,

- Zawartość YDimension niezgodna: w komentarzu-2592 w polu exif-3456,
- Data ostatniej zmiany nie jest zgodna z data zapisu: w komentarzu-2017:01:02 22:36:41 w polu exif-2017:01:08 17:37:32,
- Wykryto rotację 90 stopni,
- Oryginalna wielkość obrazu X wczytana z powodzeniem: 3456,
- Zabezpieczenie oryginalnej wielkości X niezgodne z aktualna wielkością obrazu wynoszącą: 2592,
- Oryginalna wielkość obrazu Y wczytana z powodzeniem: 2592,
- Zabezpieczenie oryginalnej wielkości Y niezgodne z aktualna wielkością obrazu wynoszącą: 3456,
- Histogram stworzony na podstawie danych zapisanych w polu exif,
- Nie było potrzeby sprawdzania poprawności znaku zabezpieczającego.

W tym przykładzie skupiamy się na wykrywaniu rotacji obrazu. Dlatego największą uwagę należy zwrócić na pozycje, które mogą świadczyć o rotacji i zmianie wielkości obrazu. Zawartość zarówno pole XDimension jak i YDimension nie zgadza się z wartością zapisaną w polu komentarza użytkownika. Możemy zauważyć, że wartości wielkości obrazu uległy przestawieniu. Pozycja „Wykryto rotację 90 stopni” mówi nam, że program na podstawie położenia znaku zabezpieczającego przypuszcza, że obraz obrócono o 90 stopni. Powoduje to, że dalsze badania obrazu oparte są na założeniu, że obraz rzeczywiście obrócono o taki kąt. Ostatnimi informacjami mogącymi być powodem uznania obrazu za edytowany są niezgodności przeczytanych zabezpieczeń wielkości z aktualną wielkością obrazu.



Rys. 35: Wynik weryfikacji dla dokładności równej 6

Jak widać na rysunku 35 program wykrył rotację obrazu i zmiany dokonane na obrazie. Dobrze widoczne są wszystkie zmienione fragmenty obrazu, co można uznać za sukces. W przypadku, gdy użytkownik nie jest w stanie samodzielnie zauważyc, że obraz uległ rotacji, raport weryfikacji może być wykorzystany jako dowód zmiany obrazu.

7.4 Przykład 4

W tym przykładzie przedstawiony zostanie efekt analizy zdjęcia, które uległo zmianie wielkości. W trakcie zabezpieczania obrazu, na obraz nakładane są zabezpieczenia oryginalnej wielkości. Jednym z etapów weryfikacji jest czytanie z tego zabezpieczenia i sprawdzenie, czy aktualna wielkość obrazu jest równa odczytanym wartościom.

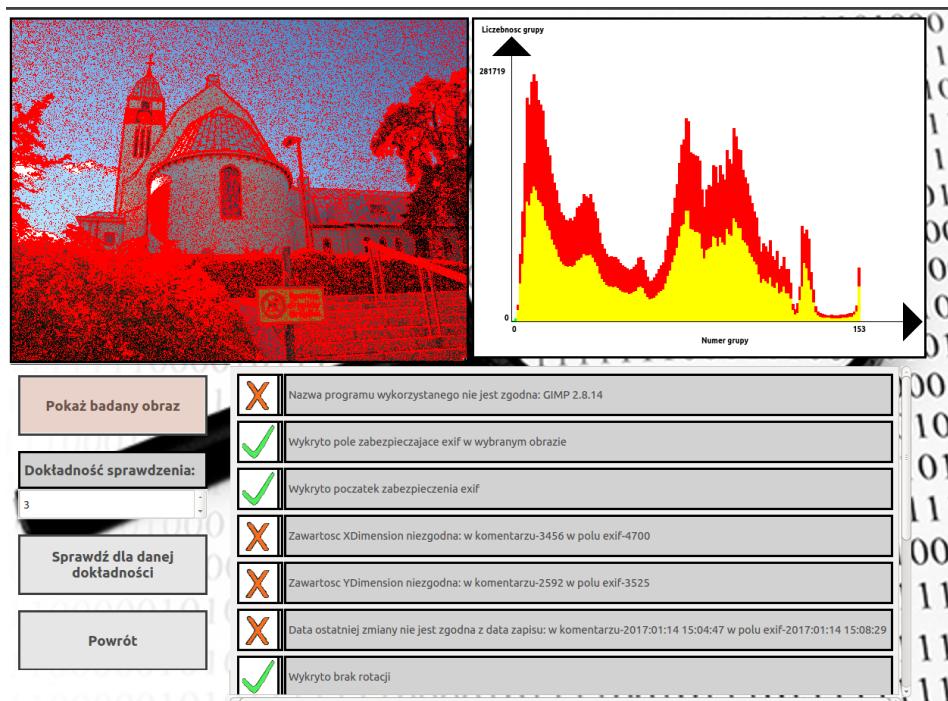


Rys. 36: Zabezpieczone zdjęcie testowe numer 4. Wielkość 3456 x 2592 piksele

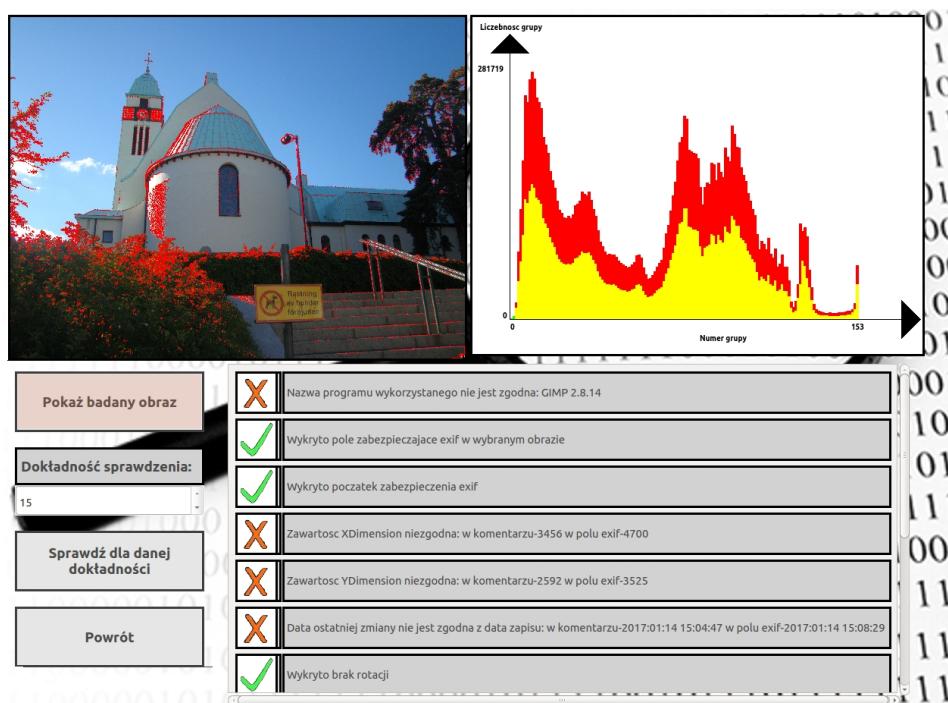


Rys. 37: Powiększone zdjęcie testowe numer 4. Wielkość 4700 x 3525 pikseli

Na rysunku numer 37 widoczny jest obraz uzyskany poprzez przeskalowanie obrazu oryginalnego widocznego na rysunku 36.



Rys. 38: Wyniki badań nad obrazem powiększonym z użyciem domyślnej dokładności weryfikacji równej 3



Rys. 39: Wynik badań tego samego obrazu dla wartości dokładności równej 15

Na rysunku 38 przedstawiono wynik badań nad powiększonym obrazem z zastosowaniem dokładności wynoszącej 3. Na rysunku 39 znajduje się wynik badań tego samego obrazu dla dokładności równej 15.

Powiększenie obrazu sprawiło, że dodane musiały zostać piksele, których barwy nie są zgodne z założeniami wprowadzonych zmian na pikselach. Przy weryfikacji sąsiedztw pikseli wykrywane jest przez to dużo nieprawidłowości. Nawet przy zastosowaniu parametru dokładności równemu 15, nadal wykrywane są nieprawidłowości. Oznacza to, że wyliczone średnie RGB w zaznaczonych obszarach odchycone jest bardziej niż o 15.

Warto zwrócić uwagę na ogromne różnice w histogramach sum barw. Widać, że liczba pikseli przyporządkowanych do poszczególnych grup sum barw dla aktualnego stanu obrazu jest większa, niż w oryginalnym. Jak widać suma pikseli ze wszystkich grup w aktualnym obrazie jest znacznie wyższa więc możemy dojść do wniosku, że obraz uległ powiększeniu.

W raporcie wykonanym podczas weryfikacji obrazu możemy zwrócić na kilka pól świadczących o niezgodności wielkości obrazu:

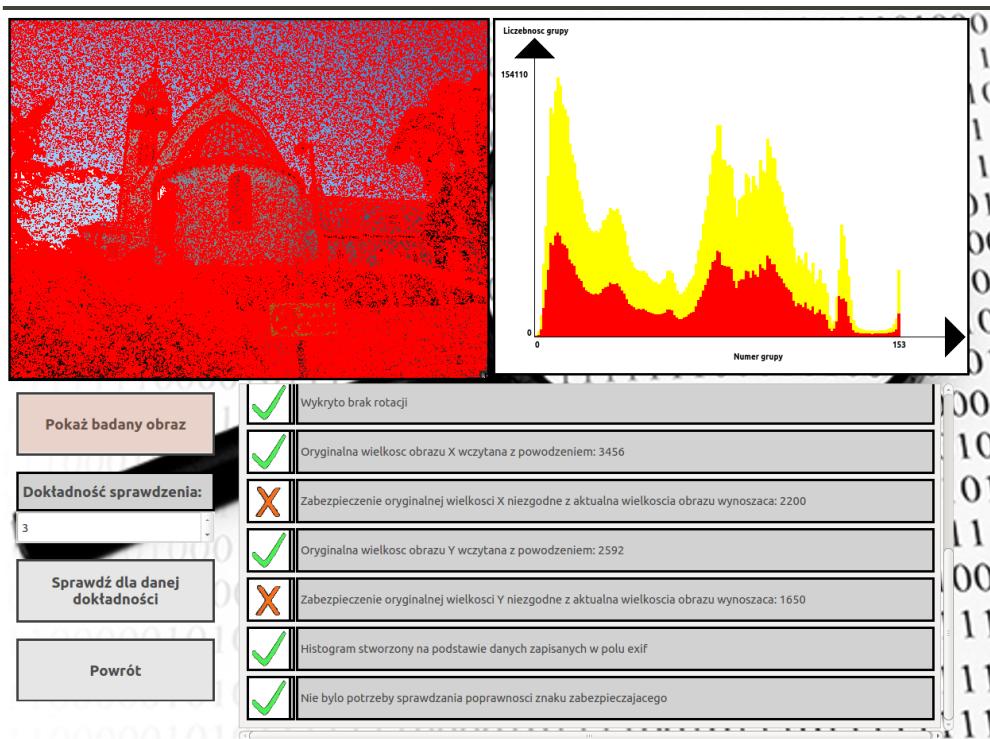
- Zawartość XDimension niezgodna: w komentarzu-3456 w polu exif-4700,
- Zawartość YDimension niezgodna: w komentarzu-2592 w polu exif-3525,
- Wykryto brak rotacji,
- Oryginalna wielkość obrazu X wczytana z powodzeniem: 3456,
- Zabezpieczenie oryginalnej wielkości X niezgodne z aktualną wielkością obrazu wynoszącą: 4700,
- Oryginalna wielkość obrazu Y wczytana z powodzeniem: 2592,
- Zabezpieczenie oryginalnej wielkości Y niezgodne z aktualną wielkością obrazu wynoszącą: 3525,

Do edycji obrazu wykorzystano program GIMP, który obsługuje metadane exif. Przy zmianie wielkości obrazu aktualizowana jest zawartość pól opisujących wielkość obrazu. Zawartość pola komentarza użytkownika nie jest zmieniana, co powoduje, że wartość zapisana w komentarzu i wartość wpisana w programie do edycji nie są sobie równe. Program uznał, że obraz nie był obracany, co oznacza, że w prawym dolnym rogu obrazu znalezione zostało miejsce, gdzie może znajdować się znak zabezpieczający.

Wielkość aktualna obrazu biorąc pod uwagę liczbę pikseli w poziomie jak i pionie nie jest zgodna z wielkością wyczytaną z zabezpieczeń wielkości obrazu. Odczytywanie oryginalnej wielkości obrazu zostało ukończone bez problemów. Wczytany rozmiar w poziomie wynosi 3456 pikseli, natomiast rozmiar w pionie wynosi 2592 piksele. Obie te wartości są zgodne z wielkością oryginalnego obrazu, co jest przykładem dowodzącym o prawidłowym zabezpieczeniu i wczytywaniu wielkości z zabezpieczonego obrazu.



Rys. 40: Zabezpieczony obraz pomniejszony. Wielkość 2200 x 1650 pikseli



Rys. 41: Wynik badań nad obrazem pomniejszonym z użyciem domyślnej dokładności weryfikacji równej 3



Rys. 42: Wynik badań nad obrazem pomniejszonym z użyciem dokładności weryfikacji równej 15

Na rysunku 40 przedstawiono pomniejszony obraz zabezpieczony. Na rysunkach 41 i 42 widać wyniki badań dla dokładności wynoszącej odpowiednio 3 i 15.

Podobnie jak w przypadku powiększania obrazu, gdy zmniejszymy obraz w trakcie weryfikacji wykryte zostanie wiele miejsc, gdzie wartość RGB piksela nie jest zgodna z średnią pikseli sąsiednich.

Tym razem ilości pikseli przyporządkowanych do poszczególnych grup dla obrazu aktualnego są znacznie mniejsze od liczb oryginalnych. Ważne pola na które trzeba zwrócić uwagę w trakcie weryfikacji obrazu to:

- Zawartość XDimension niezgodna: w komentarzu-3456 w polu exif-2200,
- Zawartość YDimension niezgodna: w komentarzu-2592 w polu exif-1650,
- Wykryto brak rotacji,
- Oryginalna wielkość obrazu X wczytana z powodzeniem: 3456,
- Zabezpieczenie oryginalnej wielkości X niezgodne z aktualną wielkością obrazu wynoszącą: 2200,
- Oryginalna wielkość obrazu Y wczytana z powodzeniem: 2592,
- Zabezpieczenie oryginalnej wielkości Y niezgodne z aktualną wielkością obrazu wynoszącą: 1650.

Podobnie jak to było w przypadku powiększenia obrazu wielkości z pól exif nie są identyczne z wartościami zawartymi w polu komentarza użytkownika. Brak rotacji oznacza, że program uznał, że znak zabezpieczający znajduje się w prawym dolnym rogu obrazu. Czytanie oryginalnej wielkości obrazu przebiegło bezproblemowo, wartości te są zgodne z oryginalnym zdjęciem i niezgodne z aktualnym stanem obrazu.

8. Plany dalszego rozwoju programu

Zakłada się, że w przyszłości praca nad projektem będzie kontynuowana. Istnieje bardzo wiele możliwości rozwoju stworzonego oprogramowania. Rozwój stworzonego oprogramowania może polegać na stworzeniu modułu korzystającego z zaproponowanych zabezpieczeń i zabezpieczającego obraz pochodzący bezpośrednio z aparatu cyfrowego lub z kamery przemysłowej. Aktualna wersja programu korzysta jedynie z dynamicznych metod wykrywania fałszerstwa, więc dobrym pomysłem byłoby wprowadzenie kilku metod statycznych. Przykładem takiej metody jest metoda wyszukująca sklonowane fragmenty obrazu opisana w punkcie 2.6.1. Dodatkowym argumentem przemawiającym za wprowadzeniem tej metody jest fakt, że aktualne zabeznięcie może nie poradzić sobie z wykryciem 2 identycznych fragmentów obrazu. Aktualnie wykorzystywane szyfrowanie danych wprowadzanych do pola komentarza użytkownika jest słabym zabezaniem. Obserwator znajdujący się na kryptografii jest w stanie szybko wykryć zależność pozwalającą na odczytanie zabezpieczonych informacji o obrazie. Dlatego do szyfrowania danych można by użyć bardziej skomplikowanej metody kryptograficznej. Główna metoda wykrywania zmian w obrazie także ma swoje słabe strony. Stworzenie nowej metody mogłoby poprawić efektywność programu. Komercyjne programy przeznaczone do wykrywania fałszerstw w obrazach cyfrowych wykorzystują wiele metod dzięki czemu są efektywne. Im większa jest liczba algorytmów weryfikacji obrazu, tym większa jest szansa na wykrycie fałszerstwa. Oznacza to, że wprowadzenie nowych zabezpieczeń poprawiłoby efektywność programu.

9.Podsumowanie

Biorąc pod uwagę jak ważną rzeczą dla funkcjonowania wielu organizacji jest konieczność korzystania ze zdjęć oryginalnych oraz fakt, że ciągły rozwój technologiczny daje coraz to nowe i bardziej dostępne metody manipulacji obrazem można stwierdzić, że zawsze będzie zapotrzebowanie na rozwój oprogramowania, służącego do badania ich oryginalności. Istnieje obecnie wiele programów zajmujących się znajdywaniem zmian dokonanych na obrazie. Dzięki ukrywaniu w tajemnicy metod wykorzystywanych do zabezpieczania i badania obrazu oprogramowanie jest skuteczniejsze. Im więcej ludzi wie, na czym polegają zabezpieczenia i weryfikacja obrazu w wybranym programie tym większe prawdopodobieństwo, że jedna osoba będzie w stanie zedytować obraz tak, aby oprogramowanie nie wykryło fałszerstwa.

Niniejsza praca jest próbą stworzenia programu z wykorzystaniem metod wymyślonych przez autora pracy. Po dowiedzeniu się nieco więcej o zabezpieczaniu obrazu autor był w stanie wymyślić sposób na zabezpieczenie obrazu i późniejszą jego weryfikację.

Wykorzystane w pracy metody okazują się być wystarczająco dobre do wykrycia podstawowych metod edycji obrazu. Program wskazuje na miejsca w obrazie, które mogły według niego ulec edycji, oraz przedstawia informacje o obrazie i przebiegu procesu weryfikacji obrazu. Przedstawiono w jaki sposób należy korzystać ze stworzonego oprogramowania i przykłady działania oprogramowania dla przykładowych obrazów oryginalnych. Jak w każdym oprogramowaniu tego typu, wykorzystane zabezpieczenia nie są niemożliwe do obejścia. Ich siła jest bardzo zależna od wiedzy użytkownika i osoby chcącej sfałszować obraz o czym mówiono w podpunkcie 3.4.

Bibliografia

- [1] Cezary Bołdak, „Cyfrowe Przetwarzanie Obrazów”, dostęp online:
<http://aragorn.pb.bialystok.pl/~boldak/DIP/CPO-W02-v3-50pr.pdf>
- [2] Cryptography, dostęp online: <https://en.wikipedia.org/wiki/Cryptography>
- [3] Steganography, dostęp online: <https://en.wikipedia.org/wiki/Steganography>
- [4] Hany Farid, „Image Forgery Detection A Survey,” IEEE SIGNAL PROCESSING MAGAZINE MARCH 2009, dostęp online:
www.cs.dartmouth.edu/farid/downloads/publications/spm09.pdf
- [5] Fotoblogia Dla początkujących: jak czytać dane EXIF, dostęp online:
<http://fotoblogia.pl/5524.dla-poczatkujacych-jak-czytac-dane-exif>
- [6] Adam Wojciechowski, Przetwarzanie obrazów Politechnika Poznańska, wykład 2, dostęp online: http://ics.p.lodz.pl/~adamwoj/WSFI/PO/2_wyklad_PO.pdf
- [7] Dokumentacja biblioteki Exiv2, dostęp online: exiv2.org
- [8] Dokumentacja biblioteki QT, dostęp online: www.qt.io
- [9] Kompresja JPEG Poradnik, dostęp online:
http://zbyma.gimpuj.info/Poradnik-Kompresja_JPEG.pdf
- [10] tutorialspoint Histograms introduction, dostęp online:
https://www.tutorialspoint.com/dip/histograms_introduction.htm
- [11] Salma Hamdy, Haytham El-Messiry, Mohamed Roushdy, Essam Kahlifa. „Quantization Table Estimation in JPEG Images.” IJACSA, Vol. 1, No. 6, December 2010, dostęp online:
<https://pdfs.semanticscholar.org/1997/39c35cc91112665cdf266194d5f5bee95e72.pdf>

Spis rysunków

| | |
|---|----|
| Rys. 1: Przykład pól exif wraz z ich zawartością..... | 12 |
| Rys. 2: Wpływ zmiany jasności obrazu na jego histogram szarości [1]..... | 13 |
| Rys. 3: Wpływ zmiany kontrastu obrazu na jego histogram szarości [1]..... | 14 |
| Rys. 4: Schemat podziału klas na moduły wzorca MVC..... | 22 |
| Rys. 5: Fragment widoku programu przedstawiający zawartość wybranych pól EXIF..... | 25 |
| Rys. 6: Widok programu przedstawiający zawartość wszystkich pól EXIF znalezione w obrazie..... | 25 |
| Rys. 7: Przykładowy histogram sum barw stworzony przez program..... | 27 |
| Rys. 8: Przykładowy histogram barw RGB stworzony przez program..... | 28 |
| Rys. 9: Histogram przedstawiający różnice histogramów sum barw..... | 29 |
| Rys. 10: Obraz przedstawiający rozmieszczenie pikseli, których kolor jest zmieniany..... | 31 |
| Rys. 11: Schemat przedstawiający sąsiadów na podstawie kolorów których wyliczany jest nowy kolor dla piksela..... | 31 |
| Rys. 12: Znak zabezpieczający..... | 34 |
| Rys. 13: Zabezpieczenia wielkości obrazu i znak zabezpieczający..... | 37 |
| Rys. 14: Oryginalne zdjęcie przed dokonaniem na nim zabezpieczeń..... | 38 |
| Rys. 15: To samo zdjęcie po dokonaniu zabezpieczeń..... | 38 |
| Rys. 16: Menu główne programu..... | 41 |
| Rys. 17: Okno wyboru pliku..... | 41 |
| Rys. 18: Menu główne po wyborze pliku obrazu..... | 42 |
| Rys. 19: Okno zawierające informacje exif i histogramy obrazu..... | 43 |
| Rys. 20: Okno przedstawiające wszystkie informacje exif umieszczone w pliku z obrazem..... | 44 |
| Rys. 21: Widok zabezpieczenia obrazu..... | 45 |
| Rys. 22: Widok weryfikacji obrazu..... | 46 |
| Rys. 23: Zabezpieczone zdjęcie testowe numer 1..... | 47 |
| Rys. 24: Zabezpieczone zdjęcie które uległo edycji..... | 47 |
| Rys. 25: Widok przedstawiający wyniki badań edytowanego zdjęcia testowego numer 1..... | 48 |
| Rys. 26: Zabezpieczone zdjęcie testowe numer 2..... | 52 |
| Rys. 27: Zabezpieczone zdjęcie testowe które uległo edycji..... | 52 |
| Rys. 28: Okno weryfikacji obrazu dla domyślnej wartości dokładności równej 3..... | 53 |
| Rys. 29: Wynik weryfikacji obrazu dla domyślnej wartości dokładności wynoszącej 3..... | 54 |
| Rys. 30: Wynik weryfikacji obrazu dla wartości dokładności wynoszącej 4..... | 54 |
| Rys. 31: Wynik weryfikacji obrazu dla wartości dokładności wynoszącej 2..... | 55 |
| Rys. 32: Zabezpieczone zdjęcie testowe numer 3..... | 56 |
| Rys. 33: Zabezpieczone zdjęcie które uległo edycji..... | 56 |
| Rys. 34: Wynik weryfikacji przy dokładności sprawdzania równej 6..... | 57 |
| Rys. 35: Wynik weryfikacji dla dokładności równej 6..... | 59 |
| Rys. 36: Zabezpieczone zdjęcie testowe numer 4. Wielkość 3456 x 2592 piksele | 60 |

| | |
|--|----|
| Rys. 37: Powiększone zdjęcie testowe numer 4. Wielkość 4700 x 3525 pikseli... | 60 |
| Rys. 38: Wyniki badań nad obrazem powiększonym z użyciem domyślnej dokładności weryfikacji równej 3..... | 61 |
| Rys. 39: Wynik badań tego samego obrazu dla wartości dokładności równej 15. | 61 |
| Rys. 40: Zabezpieczony obraz pomniejszony. Wielkość 2200 x 1650 pikseli..... | 63 |
| Rys. 41: Wynik badań nad obrazem pomniejszonym z użyciem domyślnej dokładności weryfikacji równej 3..... | 64 |
| Rys. 42: Wynik badań nad obrazem pomniejszonym z użyciem dokładności weryfikacji równej 15..... | 64 |

Opis dołączonej płyty CD

Zawartość folderów:

1. tress – folder zawiera niniejszą pracę zapisaną w formacie PDF,
2. src – pliki źródłowe stworzonej aplikacji,
3. ProgramImages – folder zawierający pliki obrazów wykorzystywanych przez program do działania,
4. Zdjecia_Oryginalne – folder zawierający przykładowe, autentyczne, niezabezpieczone obrazy,
5. Zdjecia_Edytowane – folder zawierający obrazy będące wynikiem wprowadzania zabezpieczeń na obrazach autentycznych oraz obrazy, które uległy edycji.