



UNIVERZITET U NIŠU  
ELEKTRONSKI FAKULTET



## **Steganografija na primeru FFT algoritma**

Seminarski rad

Studijski program: Računarstvo i informatika

Modul: Softversko inženjerstvo

Mentor:

Prof. dr Bratislav Predić

Student:

Miljana Zdravković 1611

Niš, Septembar 2024.

## Sadržaj

Uvod .....	2
Steganografija.....	3
Važnost upotrebe steganografije .....	3
Istorija steganografije.....	3
Algoritmi i tehnike steganografije.....	4
FFT algoritam.....	5
Šta je Furijeova transformacija?.....	5
Kako se koristi u steganografiji? .....	5
Faktor modifikacije .....	6
Faktor modifikacije u steganografiji .....	6
Kako faktor modifikacije funkcioniše .....	7
Primer u kontekstu slike .....	7
Kombinacija steganografije sa kriptografijom .....	7
Aplikacija .....	9
Kratak opis korišćenih tehnologija .....	9
Implementacija .....	10
Funkcionalnosti aplikacije.....	14
Učitavanje fajla .....	14
Izbor faktora modifikacije.....	15
Ekstrakcija sakrivene slike .....	16
Rezultati .....	16
Ispitivanje za različite vrednosti faktora tokom sakrivanja.....	16
Faktor modifikacije 0.50 .....	17
Faktor modifikacije 0.20 .....	17
Faktor modifikacije 0.10 .....	18
Faktor modifikacije 0.05 .....	18
Ispitivanje za različite vrednosti faktora tokom izvlačenja sakrivenih podataka .....	19
Faktor modifikacije 0.50 .....	19
Faktor modifikacije 0.20 .....	19
Faktor modifikacije 0.10 .....	20
Faktor modifikacije 0.05 .....	20
Zaključak .....	21
Manji faktor modifikacije .....	21

Veći faktor modifikacije .....	21
Ključni faktori uticaja .....	21
Literatura .....	22

## Uvod

U ovom radu biće objašnjena upotreba steganografije za skrivanje podataka unutar bitmape odnosno biće prikazano kako se može sakriti slika unutar druge slike tako da se ona manje ili više naruši. Step en narušenja zavisi od faktora modifikacije. Algoritam koji je korišćen u ovu svrhu jeste FFT - Fast Fourier Transform algoritam.

**Steganografija** je važna tehnika u oblasti informacione sigurnosti koja se koristi za skrivanje informacija unutar drugih podataka na način koji otežava njihovo otkrivanje. Njena primarna funkcija nije da enkriptuje podatke kako bi ih učinila nečitljivima, već da sakrije samu činjenicu da podaci postoje. Ova tehnika ima različite primene, uključujući sigurne komunikacije, zaštitu autorskih prava, te skrivanje podataka u medijima poput slika, videa, zvuka i tekstualnih dokumenata.

## Steganografija

Steganografija je praksa skrivanja poruka, podataka ili informacija unutar drugih nepovezanih podataka ili medija kako bi se održala tajnost prisutnosti komunikacije. Ovo može uključivati skrivanje teksta, slike, zvuka ili bilo kojeg drugog tipa podataka unutar drugih medija, poput slike, zvuka ili videa.

### Važnost upotrebe steganografije

1. **Sigurna komunikacija:** Steganografija omogućava prenos osjetljivih informacija na način koji je manje podložan otkrivanju. Ovaj pristup je posebno koristan u situacijama kada sama činjenica prenosa osjetljivih informacija može izazvati sumnju ili represiju.
2. **Zaštita intelektualne svojine:** Korišćenjem steganografskih tehnika, vlasnici autorskih prava mogu sakriti informacije o pravima ili druge važne podatke unutar digitalnih medija, što pomaže u identifikaciji i zaštiti njihove imovine.
3. **Zaštita podataka:** U kombinaciji s kriptografijom, steganografija može pružiti dodatni sloj zaštite, čineći podatke praktično neprimjetnim za neželjene strane.
4. **Prevenција cenzure:** U zemljama s visokim nivoom cenzure, steganografija se može koristiti za skrivanje poruka unutar benignih datoteka, što omogućava slobodnu komunikaciju bez rizika od otkrivanja.

### Istorija steganografije

- **Antičko doba:** Steganografija se koristi vekovima, čak hiljadama godina. Primer iz antičke Grčke uključuje pisanje poruka na drvenim pločama, koje bi potom bile prekrivene voskom. Na taj način, ploča bi izgledala prazno, a stvarna poruka bi bila skrivena ispod voska.
- **Srednji vek:** U srednjem veku, poruke su se sakrivale unutar umetničkih dela ili su se koristili specijalni kodovi i skripte koje bi bile vidljive samo pod određenim uslovima (npr. upotrebom određene hemikalije).
- **Drugi svetski rat:** Tokom Drugog svetskog rata, steganografija je korišćena za sakrivanje poruka u mikrotekstu, mikrodotovima i na slične načine kako bi se izbegla detekcija od strane neprijatelja.

- **Digitalna era:** Sa razvojem digitalnih tehnologija, steganografija je evoluirala u sofisticirane tehnike sakrivanja podataka unutar digitalnih medija. Savremena digitalna steganografija uključuje skrivanje podataka unutar slika, video zapisa, audio fajlova i drugih digitalnih medija.

## Algoritmi i tehnike steganografije

1. **Least Significant Bit (LSB) metoda:** Ova metoda uključuje zamenu najmanje značajnog bita (LSB) u svakom bajtu podataka koji se skriva s odgovarajućim bitom podataka koji se uvode. Ova tehnika se često koristi za skrivanje podataka unutar slika.
2. **Frequency Domain Techniques:** Ove tehnike koriste transformacije poput brze Fourierove transformacije (FFT) kako bi se skrila informacija u frekvencijskom prostoru slike ili zvuka.
3. **Spread Spectrum Metode:** Ova tehnika uključuje distribuciju informacija širom širokog spektra kako bi se smanjila osetljivost na smetnje i otkrivanje.
4. **Transformacija slike:** Ova tehnika koristi različite transformacije slike, poput transformacija boje ili promjene veličine, kako bi se sakrila informacija.
5. **Text Steganography:** Skrivanje teksta unutar drugog teksta ili dokumenta, gde se koriste različite metode za skrivanje informacija.
6. **Audio Steganography:** Skrivanje informacija unutar zvučnih datoteka, koristeći slične metode kao i slike, poput LSB metode ili manipulacije frekvencijama.
7. **Video Steganography:** Slično kao i audio steganografija, ali primenjena na video datoteke.
8. **Cryptographic Steganography:** Kombinacija steganografije i kriptografije za dodatni sloj sigurnosti.

## FFT algoritam

### Šta je Furijeova transformacija?

U izvornom obliku, Furijeova teorema kaže da proizvoljna funkcija  $f(x)$  sa prostornim periodom  $\lambda$  može da se predstavi kao zbir sinusoida čije su talasne dužine (tj. prostorni periodi) date proizvodom sa brojem koji se, počevši od  $\lambda$ , smanjuje u skladu sa povećanjem deljenika za

ceo broj (tj. ako je početni period  $\lambda$ , talasne dužine sinusoida su  $\lambda$ ,  $\lambda/2$ ,  $\lambda/3$ , itd.), dok se odgovarajući rasponi ovih sinusoida takođe smanjuju u istoj srazmeri. Oblici Furijeovog postupka koji se koriste danas su složeniji, ali opšti princip da se učestalost i raspon Furijeovih sinusoida smanjuju u istoj razmeri i dalje važi.

Moderna definicija je opštijeg karaktera, i kaže da se, u načelu, matematička funkcija  $f(x)$  proizvoljnog oblika, data u prostornom ili vremenskom domenu - tj. sa promenljivom koja se menja u prostoru ili vremenu, u odnosu na  $x$  dato u prostornim ili vremenskim jedinicama - može proizvesti sabiranjem niza sinusoidnih funkcija opadajućih raspona i učestalosti. Ako se

ovaj niz sinusoidnih funkcija predstavi kao posebna funkcija, u kojoj je učestalost  $v$  svakog od ovih sinusoida data na vodoravnoj skali, i sa odgovarajućim rasponima za svaku učestalost datim na uspravnoj skali, dobija se funkcija  $F(v)$  u domenu učestalosti, koja opisuje raspored učestalosti ovih sinusoida i njihov raspon.

Drugim rečima, primenom Furijeovog postupka na funkciju  $f(x)$  datu u vremenskom ili prostornom domenu, dobija se funkcija  $F(v)$  u domenu učestalosti. Dva osnovna rezultata Furijeovog postupka, u pogledu oblika funkcije u domenu učestalosti - koja se može nazvati

Furijeova funkcija - su *Furijeov niz*, u slučaju funkcija u kojima je razmak između periodičnih promena konačan, i *Furijeova transformacija*, u slučaju funkcija u kojima je razmak do sledeće promene beskonačan, tj. koje opisuju samo jednu promenu.

Funkcija domena učestalosti  $F(v)$  i odgovarajuća funkcija u prostornom ili vremenskom domenu  $f(x)$ , čine Furijeov par, tj.  $F(v) = f(x)$ , gde je  $F(v)$  funkcija učestalosti  $v$ , označava Furijeovu transformaciju, i  $f(x)$  funkciju promenljive  $x$  u prostornom ili vremenskom domenu. Funkcija  $f(x)$  je obrnuta Furijeova transformacija funkcije  $F(v)$ , tj.  $f(x) = {}^{-1}F(v)$ .

### Kako se koristi u steganografiji?

Frequential Domain Techniques, ili tehnike u frekvencijskom domenu, često uključuju primenu različitih matematičkih transformacija, pri čemu svega brže Fourierove transformacije (FFT) ili srodne tehnike, kako bi se slika, zvuk ili drugi podaci predstavili u frekvencijskom prostoru umesto u vremenskom ili prostornom domenu.

Evo kako brza Fourierova transformacija (FFT) funkcioniра i kako se može primieniti u kontekstu steganografije:

**1. Brza Fourierova Transformacija (FFT):**

- FFT je matematički algoritam koji transformiše podatke iz vremenskog domena u frekvencijski domen. Na primer, ako imate vremenski signal, FFT će ga pretvoriti u spektar frekvencija.

**2. Predstavljanje u frekvencijskom domenu:**

- Nakon primene FFT-a, podaci se predstavljaju u frekvencijskom domenu pomoću kompleksnih brojeva. Komponente kompleksnih brojeva predstavljaju različite frekvencije u originalnom signalu.

**3. Manipulacija frekvencijskim komponentama:**

- Frekvencijske komponente, tj. vrednosti dobijene primenom FFT-a, mogu se menjati kako bi se skrili podaci. To uključuje dodavanje, oduzimanje ili zamenu određenih vrednosti.

**4. Inverzna Fourierova Transformacija (Inverse FFT):**

- Nakon što su podaci modulirani u frekvencijskom domenu, primenjuje se inverzna Fourierova transformacija kako bi se dobili modifikovani podaci nazad u vremenskom domenu.

**5. Steganografsko umetanje podataka:**

- Modifikovane frekvencijske komponente, koje sadrže skrivene podatke, mogu se umetnuti u originalne podatke, poput slike, zvuka ili videozapisa.

Primer korištenja frekvencijskih tehnika u steganografiji može uključivati zamenu određenih frekvencijskih komponenti spektra s novim podacima. Ova zamena mora biti dovoljno suptilna

kako bi bila neprimetna ljudskom oku ili uhu, ali istovremeno dovoljno značajna kako bi omogućila efikasno skrivanje informacija.

Važno je napomenuti da postoje i druge transformacije u frekvencijskom domenu, poput diskretne kosinusne transformacije (DCT), koje se često koriste u steganografskim aplikacijama, zavisno o vrsti podataka koje se skrivaju.

## Faktor modifikacije

**Faktor modifikacije (Modification Factor)** se često koristi u kontekstu steganografije i sličnih tehnika u kojima se podaci skrivaju unutar drugih podataka, poput slike. U ovom kontekstu, faktor modifikacije se odnosi na količinu promene koja se primenjuje na nosioce podataka (npr. piksele slike) kako bi se umetnuli skriveni podaci.

## Faktor modifikacije u steganografiji

- **Definicija:** Faktor modifikacije određuje koliko će izvorni podaci biti promenjeni prilikom umetanja skrivenih informacija. Ovaj faktor je ključan za balansiranje između neprimetnosti umetnutih podataka i njihove otpornosti na otkrivanje ili gubitak.
- **Vrednost:** Faktor modifikacije može imati različite vrednosti, zavisno od algoritma. Viši faktor obično znači da su promene u originalnim podacima (npr. pikselima slike) vidljivije, dok niži faktor znači da su promene suptilnije, ali možda i manje otporne na smetnje ili gubitak.

## Kako faktor modifikacije funkcioniše

### 1. Visoki faktor modifikacije:

- Umetnuti podaci su jasniji i lakše ih je kasnije izvući.
- Može uzrokovati vidljive promene u originalnoj slici, čime se povećava rizik od otkrivanja skrivenih podataka.
- Koristi se kada je otpornost na gubitak podataka važnija od skrivanja.

### 2. Niski faktor modifikacije:

- Promene u originalnoj slici su manje primetne, što omogućava bolje skrivene podatke.
- Skriveni podaci mogu biti teže izvući i mogu biti podložniji gubitku ili degradaciji.
- Koristi se kada je neprimetnost važnija od otpornosti na gubitak podataka.

## Primer u kontekstu slike

Ako umetnete crno-belu sliku u sliku u boji, faktor modifikacije može kontrolisati koliko će RGB vrednosti originalne slike biti promenjene. Na primer:

- **Faktor modifikacije 0.1:** Svaka RGB vrednost piksela se promeni za 10% vrednosti umetnute slike. Ovo može biti teško primetno golim okom, ali podaci mogu biti ranjiviji.
- **Faktor modifikacije 1.0:** RGB vrednosti se menjaju direktno u skladu sa umetnutom slikom, što može uzrokovati jasne promene, ali i bolje očuvanje umetnutih podataka.

## Kombinacija steganografije sa kriptografijom

Često se koristi kako bi se stvorila složenija i sigurnija metoda za skrivanje i zaštitu podataka. Nekoliko načina kako se ove dvije tehnike mogu kombinirati:

### 1. Kriptovanje pre steganografskog procesa:

- Prije nego što se podaci umetnu u medij pomoću steganografije, koriste se kriptografski algoritmi za enkripciju podataka. Ovo osigurava da, čak i ako se otkrije prisutnost skrivenih podataka, oni ostaju nečitljivi bez odgovarajućeg ključa za dekripciju.

### 2. Kombinacija ključeva:

- Kriptografski ključ može se koristiti zajedno s ključem za steganografski postupak. Bez oba ključa, bilo bi izazovno pristupiti i dekriptirati originalne podatke.

### 3. Digitalni potpisi:

- Digitalni potpisi mogu se koristiti za potvrdu identiteta pošiljaoca podataka i osiguranje autentičnosti. Ovi potpisi mogu biti skriveni unutar steganografskih podataka.

### 4. Temporalna kriptografija:

- Umjesto stalnog kriptiranja, kriptografski ključ može se generisati dinamički ili periodično, čime se otežava dekripcija bez odgovarajućeg ključa u tačnom vremenskom trenutku.

### 5. Korišćenje kriptografskih hash funkcija:

- Kriptografske hash funkcije mogu stvoriti jedinstveni hash originalnih podataka prije steganografskog postupka, pružajući proveru integriteta.

### 6. Višestruki slojevi enkripcije i steganografije:



## Steganografija (FFT)

- Kombinovanje više različitih metoda steganografije i kriptografije može stvoriti višeslojnu sigurnosnu obranu. Svaki sloj dodaje dodatni nivo zaštite.

Kombinacija ovih tehnika može pridonijeti povećanju sigurnosti i otežati pokušaje neovlaštenog pristupa podacima. Međutim, uvijek treba razmotriti specifične zahteve i kontekst primene kako bi se odabrala odgovarajuća kombinacija steganografskih i kriptografskih tehnika.

## Aplikacija

Aplikacija je napravljena tako da simulira skrivanje jedne slike u drugu, sa proizvoljnim stepenom modifikacije. Korisnik prvo bira slike koje želi da koristi, zatim se prikazuje proces steganografije, a moguće je uraditi i inverzni postupak i prikazati skrivenu informaciju (sliku).

### Kratak opis korišćenih tehnologija

WinForms (Windows Forms) je GUI framework koji je deo .NET-a i koristi se za izgradnju desktop aplikacija za Windows operativni sistem. Pruža jednostavan način za kreiranje aplikacija sa grafičkim korisničkim interfejsom koristeći kontrole poput dugmadi, tekstualnih polja, lista, i drugih elemenata interfejsa.

**MathNet.Numerics** paket, koji je popularan u .NET ekosistemu za napredne matematičke i numeričke operacije. Konkretno, namespace **MathNet.Numerics**. **IntegralTransforms** omogućava rad sa integralnim transformacijama, poput Fourierove transformacije, Kosinusne transformacije i Wavelet transformacije. Ove transformacije se široko koriste u obradi signala, analizi podataka i mnogim naučnim primenama.

Namespace **System.Drawing.Imaging** pruža podršku za rad sa slikama u .NET aplikacijama, fokusirajući se na napredne opcije rada sa slikovnim formatima, paletama, kompresijama, i obradom piksela. Ova biblioteka je korisna kada se radi sa formatima poput JPEG, PNG, BMP, i drugim grafičkim formatima. Glavne mogućnosti:

- **ImageCodecInfo:** Omogućava informacije o dostupnim kodirnim mehanizmima za različite formate slika.
- **EncoderParameters:** Koristi se za konfiguraciju parametara prilikom čuvanja slike, na primer podešavanje nivoa kompresije za JPEG format.
- **PixelFormat:** Omogućava rad sa različitim formatima piksela (24-bita, 32-bita, itd.), što je ključno u obradi slika na niskom nivou.

**System.Numerics** namespace sadrži tipove podataka koji su korisni za rad sa kompleksnim brojevima i vektorskim operacijama, ključnim za numeričku i grafičku obradu. Ključni tipovi:

- **Complex:** Predstavlja kompleksan broj, koji se koristi u mnogim matematičkim primenama, kao što su Fourierove transformacije.
- **Vector:** Pruža podršku za rad sa vektorima, uključujući SIMD (Single Instruction, Multiple Data) optimizovane operacije za bolje performanse u radu sa velikim skupovima podataka.

## Implementacija

```

2 references
private Bitmap ProcessImages(Bitmap coverBitmap, Bitmap secretBitmap, double alpha)
{
    var coverArray = BitmapToComplexArray(coverBitmap);
    var secretArray = BitmapToComplexArray(secretBitmap);

    var coverFFT = (Complex[])coverArray.Clone();
    var secretFFT = (Complex[])secretArray.Clone();

    Fourier.Forward(coverFFT, FourierOptions.Matlab);
    Fourier.Forward(secretFFT, FourierOptions.Matlab);

    var resultFFT = new Complex[coverFFT.Length];
    for (int i = 0; i < coverFFT.Length; i++)
    {
        resultFFT[i] = coverFFT[i] + alpha * secretFFT[i];
    }

    Fourier.Inverse(resultFFT, FourierOptions.Matlab);
    var resultArray = ComplexArrayToBitmap(resultFFT);
    return DoubleArrayToBitmap(resultArray, coverBitmap.Width, coverBitmap.Height);
}

```

`BitmapToComplexArray(coverBitmap)` i `BitmapToComplexArray(secretBitmap)`: Ovi pozivi konvertuju slike u nizove kompleksnih brojeva. Svaki pixel slike se prevodi u kompleksni broj gde je realni deo intenzitet piksela, a imaginarni deo je nula.

`Fourier.Forward`: Izvodi Fourierovu transformaciju na nizovima kompleksnih brojeva (`coverFFT` i `secretFFT`). Ovo pretvara slike iz domena prostora (slike) u domen frekvencije (Fourierov spektar).

`resultFFT[i] = coverFFT[i] + alpha * secretFFT[i]`: Kombinacija dva Fourierova spektra. Ovdje se dodaje modifikovani spektar skrivene slike (pomnožen sa `alpha`) na spektar pokrovne slike.

`Fourier.Inverse(resultFFT, FourierOptions.Matlab)`: Izvodi inverznu Fourierovu transformaciju na rezultatnom spektru kako bi se dobila modifikovana slika u domenu prostora.

`ComplexArrayToBitmap(resultFFT)`: Konvertuje rezultat Fourierove transformacije nazad u niz double vrednosti, gde je svaka vrednost intenzitet piksela.

`DoubleArrayToBitmap(resultArray, coverBitmap.Width, coverBitmap.Height)`: Pretvara niz double vrednosti u `Bitmap` objekat koji predstavlja rezultat kombinovanja slika.

```

4 references
private Complex[] BitmapToComplexArray(Bitmap bitmap)
{
    var width = bitmap.Width;
    var height = bitmap.Height;
    var array = new Complex[width * height];

    for (int y = 0; y < height; y++)
    {
        for (int x = 0; x < width; x++)
        {
            var pixel = bitmap.GetPixel(x, y);
            double value = pixel.R;
            array[y * width + x] = new Complex(value, 0);
        }
    }

    return array;
}

```

BitmapToComplexArray: Ovaj metod konvertuje Bitmap sliku u niz kompleksnih brojeva. Svaki pixel se uzima iz slike, a samo crvena komponenta (`pixel.R`) se koristi kao intenzitet. U ovom slučaju, pretpostavlja se da je slika u nijansama sive, pa je sve u istoj komponenti (`R`, `G`, `B` su isti).

```

2 references
private Bitmap DoubleArrayToBitmap(double[] array, int width, int height)
{
    var bitmap = new Bitmap(width, height, PixelFormat.Format24bppRgb);

    for (int y = 0; y < height; y++)
    {
        for (int x = 0; x < width; x++)
        {
            var value = (int)Math.Clamp(array[y * width + x], 0, 255);
            var color = Color.FromArgb(value, value, value);
            bitmap.SetPixel(x, y, color);
        }
    }

    return bitmap;
}

```

DoubleArrayToBitmap: Pretvara niz double vrednosti u Bitmap. Svaka vrednost u nizu predstavlja intenzitet piksela. Vrednosti su ograničene na opseg od 0 do 255 pomoću `Math.Clamp`, a svaki pixel se postavlja sa istom vrednošću za crvenu, zelenu i plavu komponentu (pravljenje slike u nijansama sive).

2 references

```
private double[] ComplexArrayToBitmap(Complex[] array)
{
    var doubleArray = new double[array.Length];

    for (int i = 0; i < array.Length; i++)
    {
        doubleArray[i] = Math.Clamp(array[i].Real, 0, 255);
    }

    return doubleArray;
}
```

`ComplexArrayToBitmap`: Pretvara niz kompleksnih brojeva u niz double vrednosti. U ovom slučaju, samo se realni deo kompleksnih brojeva koristi kao intenzitet piksela, a vrednosti se ograničavaju na opseg od 0 do 255 pomoću `Math.Clamp`.

1 reference

```
public Bitmap ExtractImageFromImage(Bitmap coverBitmap, Bitmap modifiedBitmap, double alpha)
{
    var coverArray = BitmapToComplexArray(coverBitmap);
    var modifiedArray = BitmapToComplexArray(modifiedBitmap);

    var coverFFT = (Complex[])coverArray.Clone();
    var modifiedFFT = (Complex[])modifiedArray.Clone();

    Fourier.Forward(coverFFT, FourierOptions.Matlab);
    Fourier.Forward(modifiedFFT, FourierOptions.Matlab);

    // Ekstrakcija skrivene slike
    var secretFFT = new Complex[coverFFT.Length];
    for (int i = 0; i < coverFFT.Length; i++)
    {
        secretFFT[i] = (modifiedFFT[i] - coverFFT[i]) / alpha;
    }

    Fourier.Inverse(secretFFT, FourierOptions.Matlab);

    var extractedArray = ComplexArrayToBitmap(secretFFT);
    return DoubleArrayToBitmap(extractedArray, coverBitmap.Width, coverBitmap.Height);
}
```

- Konverzija slika u niz kompleksnih brojeva pomoću `BitmapToComplexArray`. To se radi tako da se svaka vrednost piksela (u ovom slučaju, pretpostavljamo da je slika crno-bela, tj. grayscale) pretvara u realni deo kompleksnog broja, dok imaginarni deo ostaje nula. Ova konverzija je potrebna da bismo kasnije mogli da primenimo Fourierovu transformaciju.

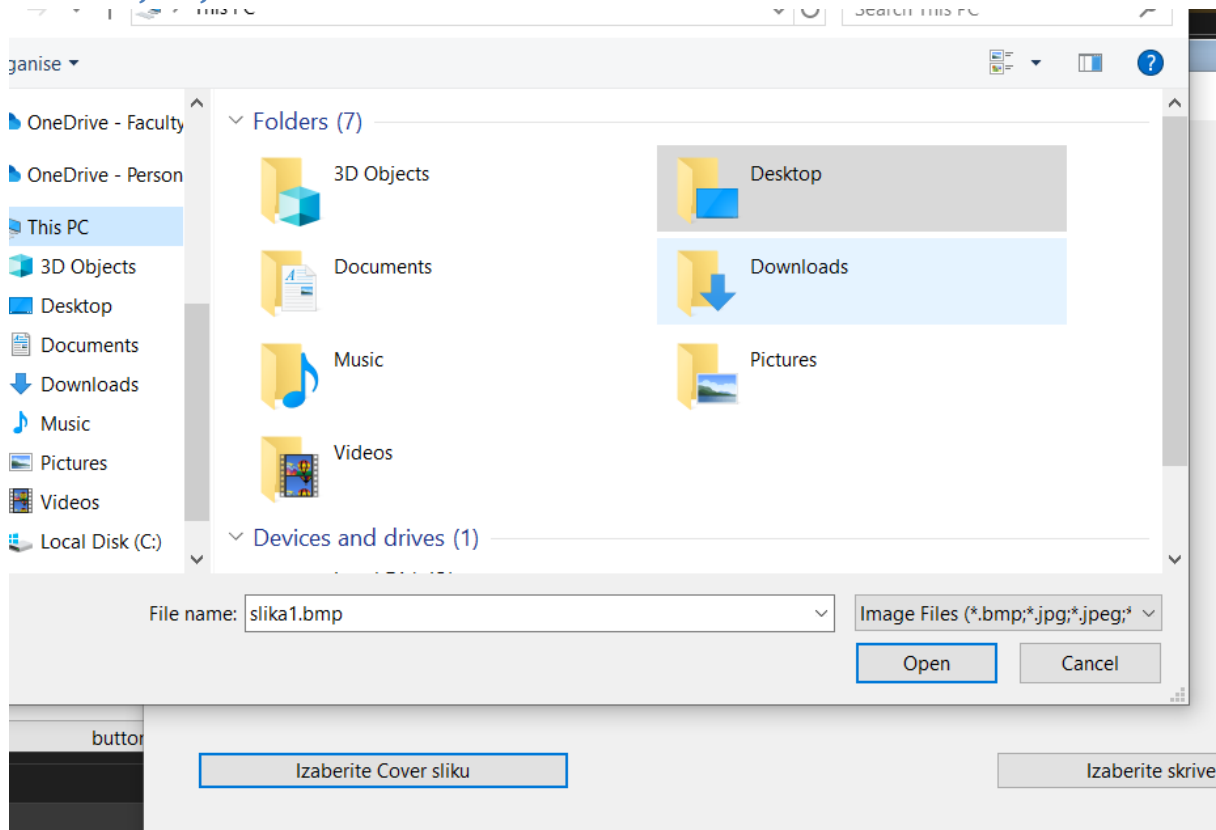
- Zatim primenjujemo Fourierovu transformaciju na oba niza (za `cover` i `modified` sliku) pomoću metode `Fourier.Forward`. Fourierova transformacija pretvara signal iz prostora u domen frekvencije, omogućavajući nam da manipuliramo frekvencijskim komponentama slike.
- Nakon ove transformacije, dobijamo nizove `coverFFT` i `modifiedFFT` koji sadrže Fourierove spektre obe slike.
- **Izračunavanje skrivene slike:**
  - Pretpostavljamo da je skrivena slika dodata u frekvencijski spektar pokrovne slike pomoću formule:
$$\text{modifiedFFT}[i] = \text{coverFFT}[i] + \alpha \cdot \text{secretFFT}[i]$$
  - Da bismo sada izvadili skrivenu sliku, moramo preurediti ovu formulu i izraziti `secretFFT[i]`:
$$\text{secretFFT}[i] = \frac{\text{modifiedFFT}[i] - \text{coverFFT}[i]}{\alpha}$$
  - Ova formula se koristi da se izračunaju frekvencijske komponente skrivene slike (`secretFFT[i]`).
- Nakon što smo dobili frekvencijski spektar skrivene slike, koristimo **inverznu Fourierovu transformaciju** da vratimo skrivenu sliku iz domena frekvencije u domen prostora. Time dobijamo rekonstrukciju skrivene slike u originalnom obliku.
- **ComplexArrayToBitmap**: Ova funkcija uzima rezultat inverzne Fourierove transformacije (koji je niz kompleksnih brojeva) i pretvara ga u niz realnih brojeva (piksela slike).
- **DoubleArrayToBitmap**: Ova funkcija pretvara niz piksela u sliku (tipa `Bitmap`) koju možemo prikazati.

Ovaj kod koristi Fourierovu transformaciju da iz modifikovane slike (koja sadrži skrivenu informaciju) ekstrahuje skriveni sadržaj, koristeći pristup baziran na manipulaciji frekvencijskim komponentama. Rezultat je slika koja je bila skrivena u modifikovanoj slici, a faktor `alpha` je korišćen kako bi se pravilno izračunale te komponente.

## Funkcionalnosti aplikacije

U nastavku je dato objašnjenje rada aplikacije.

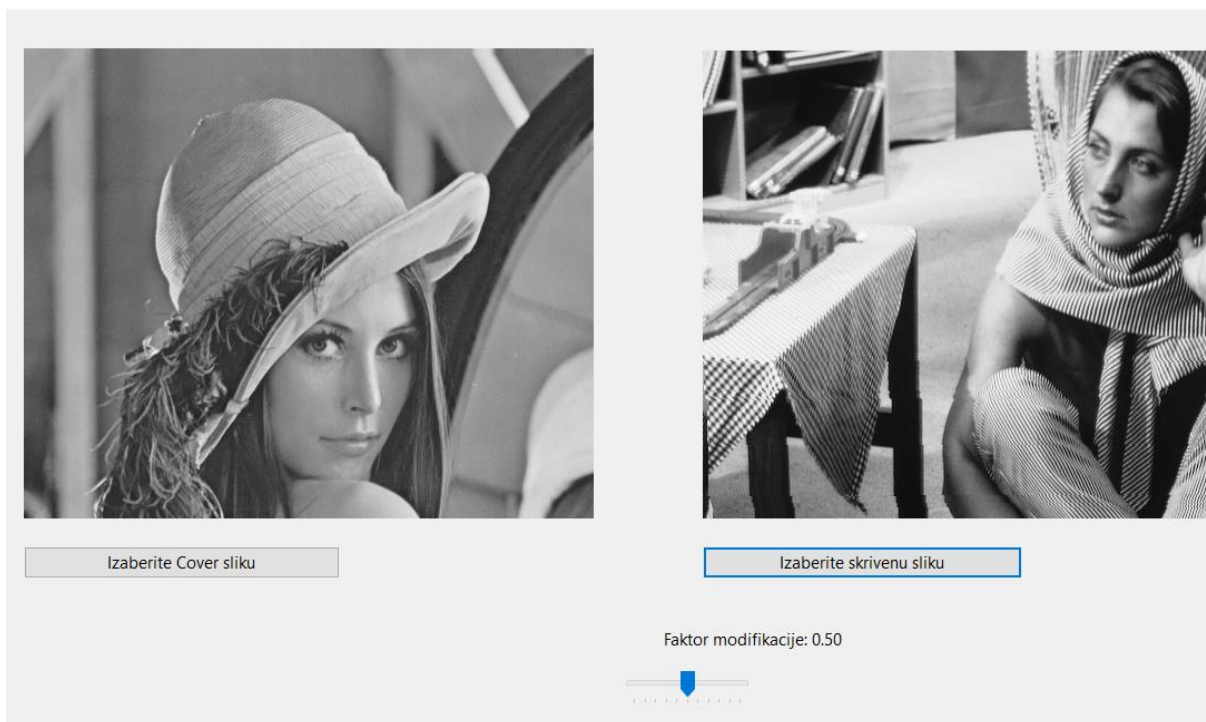
### Učitavanje fajla



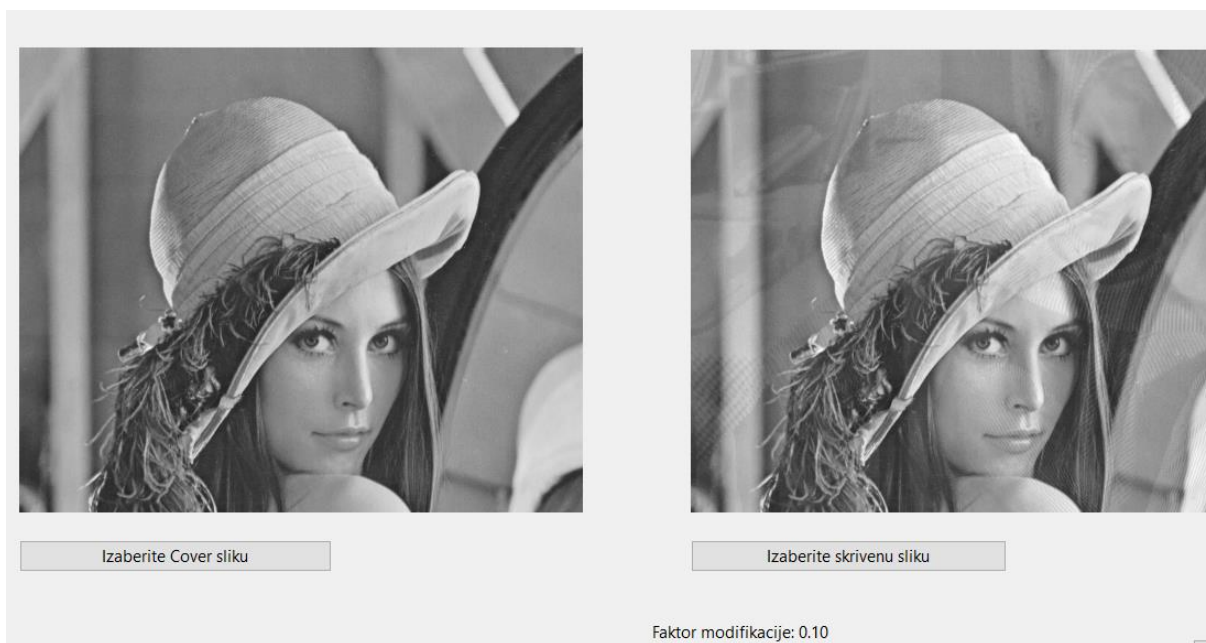
Klikom na izaberi Cover sliku I izaberi skrivenu sliku otvara se izbor datoteka sa fajl sistema, potrebno je odabrati jednu pa drugu sliku.



## Izbor faktora modifikacije



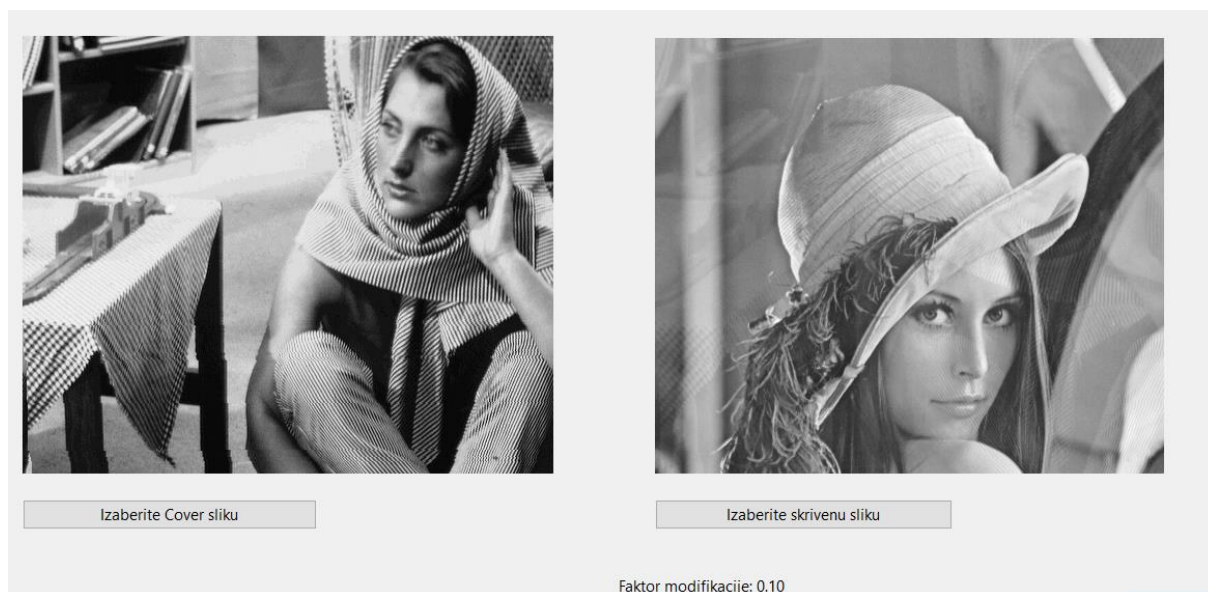
Nakon izbora slika možemo izabrati factor modifikacije što će na desnoj strani prikazati modificovanu sliku. Difoltna vrednost faktora modifikacije je 0.50 i može se menjati.



Dakle, na desnoj strani je prikazana modificovana slika nakon procesa steganografije sa faktorom modifikacije 0.10.



## Ekstrakcija sakrivene slike



Klikom na izvuci sakrivenu sliku možemo videti da je iz slike desno izvučena sakrivena slika koju smo umetnuli na početku.

## Rezultati

### Ispitivanje za različite vrednosti faktora tokom sakrivanja

Na sledećim primerima je prikazano kako faktor modifikacije utiče na promenu kvaliteta originalne (Cover) slike:

### Faktor modifikacije 0.50



Izaberite Cover sliku



Izaberite skrivenu sliku

Faktor modifikacije: 0.50

### Faktor modifikacije 0.20




Izaberite Cover sliku




Izaberite skrivenu sliku

Faktor modifikacije: 0.20

### Faktor modifikacije 0.10




Izaberite Cover sliku




Izaberite skrivenu sliku

Faktor modifikacije: 0.10

### Faktor modifikacije 0.05



Izaberite Cover sliku



Izaberite skrivenu sliku

Faktor modifikacije: 0.05

## Ispitivanje za različite vrednosti faktora tokom izvlačenja sakrivenih podataka

### Faktor modifikacije 0.50



Izaberite Cover sliku



Izaberite skrivenu sliku

Faktor modifikacije: 0.50

### Faktor modifikacije 0.20



Izaberite Cover sliku



Izaberite skrivenu sliku

Faktor modifikacije: 0.20



## Faktor modifikacije 0.10



Izaberite Cover sliku



Izaberite skrivenu sliku

Faktor modifikacije: 0.10

## Faktor modifikacije 0.05



Izaberite Cover sliku



Izaberite skrivenu sliku

Faktor modifikacije: 0.05

## Zaključak

**Izvučena slika** u steganografiji zavisi od **faktora modifikacije** jer on direktno utiče na količinu promena koje pravimo na pikselima originalne slike da bismo sakrili podatke. Evo kako to funkcioniše:

### Manji faktor modifikacije

- Kada koristimo **manji faktor modifikacije**, npr. menjamo samo najmanje značajan bit svakog piksela (LSB metoda), originalna slika se minimalno menja.
- Ovo znači da je šum u originalnoj slici gotovo neprimetan, ali takođe se skriva manja količina podataka, što može ograničiti kvalitet izvučene slike.
- **Kvalitet izvučene slike** takođe može biti lošiji jer promenjeni bitovi ne prenose dovoljno informacija da bi sakrili detalje sakrivene slike.

### Veći faktor modifikacije

- Kada koristimo **veći faktor modifikacije**, menjamo više značajnih bitova svakog piksela, što vam omogućava da sakrijete više podataka.
- Međutim, ovde dolazi do problema: **više značajnih bitova znači veću promenu u originalnoj slici**, što uvodi vidljiv šum. Originalna slika postaje iskrivljena, a šum se povećava proporcionalno broju promenjenih bitova.
- **Izvučena slika** (slika koju pokušavate da sakrijete) može imati više šuma, jer informacije koje se sakrivaju postaju manje precizne zbog prekomerne promene piksela u originalnoj slici. To znači da može doći do izobličenja kada pokušate da je izvučete.

### Ključni faktori uticaja

- **Preciznost skrivanja:** Sa manjim faktorom modifikacije, steganografske informacije (sakrivena slika) nisu dovoljno precizno "zapamćene" u originalnoj slici, što rezultira slabijom kvalitetom izvučene slike.
- **Vidljivost šuma:** Sa većim faktorom modifikacije, dolazi do većih promena u originalnoj slici, pa iako možete sakriti više informacija, vaša izvučena slika će imati više šuma i verovatno biti izobličena.

Dakle, manji faktor modifikacije daje manje vidljiv šum u originalnoj slici, ali takođe smanjuje preciznost sakrivene slike kada se izvuče. Veći faktor modifikacije omogućava skrivanje više informacija, ali povećava šum i izobličenje kako originalne tako i sakrivene slike.

Da bi ste optimizovali kvalitet izvučene slike, potrebno je pronaći balans između količine sakrivenih podataka i stepena modifikacije originalne slike.

## Literatura

<https://eeweb.engineering.nyu.edu/~yao/EL5123/SampleData.html>

[https://sr.wikipedia.org/sr-](https://sr.wikipedia.org/sr-el/%D0%A4%D1%83%D1%80%D0%B8%D1%98%D0%B5%D0%BE%D0%B2%D0%B0%D0%BE%D0%BF%D1%82%D0%B8%D0%BA%D0%B0)

[el/%D0%A4%D1%83%D1%80%D0%B8%D1%98%D0%B5%D0%BE%D0%B2%D0%B0%D0%BE%D0%BF%D1%82%D0%B8%D0%BA%D0%B0](https://sr.wikipedia.org/sr-el/%D0%A4%D1%83%D1%80%D0%B8%D1%98%D0%B5%D0%BE%D0%B2%D0%B0%D0%BE%D0%BF%D1%82%D0%B8%D0%BA%D0%B0)

[https://cs.elfak.ni.ac.rs/nastava/pluginfile.php/8039/mod\\_resource/content/1/DF%20-%20Predavanje%206%20-%20Forenzika%20rasterskih%20slika.pdf](https://cs.elfak.ni.ac.rs/nastava/pluginfile.php/8039/mod_resource/content/1/DF%20-%20Predavanje%206%20-%20Forenzika%20rasterskih%20slika.pdf)

<https://www.techtarget.com/searchsecurity/definition/steganography>

<https://www.kaspersky.com/resource-center/definitions/what-is-steganography>

<https://www.garykessler.net/library/steganography.html>

<https://www.okta.com/identity-101/steganography/>