

Sveučilište u Zagrebu

Prirodoslovno-matematički fakultet

Steganografija SVD

Naziv tima:

Glitch u matrici

Članovi tima:

Željka Baća, Nikola Kašnar, Ivana Kristić

Mentor:

prof. dr. sc. Zlatko Drmač

Zagreb, 6. prosinca 2024.

Sadržaj

1	Uvod	2
2	Singular Value Decomposition (SVD)	3
2.1	Motivacija	3
2.2	SVD	3
3	Proposed Watermarking Scheme	5
3.1	Dekompozicija slike	5
3.2	Umetanje vodenog žiga u D matricu	5
3.3	Umetanje vodenog žiga u U matricu	6
3.4	Konačna slika s vodenim žigom	7
4	Eksperimentalni rezultati	8
4.1	Steganografija na slici	8
4.2	Watermarking scheme	15
4.3	Testiranje robustnosti	21
4.4	Photo collage experiment	21
4.4.1	Umetanje žiga u blokove	24
4.4.2	Zamućivanje	27
4.4.3	Rotacija	29
4.4.4	Ostalo	31
4.4.5	JPG format	33
5	Zaključak	37
6	Literatura	38

1 Uvod

Steganografija je tehnika skrivanja informacija unutar drugih, naizgled bezazlenih podataka, kako bi se prikrila prisutnost tajne poruke. Ova metoda se koristi od davnina, a s razvojem digitalnih tehnologija doživjela je značajan napredak, omogućujući sigurno i neprimjetno slanje informacija u digitalnom obliku. U kontekstu digitalne komunikacije, steganografija se često koristi za zaštitu privatnosti i osiguranje tajnosti podataka, čime se smanjuje rizik od otkrivanja osjetljivih informacija.

Jedna od glavnih prednosti steganografije u odnosu na druge metode skrivanja informacija, poput šifriranja, je ta što steganografija ne izaziva sumnju. Dok šifriranje jasno ukazuje da je informacija zaštićena, steganografija skriva samu činjenicu da se informacije razmjenjuju, što je osobito važno u vojnim i špijunskim aplikacijama, gdje je ključno da komunikacija ostane neprimjetna.

U ovom seminaru istražujemo primjenu steganografije u digitalnim medijima, s posebnim naglaskom na umetanje vodenih žigova u slike. Testirat ćemo i istražiti metodu Singular Value Decomposition (SVD), analizirajući njezinu učinkovitost i robusnost u skrivenju informacija. Cilj je istaknuti važnost ovih metoda u zaštiti digitalnih informacija, te prikazati mogućnosti SVD metode u praktičnoj primjeni.

2 Singular Value Decomposition (SVD)

2.1 Motivacija

Singular Value Decomposition (SVD) predstavlja jednu od najvažnijih tehnika u obradi podataka, a njezina široka primjena čini je ključnim alatom u različitim disciplinama. S obzirom na sve veći značaj obrade digitalnih informacija, SVD se ističe svojom sposobnošću da efikasno razlaže matricu na komponente, čime se omogućava pojednostavljanje složenih podataka.

Jedna od ključnih primjena SVD-a je u kompresiji slika, što omogućava smanjenje veličine datoteka bez značajnog gubitka kvalitete. U današnjem svijetu, gdje se suočavamo s velikim količinama vizualnih podataka, učinkovita kompresija slika postaje sve važnija.

Osim toga, SVD se koristi u rekonstrukciji preporučivih sustava, što pomaže u prepoznavanju obrazaca u korisničkim podacima i omogućava precizniju predikciju korisničkih preferencija. U kontekstu preporuka, SVD se koristi za analizu velikih skupova podataka i identifikaciju sličnosti među korisnicima i predmetima.

Dodatno, SVD se koristi u obnovi uništenih slika ljudskih lica, čime se omogućava popravak slika koje su oštećene ili uništene. Ova tehnika također igra važnu ulogu u klasifikaciji ručno pisanih znamenki, što poboljšava učinkovitost prepoznavanja obrazaca.

Iako SVD ima široku primjenu, u ovom seminaru fokusirat ćemo se na umetanje žigova u slike. Umetanje vodenih žigova predstavlja važnu metodu za zaštitu autorskih prava i integritet digitalnih sadržaja. Korištenjem SVD-a za umetanje žigova, istražit ćemo kako se informacije mogu sigurno integrirati u slike bez značajnih promjena u vizuelnoj kvaliteti, čime se doprinosi očuvanju autorskih prava u digitalnom okruženju.

U kontekstu digitalnog vodenog žigovanja, SVD igra ključnu ulogu u osiguravanju robustnosti umetnutih informacija. Ova metoda omogućuje umetanje vodenih žigova koji su otporni na različite vrste napada, čime se osigurava autentičnost i vlasništvo nad digitalnim sadržajem. Također, SVD se koristi u analizi korisničkih podataka u sustavima preporuka, što dodatno naglašava njegovu svestranost.

2.2 SVD

Za danu matricu A dimenzija $m \times n$ i ranga r , singularna vrijednosna dekompozicija omogućuje da se A izrazi kao produkt triju matrica:

$$A = U\Sigma V^*,$$

gdje:

U je unitarna matrica dimenzija $m \times m$, Matrica U sadrži lijeve singularne vektore matrice A kao svoje stupce. Budući da je U unitarna matrica, vrijedi $U^*U = I_m$, gdje je I_m identična matrica dimenzija $m \times m$.

Σ je dijagonalna matrica dimenzija $m \times n$ sa singularnim vrijednostima matrice A na glavnoj dijagonali, Matrica Σ je dijagonalna matrica dimenzija $m \times n$ sa singularnim vrijednostima σ_i na dijagonali:

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0.$$

Singularne vrijednosti σ_i definirane su kao kvadratni korijeni svojstvenih vrijednosti matrice A^*A .

V je unitarna matrica dimenzija $n \times n$, Matrica V sadrži desne singularne vektore matrice A kao svoje stupce i također je unitarna, što znači da $V^*V = I_n$, gdje je I_n identična matrica dimenzija $n \times n$.

V^* označava Hermitski konjugat matrice V (transponiranje i konjugiranje).

Matrica A može se izraziti kao:

$$A = U\Sigma V^*,$$

što znači da se A može rekonstruirati kao suma produkata:

$$A = \sum_{i=1}^r \sigma_i u_i v_i^*,$$

gdje su u_i lijevi singularni vektori (stupci matrice U), v_i desni singularni vektori (stupci matrice V), a σ_i singularne vrijednosti.

Singularne vrijednosti σ_i određene su kao kvadratni korijeni svojstvenih vrijednosti matrice A^*A :

$$\sigma_i = \sqrt{\lambda_i},$$

gdje su λ_i svojstvene vrijednosti matrice A^*A .

Lijevi i desni singularni vektori definirani su relacijama:

$$Av_i = \sigma_i u_i,$$

$$A^*u_i = \sigma_i v_i.$$

Ovdje su u_i lijevi singularni vektori, a v_i desni singularni vektori.

Oznake matrica

$$U, \Sigma, V^*$$

ponekad se označavaju i kao

$$U, D, V^*.$$

3 Proposed Watermarking Scheme

Predstaviti ćemo metodu umetanja vodenog žiga u slike baziranu na dekompoziciji slike pomoću SVD-a. U ovoj metodi, tehnike SVD-a koriste se za umetanje vodenog žiga u digitalnu sliku uz očuvanje kvalitete slike i otpornost na različite vrste napada. Posebno se koriste D i U matrice dobivene SVD-om, s primjenom Dither kvantizacije za kontrolu modifikacija.

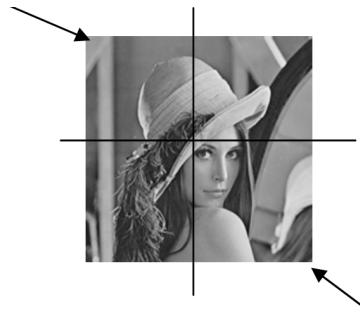
Glavni koraci metode su slijedeći:

3.1 Dekompozicija slike

Originalna slika $f(i, j)$ veličine $N \times N$ dijeli se na četiri pod-slike:

$$\begin{aligned} f_{tl}(p, q) & (\text{gore lijevo}) \\ f_{tr}(p, q) & (\text{gore desno}) \\ f_{bl}(p, q) & (\text{dolje lijevo}) \\ f_{br}(p, q) & (\text{dolje desno}) \end{aligned}$$

Vodeni žig ugraditi ćemo samo u pod-slike $f_{tl}(p, q)$ i $f_{br}(p, q)$ radi smanjenja vizualnih promjena i postizanja boljeg PSNR-a.



Slika 1: Podjela slike na blokove

3.2 Umetanje vodenog žiga u D matricu

Na svaki $M \times M$ blok pod-slike $f_{tl}(p, q)$ (gdje je $M \times M$ veličina vodenog žiga) primjenjuje se SVD, čime dobivamo D matricu.

Iz svakog bloka D -matrice uzimamo najveći koeficijent $D(1, 1)$ i formiramo matricu D_{large} koja odgovara veličini slike vodenog žiga.

Koristeći Dither kvantizaciju, elementi D_{large} mijenjaju se ovisno o vrijednosti bita vodenog žiga (1 ili 0) i kategoriji kvantizacijske tablice:

<i>bin no.</i>	d_{low}	d_{high}
1	$d_{min} - T$	d_{min}
2	d_{min}	$d_{min} + T$
3	$d_{min} + T$	$d_{min} + 2T$
b_{n-1}	$d_{max} - T$	d_{max}
.
.
b_n	d_{max}	$d_{max} + T$

Slika 2: Kvantizacijska tablica

Ako je bit vodenog žiga ‘1’, onda pripada intervalu Range1, gdje je Range1 definiran na slijedeći način:

$$\text{Range1} = d_{\text{low}}(n) \quad \text{do} \quad \frac{d_{\text{low}}(n) + d_{\text{high}}(n)}{2} \quad (1)$$

Tada se D_{large} mijenja u:

$$D_{\text{large}} = \frac{d_{\text{low}}(n) + (d_{\text{low}}(n) + d_{\text{high}}(n)) / 2}{2} \quad (2)$$

Ako je bit vodenog žiga ‘0’, onda pripada intervalu Range2, gdje je Range2 definiran na slijedeći način:

$$\text{Range2} = \frac{d_{\text{low}}(n) + d_{\text{high}}(n)}{2} \quad \text{do} \quad d_{\text{high}}(n) \quad (3)$$

Tada se D_{large} mijenja u:

$$D_{\text{large}} = \frac{d_{\text{high}}(n) + (d_{\text{low}}(n) + d_{\text{high}}(n)) / 2}{2} \quad (4)$$

Nakon modifikacija, inverznom SVD transformacijom dobijemo prvi dio slike s vodenim žigom.

3.3 Umetanje vodenog žiga u U matricu

Na pod-sliku $f_{br}(p, q)$ također primjenjujemo SVD u $M \times M$ blokovima, sada ciljajući U matricu za umetanje vodenog žiga.

Vodni žig $w(i, j)$, $1 \leq i \leq N/2M$ i $1 \leq j \leq N/2M$ umeće se u stupce svakog bloka U matrice. Za svaki $M \times M$ blok U matrice, u_{11} (prvi red, prvi stupac) i u_{21} (drugi red, prvi stupac) modificiraju se na sljedeći način:

$$u_{\text{diff}} = |u_{11}| - |u_{21}| \quad (5)$$

Ako je $w(i, j) = 1$ i $u_{\text{diff}} > \alpha$ ili $w(i, j) = 0$ i $u_{\text{diff}} < \alpha$, tada:

$$u_{21} = - \left| u_{21} - \frac{(\alpha - u_{\text{diff}})}{2} \right| \quad (6)$$

$$u_{11} = - \left| u_{11} + \frac{(\alpha - u_{\text{diff}})}{2} \right| \quad (7)$$

Ako je $w(i, j) = 1$ i $u_{\text{diff}} < \alpha$ ili $w(i, j) = 0$ i $u_{\text{diff}} > \alpha$, tada:

$$u_{21} = - \left| u_{21} - \frac{(\alpha + u_{\text{diff}})}{2} \right| \quad (8)$$

$$u_{11} = - \left| u_{11} + \frac{(\alpha + u_{\text{diff}})}{2} \right| \quad (9)$$

Nakon modifikacije U matrice, inverznim SVD-om na svakom bloku dobijemo drugi dio slike s vodenim žigom.

3.4 Konačna slika s vodenim žigom

Na kraju, sve pod-slike $f_{tlw}(p, q)$, $f_{tr}(p, q)$, $f_{bl}(p, q)$ i $f_{brw}(p, q)$ kombiniraju se u konačnu sliku s vodenim žigom $F(i, j)$.

Predložena metoda pruža visoku razinu otpornosti na uobičajene napade na digitalne slike, kao što su kompresija, rotacija i dodavanje šuma. Korištenjem Dither kvantizacije u procesu umetanja vodenog žiga postiže se precizno prilagođavanje vrijednosti u D matrici, čime se osigurava da vodiči žig ostane skriven, ali istovremeno i otporan na razne modifikacije slike.

Uzimajući u obzir kvalitetu vodenog žiga i neprimjetnost prisutnosti vodenog žiga, metoda nudi učinkovito rješenje za zaštitu autorskih prava i sigurnost digitalnih slika u raznim primjenama.

4 Eksperimentalni rezultati

4.1 Steganografija na slici

Sa sljedećim kodom koristimo steganografiju na primjeru slike mačke i psa gdje sliku psa sakrijemo u sliku mačke. Kod možete vidjeti na našem Github repozitoriju koji se nalazi na [4] pod nazivom "steganografija.m".



Slika 3: Originalne slike

```
1 % Definiramo imena fileova
2 orig_file = 'macka.jpg'; % Originalna slika
3 zig_file = 'pas.jpg'; % Maska slika
4
5 % Ucitamo i procesuiramo prvu sliku
6 coverImage = imread(orig_file);
7 % Pretvorimo ju u crno-bijelu sliku radi izgleda(makar ova
8 vec je
9 coverImage = rgb2gray(coverImage);
10 % Pretvorimo u tip double radi SVDa
11 coverImage = im2double(coverImage);
12 figure(1), imshow(coverImage), title('Originalna slika');
13
14 % Na njoj napravimo SVD
15 [U_cover, S_cover, V_cover] = svd(coverImage);
```

Slika 4: Prvi dio koda

```

1      % Napravimo isto procesiranje za drugu sliku
2      secretImage = imread(zig_file);
3      secretImage = rgb2gray(secretImage);
4      secretImage = im2double(secretImage);
5      [rows, cols] = size(coverImage);
6      % U slucaju da treba masku smanjimo/povecamo na na velicinu
7      % originala
8      secretImage = imresize(secretImage, [rows, cols]);
9      % Napravimo SVD na tajnoj slici
10     [U_secret, S_secret, V_secret] = svd(secretImage);
11     % Ovdje mozemo mijenjati alpha za steganometriju
12     alpha = 0.5;
13     S_stega = S_cover + alpha * S_secret;

14     % Napravimo steganometrijsku sliku
15     stegaImage = U_cover * S_stega * V_cover';
16
17     % Spremanje slike
18     figure(2), imshow(stegaImage), title('Steganografska slika');
19     imwrite(stegaImage, 'stega_output.png');

20     % Ponovno ucitamo stega sliku i napravimo svd na njoj
21     stegaImageReloaded = im2double(imread('stega_output.png'));
22     [U_stega, S_stega_reloaded, V_stega] = svd(
23     stegaImageReloaded);

24     % Pomocu poznate alphe izvucemo tajnu sliku
25     S_extracted_secret = (S_stega_reloaded - S_cover) / alpha;

26     % Rekonstruiramo staru sliku
27     extractedSecretImage = U_secret * S_extracted_secret *
28     V_secret';

29     figure(3), imshow(extractedSecretImage), title('Tajna slika');

30     % Rekonstruiramo originalnu sliku iz stega slike
31     S_reconstructed_original = S_stega_reloaded - alpha *
32     S_secret;
33     reconstructedOriginal = U_stega * S_reconstructed_original *
34     V_stega';

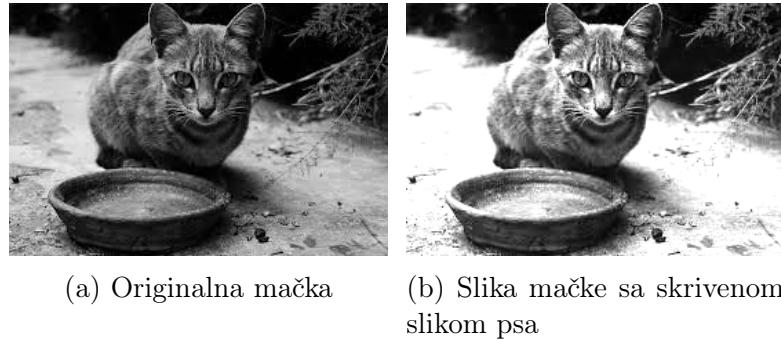
35     figure(4), imshow(reconstructedOriginal), title(
36     'Rekonstruirana originalna slika');

37
38

```

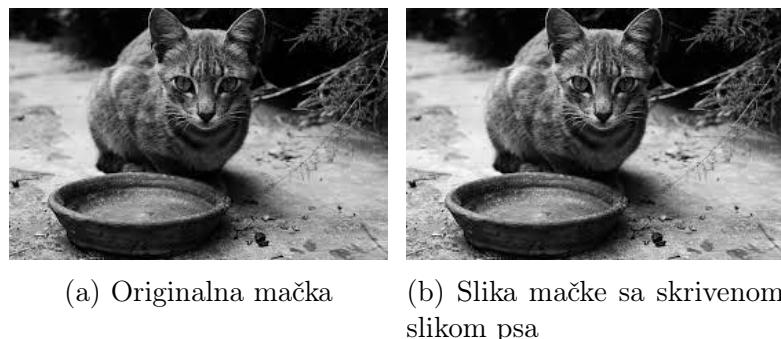
Slika 5: Drugi dio koda

U ovom primjeru smo koristili da nam je $\alpha = 0.5$ te smo dobili sljedeći rezutat (sa strane stavljamo originalnu sliku za usporedbu):



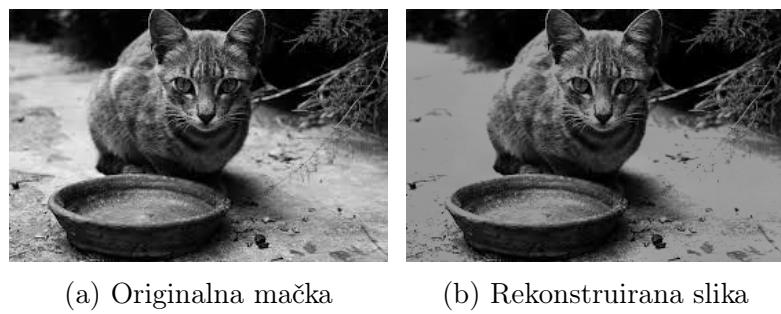
Slika 6: Steganometrija sa $\alpha = 0.5$

U ovom slučaju je α bio relativno velik pa se primjeti da je slika dosta svjetlija zbog toga. Ako uvrstimo $\alpha = 0.05$ razlika u odnosu na originalnu sliku se više baš i ne prepoznaje:



Slika 7: Steganometrija sa $\alpha = 0.05$

S obzirom da ponajemo α možemo i rekonstruirati sliku:



Slika je malo tamnija radi zaokruživanja te ne možemo dobiti potpuno istu sliku.

Ako povećamo α na tipa 0.9, steganografska slika će biti još svjetlijia:



(a) Originalna mačka (b) Rekonstruirana slika

Slika 9: Steganometrija sa $\alpha = 0.9$

Ujedno će i rekonstrukcija biti tamnija:



(a) Originalna mačka

(b) Rekonstruirana slika

To možemo primijetiti i na tajnoj slici koju radi poznavanja α možemo izvući iz steganografske slike:

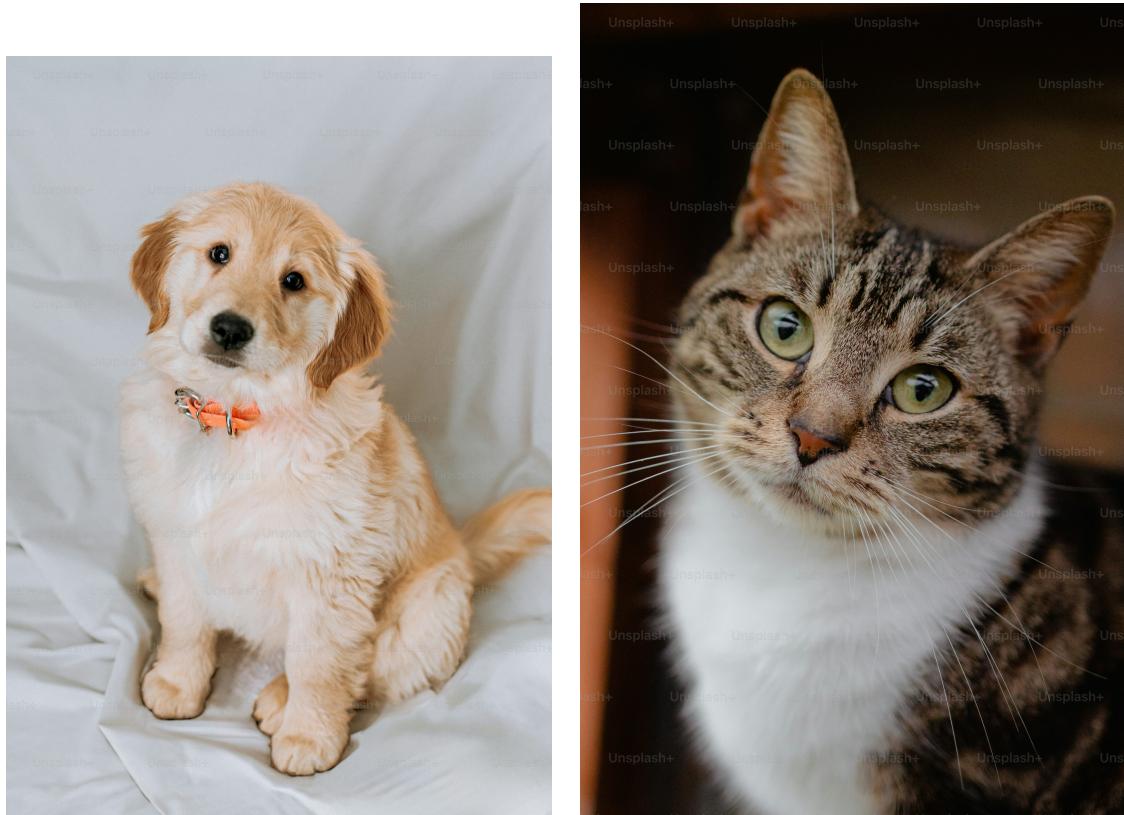


(a) Originalni pas

(b) Rekonstruirana tajna slika
psa

Osim toga, napravili smo i varijantu za slike u boji. Kod za tu varijantu se nalazi

u našem Github repozitoriju[4] pod nazivom "steganografijaColor.png". Ovo su dvije slike koje smo koristili:



Slika 12: originalne slike

U ovom slučaju smo мало obrnuli situaciju te spremili sliku mačke u sliku psa. Pokretanjem koda smo dobili sljedeće:



(a) Originalni pas



(b) Slika pas sa skrivenom sli-
kom mačke

Slika 13: Stegenometrija sa $\alpha = 0.05$

Za situaciju gdje je $\alpha = 0.5$ opet dobijemo više primjetljivu razliku:



(a) Originalni pas



(b) Slika pas sa skrivenom sli-
kom mačke

Slika 14: Stegenometrija sa $\alpha = 0.5$

Također, pošto znamo α možemo izvući tajnu sliku

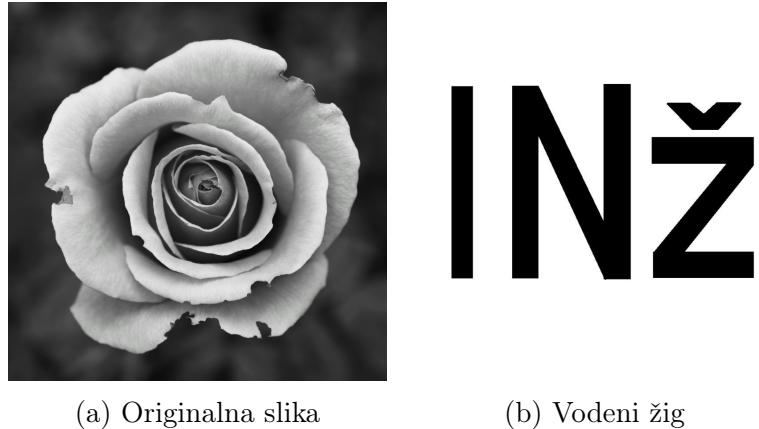


(a) Originalna mačka

(b) Rekonstruirana slika
mačke iz slike psa

4.2 Watermarking scheme

Osim prikazanog korištenja steganografije na slikama, predstaviti ćemo implementiranu metodu iz 3. poglavlja - Proposed Watermarking Scheme. Kod se nalazi na našem Github repozitoriju pod nazivom "watermarking_scheme.m".



(a) Originalna slika

(b) Vodeni žig

Slika 16: Početne slike

```
1 % Ucitavanje slika
2 orig_file = 'cvijet.jpg'; % Originalna slika
3 original = imread(orig_file);
4 original = rgb2gray(original);
5 original = im2double(original);
6 watermark_file = 'inz.png'; % Vodeni žig
7 watermark = imread(watermark_file);
8 watermark = rgb2gray(watermark);
9 watermark = im2double(watermark);

10
11 % Pretvorba dimenzija
12 original = imresize(original, [4000, 4000]);
13 watermark = imresize(watermark, [100, 100]);
14 block_size = 20; % Dimenzija M x M blokova
15 alpha = 0.01;
16
```

Slika 17: Prvi dio koda

```

1 % Osiguravamo da su u vodenom zigu vrijednosti 0 ili 1
2 threshold = 0.5;
3 watermark = watermark > threshold;
4
5 % Podjela originalne slike u 4 bloka
6 [rows, cols] = size(original);
7 tl = original(1:(rows / 2), 1:(cols / 2));
8 tr = original(1:(rows / 2), (cols / 2)+1:end);
9 bl = original((rows / 2)+1:end, 1:(cols / 2));
10 br = original((rows / 2)+1:end, (cols / 2)+1:end);
11
12 % Umetanje vodenog ziga u gornji lijevi blok
13 [tl_rows, tl_cols] = size(tl);
14 [watermark_rows, watermark_cols] = size(watermark);
15 block_size_tl = tl_rows / watermark_rows;
16 tl_w = zeros(tl_rows, tl_cols);
17 modified_d = zeros(watermark_rows, watermark_cols);
18
19 [tl_rows, tl_cols] = size(tl);
20 [watermark_rows, watermark_cols] = size(watermark);
21 block_size_tl = tl_rows / watermark_rows;
22 tl_w = zeros(tl_rows, tl_cols);
23 modified_d = zeros(watermark_rows, watermark_cols);
24 for i = 1:block_size_tl:tl_rows
25     for j = 1:block_size_tl:tl_cols
26         sub_block = tl(i:i+block_size_tl-1, j:j+block_size_tl-1);
27         [U, S, V] = svd(sub_block);
28         d = S(1, 1);
29         row_index = ceil(i / block_size_tl);
30         col_index = ceil(j / block_size_tl);
31         w_bit = watermark(row_index, col_index);
32         % Dither kvantizacija
33         low = floor(d);
34         high = ceil(d);
35         mid = (low + high) / 2;
36         if w_bit == 1
37             d = (low + mid) / 2;
38         else
39             d = (mid + high) / 2;
40         end
41         S(1, 1) = d;
42         modified_d(row_index, col_index) = d;
43         tl_w(i:i+block_size_tl-1, j:j+block_size_tl-1) = U * S * V';
44     end
45 end
46
47 % Umetanje vodenog ziga u donji desni blok
48 [br_rows, br_cols] = size(br);
49 block_size_br = br_rows / watermark_rows;
50 br_w = zeros(br_rows, br_cols);    16
51
52

```

Slika 18: Drugi dio koda

```

1
2 for i = 1:block_size_br:br_rows
3     for j = 1:block_size_br:br_cols
4         sub_block = br(i:i+block_size_br-1, j:j+block_size_br-1);
5         [U, S, V] = svd(sub_block);
6         u11 = U(1, 1);
7         u21 = U(2, 1);
8         row_index = ceil(i / block_size_br);
9         col_index = ceil(j / block_size_br);
10        w_bit = watermark(row_index, col_index);
11        % Modifikacija vrijednosti u11 i u21
12        diff = abs(u11) - abs(u21);
13        if (w_bit == 1 && diff > alpha) || (w_bit == 0 && diff <
alpha)
14            u21 = u21 - (alpha - diff) / 2;
15            u11 = u11 + (alpha - diff) / 2;
16        else
17            u21 = u21 - (alpha + diff) / 2;
18            u11 = u11 + (alpha + diff) / 2;
19        end
20        U(1, 1) = u11;
21        U(2, 1) = u21;
22        br_w(i:i+block_size_br-1, j:j+block_size_br-1) = U * S * V';
23    end
24 end
25
26 % Rekonstrukcija slike s vodenim zigom
27 watermarked_image = [tl_w, tr; bl, br_w];
28 imshow(watermarked_image);
29 title(['Slika s umetnutim vodenim zigom (alpha = ', num2str(alpha),
')']);
30 imwrite(watermarked_image, ['watermarked_image_alpha_', num2str(
alpha), '.jpg']);
31

```

Slika 19: Treći dio koda

Slijedno, prikazujemo kod za ekstrakciju vodenog žiga. Može se pronaći na našem repozitoriju pod nazivom "watermark_extract.m".

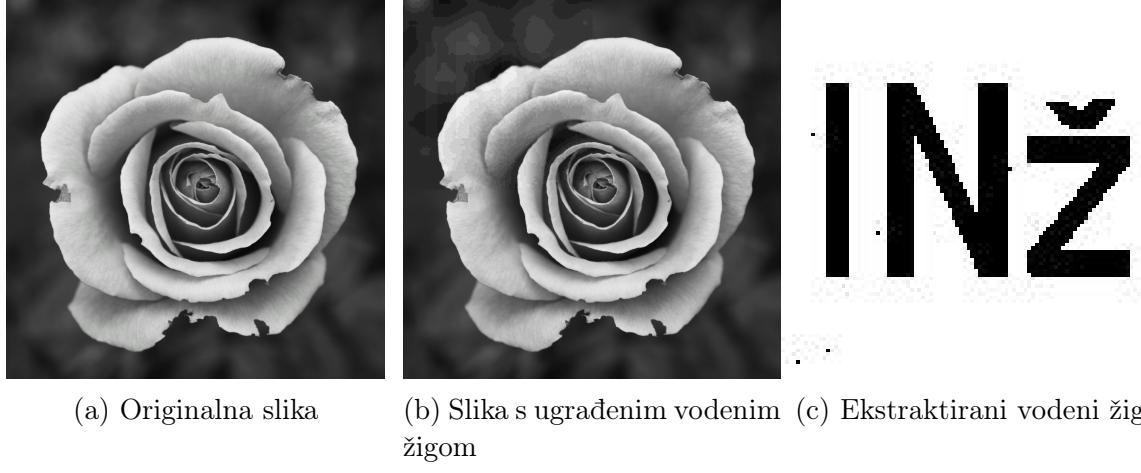
```

1 image = imread('zgrade.jpg');
2 image = im2double(image);
3
4 % Pretvorba dimenzija
5 image = imresize(image, [4000, 4000]);
6
7 block_size = 20; % Dimenzija M x M blokova
8
9 % Podjela slike u 4 bloka
10 [rows, cols] = size(image);
11 tl = image(1:(rows / 2), 1:(cols / 2));
12 tr = image(1:(rows / 2), (cols / 2)+1:end);
13 bl = image((rows / 2)+1:end, 1:(cols / 2));
14 br = image((rows / 2)+1:end, (cols / 2)+1:end);
15
16 % Ekstrakcija vodenog ziga
17 extracted_watermark_d2 = zeros([100, 100]);
18 [tl_rows, tl_cols] = size(tl);
19 watermark_rows = 100;
20 watermark_cols = 100;
21 block_size_tl = tl_rows / watermark_rows;
22
23 for i = 1:block_size_tl:tl_rows
24     for j = 1:block_size_tl:tl_cols
25         sub_block = tl(i:i+block_size_tl-1, j:j+block_size_tl-1);
26         [U, S, V] = svd(sub_block);
27         d = abs(S(1, 1));
28         row_index = ceil(i / block_size_tl);
29         col_index = ceil(j / block_size_tl);
30
31         low = floor(d);
32         high = ceil(d);
33         mid = (low + high) / 2;
34
35         if abs(d - mid) < abs(d - low)
36             extracted_watermark_d2(row_index, col_index) = 0;
37         else
38             extracted_watermark_d2(row_index, col_index) = 1;
39         end
40     end
41 end
42
43 figure;
44 imshow(extracted_watermark_d2, []);
45 title(['Rekonstruirani vodeni zig ']);
46
47

```

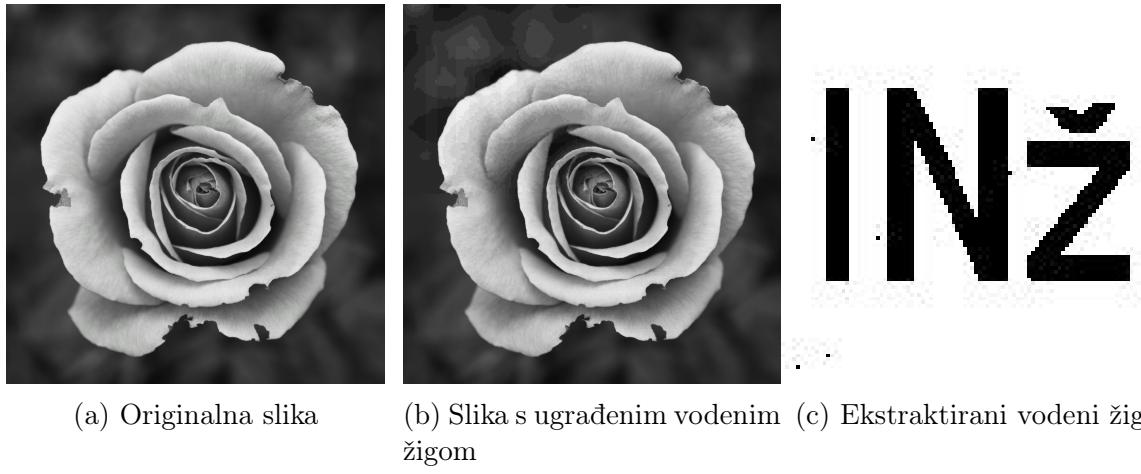
Slika 20: Ekstrakcija vodenog žiga

U ovom primjeru smo koristili $\alpha = 0.01$ te smo dobili slijedeći rezutat):



Slika 21: $\alpha = 0.01$

Vidimo da je na sliku poprilično neprimjetno ugrađen vodeni žig. Ako povećamo α na 0.05 dobivamo slijedeće rezultate:



Slika 22: $\alpha = 0.05$

Vidimo da je vodeni žig još uvijek neprimjetan. Ako drastičnije povećamo α , npr. na 0.9 dobivamo slijedeće rezultate:



(a) Originalna slika



(b) Slika s ugrađenim vodenim žigom



(c) Ekstraktirani vodeni žig

Slika 23: $\alpha = 0.9$

Sada možemo primijetiti promjenu na slici. Pošto se α koristi samo kod umetanja žiga u donji desni dio slike, promjena je vidljiva samo na tom bloku slike. Također, pošto smo za ekstrakciju vodenog žiga koristili gornji lijevi blok slike, ekstraktirani vodeni žig je isti za sve vrijednosti α .

4.3 Testiranje robustnosti

4.4 Photo collage experiment

Metoda kolaža u obradi slika podrazumijeva sastavljanje nove slike kombiniranjem više pojedinačnih slika ili dijelova slika. Ova tehnika često se koristi u umjetničkim projektima, reklamama, te u digitalnom dizajnu kako bi se kreirali jedinstveni vizualni prikazi.



Slika 24: Primjer fotografskog kolaža

Na slici 24 prikazan je fotografski kolaž koji kombinira različite elemente, uključujući dječji portret, pejzaže i arhitektonске motive poput kuća. Ovakve slike često služe kao ilustracija kreativnog spajanja više izvora, ali istovremeno predstavljaju izazov za metode digitalnog vodenog žiga, kao što je SVD.

Cilj ovog eksperimenta je istražiti otpornost metode singularne dekompozicije vrijednosti (SVD) za umetanje i ekstrakciju digitalnog vodenog žiga u kontekstu manipulacije slikama tehnikom kolaža. Konkretno, fokusirat ćemo se na umetanje vodenog žiga u fotografiju cvijeta, a zatim izmijeniti tu sliku dodavanjem vizualnih elemenata u stilu kolaža.

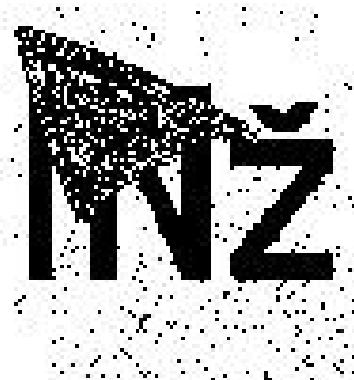
Nakon što je na izvornu sliku cvijeta umetnut vodeni žig pomoću metode SVD, slika će biti dodatno manipulirana kako bismo simulirali tehniku kolaža. Specifično, na sliku cvijeta dodane su četiri slike zgrada koje simbolično "izlaze" iz cvijeta.



(a) Fotografija s watermarkom
(double)



(b) Manipulirana fotografija



(c) Ekstraktirani vodeni žig

Analiza rezultata pokazala da je žig oštećen, iako manipulacija nije bila opsežna. Dodavanje vizualnih elemenata u stilu kolaža izmijenilo glavne strukture slike.



(a) Fotografija s watermarkom
(double)



(b) Manipulirana fotografija



(c) Ekstraktirani vodeni žig

U provedenom eksperimentu primijećeno je da metoda ne funkcioni optimalno u ovom specifičnom slučaju zbog ograničenog broja blokova korištenih za umetanje i ekstrakciju žiga. Naime, slika je podijeljena u samo četiri bloka, što značajno smanjuje kapacitet za umetanje vodenog žiga. Ograničeni broj blokova otežava precizno kodiranje informacija vodenog žiga, što rezultira njegovim djelomičnim gubitkom ili teškoćom u ekstrakciji nakon manipulacija slikom, poput dodavanja elemenata kolaža. Iako manipulacija slikom u ovom slučaju nije bila opsežna, nedovoljan broj blokova uzrokuje visoku osjetljivost na čak i manje promjene.

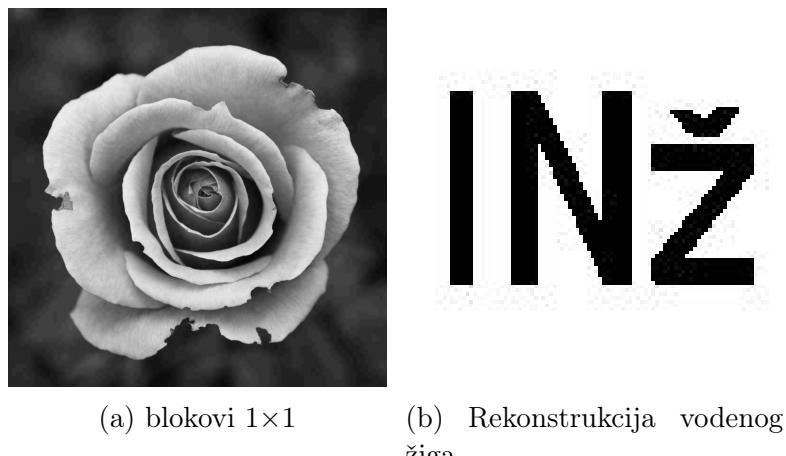
```

1 image = imread('zgrade.jpg');
2 image = im2double(image);
3
4 % Pretvorba dimenzija
5 image = imresize(image, [2000, 2000]);
6
7 block_size = 10; % Dimenzija M x M blokova
8
9 % Podjela slike u 4 bloka
10 [rows, cols] = size(image);
11 tl = image(1:(rows / 2), 1:(cols / 2));
12 tr = image(1:(rows / 2), (cols / 2)+1:end);
13 bl = image((rows / 2)+1:end, 1:(cols / 2));
14 br = image((rows / 2)+1:end, (cols / 2)+1:end);
15
16 % Ekstrakcija vodenog ziga v2
17 extracted_watermark_d2 = zeros([100, 100]);
18 [tl_rows, tl_cols] = size(tl);
19 watermark_rows = 100;
20 watermark_cols = 100;
21 block_size_tl = tl_rows / watermark_rows;
22
23 for i = 1:block_size_tl:tl_rows
24     for j = 1:block_size_tl:tl_cols
25         sub_block = tl(i:i+block_size_tl-1, j:j+block_size_tl-1);
26         [U, S, V] = svd(sub_block);
27         d = abs(S(1, 1));
28         row_index = ceil(i / block_size_tl);
29         col_index = ceil(j / block_size_tl);
30
31         low = floor(d);
32         high = ceil(d);
33         mid = (low + high) / 2;
34
35         if abs(d - mid) < abs(d - low)
36             extracted_watermark_d2(row_index, col_index) = 0;
37         else
38             extracted_watermark_d2(row_index, col_index) = 1;
39         end
40     end
41 end
42
43 figure;
44 imshow(extracted_watermark_d2, []);
45 title(['Rekonstruirani vodič zig v2 (alpha = ', num2str(alpha), ')']);
46 imwrite(extracted_watermark_d2, ['reconstructed_watermark_v2_alpha_',
47 , num2str(alpha), '.jpg']);

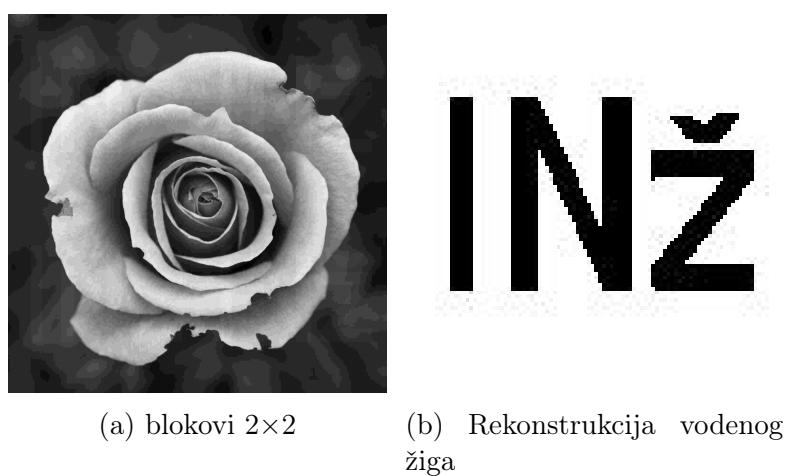
```

4.4.1 Umetanje žiga u blokove

Za umetanja vodenog žiga, koristimo mrežu blokova kako bismo povećali robusnost žiga. Razbijanje slike na blokove omogućava ravnomernu distribuciju žiga kroz cijelu sliku, čime se smanjuje utjecaj pojedinačnih promjena u većim dijelovima slike i čini ga otpornijim na različite vrste manipulacija, poput kompresije ili promjena u kontrastu. U sljedećim primjerima prikazat ćemo kako različite veličine mreže (grid) utječu na izgled slike i jasnoću vodenog žiga, prikazujući rezultate za nekoliko veličina mreže, poput 4×4 , 5×5 i 6×6 .



Slika 28: Početne slike



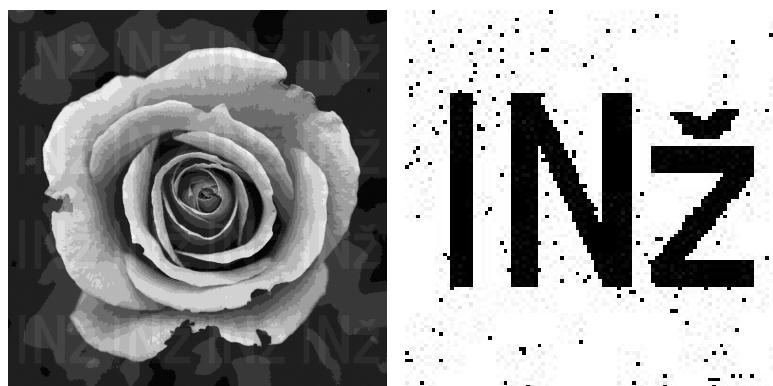
Slika 29: Početne slike



(a) blokovi 3×3

(b) Rekonstrukcija vodenog ūiga

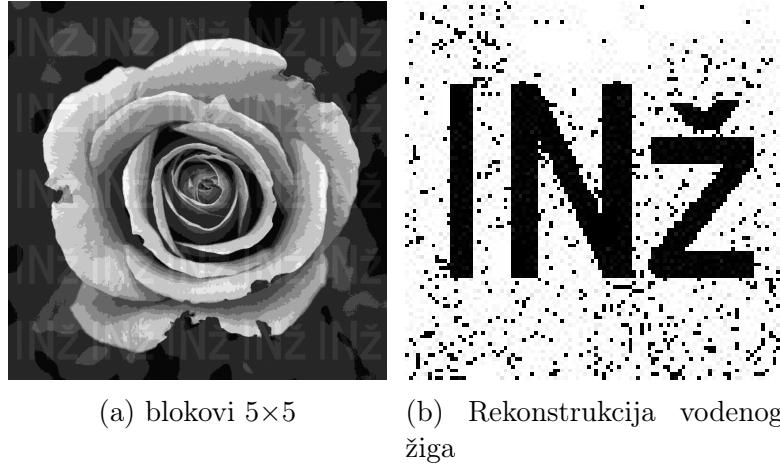
Slika 30: Početne slike



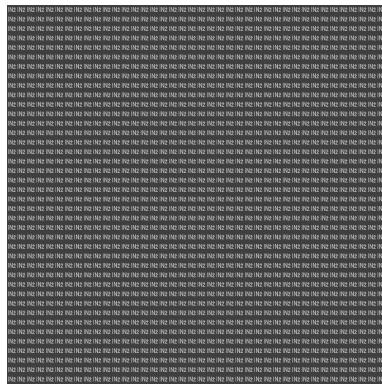
(a) blokovi 4×4

(b) Rekonstrukcija vodenog ūiga

Slika 31: Početne slike



Slika 32: Početne slike



(a) blokovi 40×40

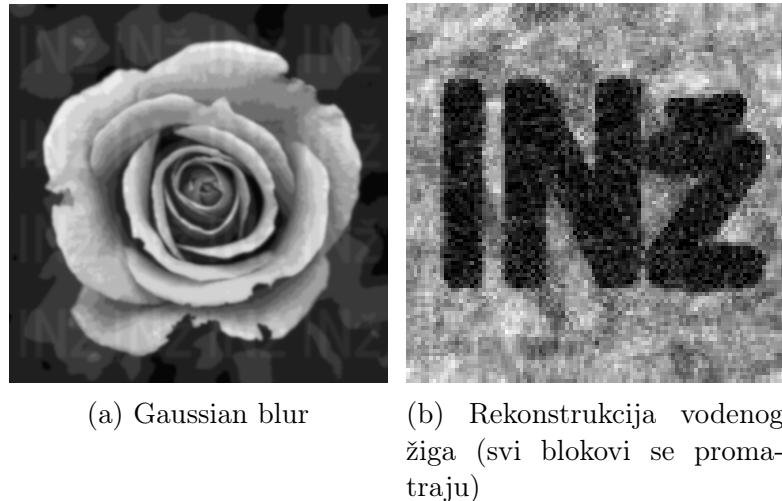
Slika 33: Početne slike

Korištenje veće mreže blokova može povećati robusnost vodenog žiga, ali istovremeno može otežati njegovu ekstrakciju, jer se informacije o žigu raspodjeljuju kroz manji broj piksela unutar svakog bloka. Previše blokova može dovesti do toga da žig postane previše "raštrkan" i teško prepoznatljiv tijekom procesa ekstrakcije, čime se smanjuje njegov kvalitet i jasnoća. Stoga, važno je pronaći ravnotežu u veličini mreže, kako bi žig bio dovoljno otporan na manipulacije, a istovremeno očuvala njegova prepozнатljivost prilikom kasnije ekstrakcije.

Daljnja testiranja ćemo provoditi s mrežom 4×4 blokova, jer razlika u ekstrakciji vodenog žiga između 4×4 i 5×5 blokova pokazuje značajnu degradaciju kvalitete žiga. S druge strane, razlika između 3×3 i 4×4 blokova nije toliko izražena, a korištenjem

mreže 4×4 želimo postići bolju pokrivenost prostora i ravnotežu između robusnosti i očuvanja vidljivosti žiga.

4.4.2 Zamućivanje



(a) Gaussian blur

(b) Rekonstrukcija vodenog žiga (svi blokovi se promatraju)

Slika 34: Gaussian blur (strong)



(a) Gaussian blur

(b) Rekonstrukcija vodenog žiga (svi blokovi se promatraju)

(c) Rekonstrukcija vodenog žiga (samo zadnji blok se promatra)

Slika 35: Gaussian blur (weak)



(a) Focus blur



(b) Rekonstrukcija vodenog
žiga (samo zadnji blok se pro-
matra)



(c) Rekonstrukcija vodenog
žiga (svi blokovi se proma-
traju)



(a) Lens blur



(b) Rekonstrukcija vodenog
žiga (samo zadnji blok se pro-
matra)



(c) Rekonstrukcija vodenog
žiga (svi blokovi se proma-
traju)

4.4.3 Rotacija



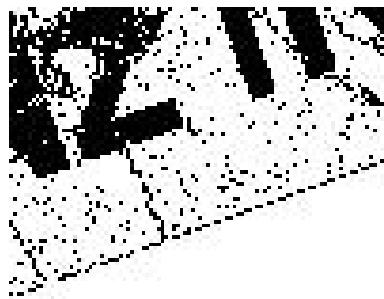
(a) Small rotation

(b) Rekonstrukcija vodenog žiga (samo zadnji blok se promatra)

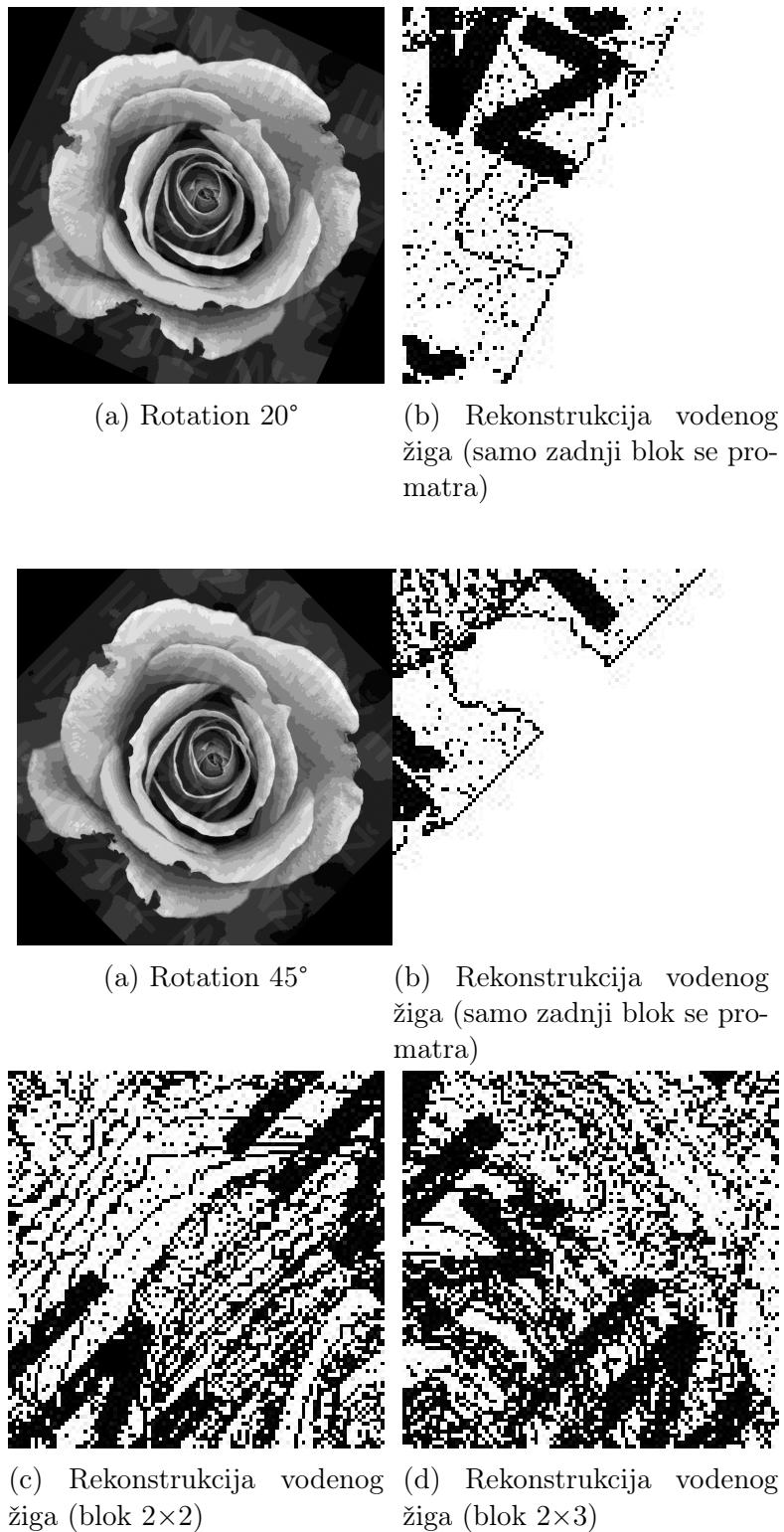
(c) Rekonstrukcija vodenog žiga (svi blokovi se promatraju)



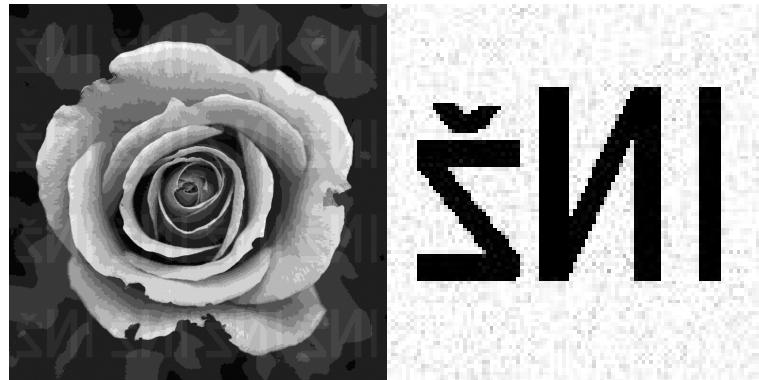
(a) Rotation -20°



(b) Rekonstrukcija vodenog žiga (samo zadnji blok se promatra)

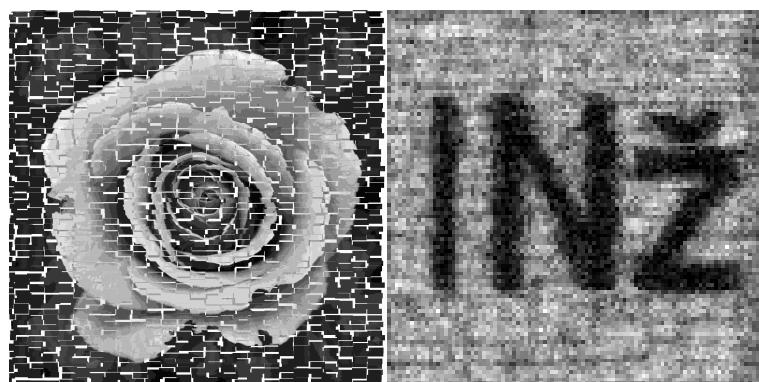


4.4.4 Ostalo



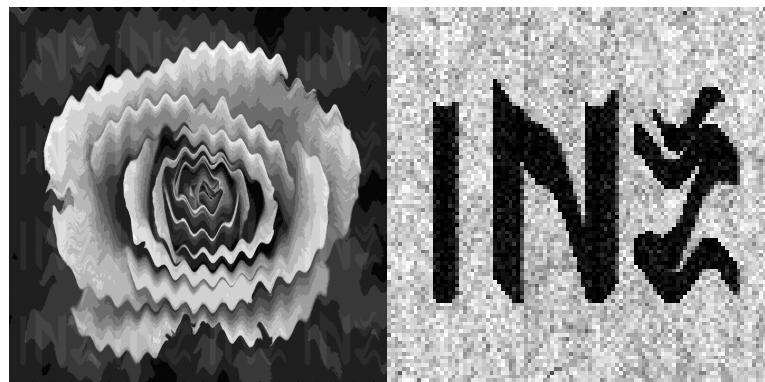
(a) Horizontal flip

(b) Rekonstrukcija vodenog
žiga (samo zadnji blok se pro-
matra)



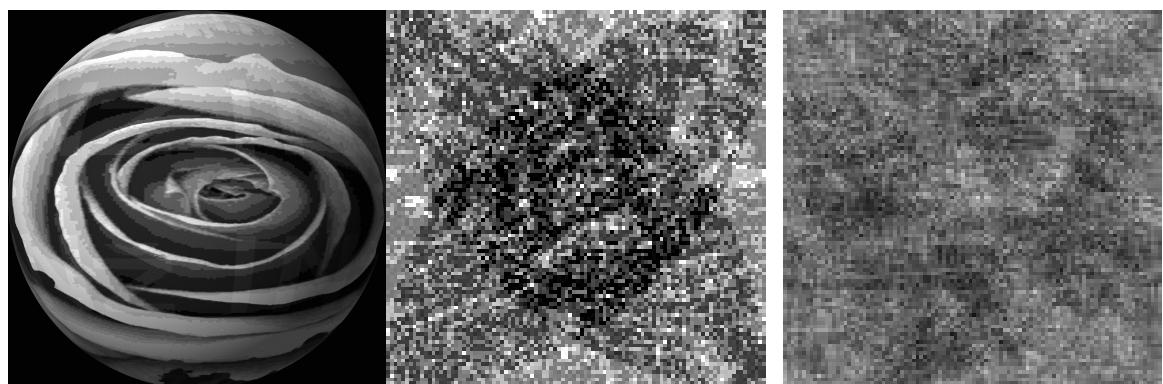
(a) Paper tale

(b) Rekonstrukcija vodenog
žiga (svi blokovi se proma-
traju)



(a) Distort ripple

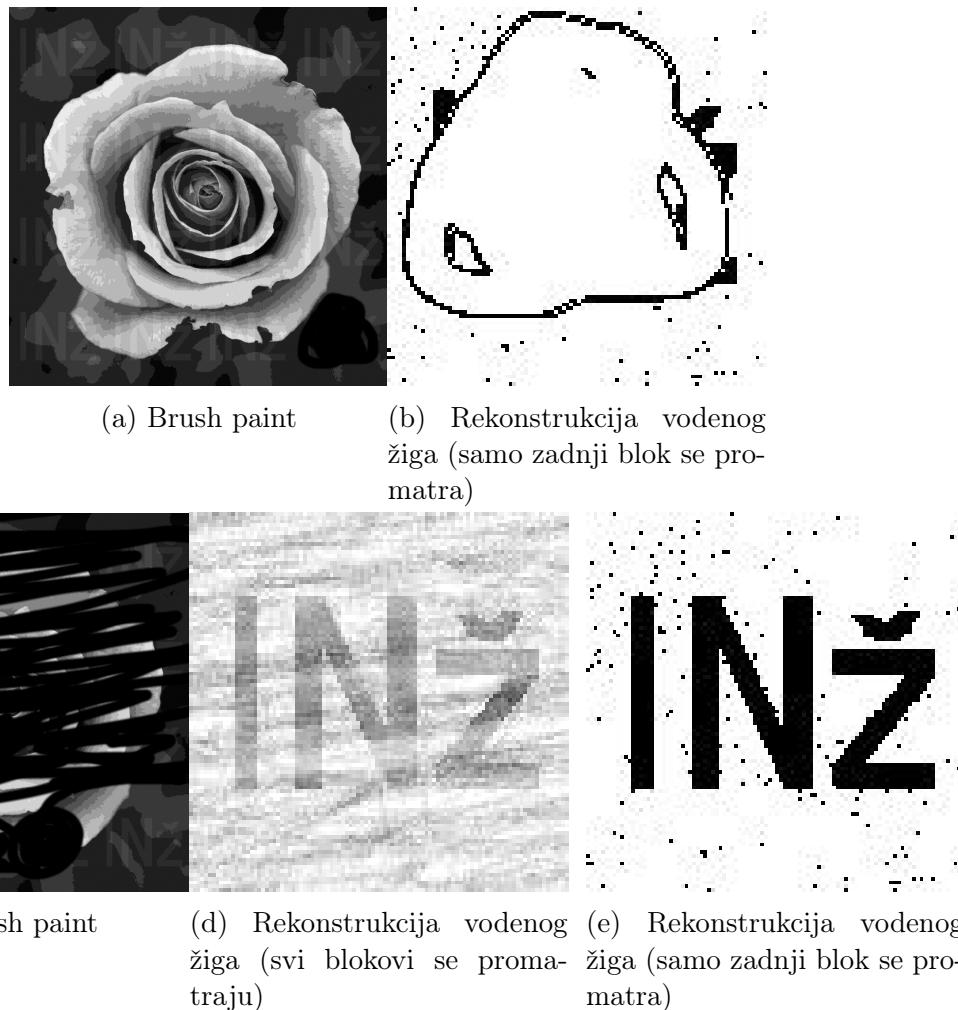
(b) Rekonstrukcija vodenog
žiga (svi blokovi se proma-
traju)



(a) Spinning globe

(b) Rekonstrukcija vodenog
žiga (svi blokovi se proma-
traju)

(c) Rekonstrukcija vodenog
žiga (zadnji blok se promatra)



4.4.5 JPG format

Također smo pokušali ubaciti žig u sliku, spremiti sliku u JPG format te zatim pokušati izvaditi žig iz nje te vidjeti je li žig ostao sačuvan. U našem slučaju je žig i dalje ostao prepoznatljiv:

INŽ

Slika 47: Žig izvađen iz slike u JPG formatu

```

1 % Ucitavanje slike
2 orig_file = '../Slike/cvijet.jpg'; % Originalna slika
3 original = imread(orig_file);
4 original = rgb2gray(original);
5 original = im2double(original);
6 watermark_file = '../Slike/inz.png'; % Vodenica
7 watermark = imread(watermark_file);
8 watermark = rgb2gray(watermark);
9 watermark = im2double(watermark);
10
11 % Pretvorba dimenzija
12 original = imresize(original, [4000, 4000]);
13 watermark = imresize(watermark, [100, 100]);
14 alpha = 0.01;
15
16 % Osiguravamo da su u vodenom igu vrijednosti 0 ili 1
17 threshold = 0.5;
18 watermark = watermark > threshold;
19
20 % Parametri za blokove
21 [rows, cols] = size(original);
22 num_blocks = 4; % Mreza 4x4 blokova
23 block_rows = rows / num_blocks;
24 block_cols = cols / num_blocks;
25
26 % Dimenzije vodenog iga u skladu s blokovima
27 [watermark_rows, watermark_cols] = size(watermark);
28 block_size = block_rows / watermark_rows;
29
30 % Umetanje vodenog iga u svaki blok
31 watermarked_image = zeros(rows, cols); % Slika s vodenim igom
32 for row_block = 1:num_blocks
33     for col_block = 1:num_blocks
34         % Izolacija trenutnog bloka
35         r_start = (row_block - 1) * block_rows + 1;
36         r_end = row_block * block_rows;
37         c_start = (col_block - 1) * block_cols + 1;
38         c_end = col_block * block_cols;
39         block = original(r_start:r_end, c_start:c_end);
40
41         % Umetanje vodenog iga u trenutni blok
42         block_w = zeros(size(block));
43         for i = 1:block_size:block_rows
44             for j = 1:block_size:block_cols
45                 sub_block = block(i:i+block_size-1, j:j+block_size
46 -1);
46                 [U, S, V] = svd(sub_block);
47                 d = S(1, 1);
48                 row_index = ceil(i / block_size);
49                 col_index = ceil(j/ block_size);
50                 w_bit = watermark(row_index, col_index);
51                 % Dither kvantizacija
52                 low = floor(d);
53                 high = ceil(d);
54                 mid = (low + high) / 2;
55                 if w_bit == 1
56                     d = (low + mid) / 2;
57                 else
58                     d = (mid + high) / 2;
59                 end
60                 S(1, 1) = d;

```

```

1 % Ucitavanje slike s vodenim igaom
2 image = imread('watermarked_image_grid_4x4_alpha_0.01_spinning_globe
   .jpg');
3 image = im2double(image);
4
5 % Pretvorba dimenzija
6 image = imresize(image, [4000, 4000]);
7
8 % Parametri za blokove
9 [rows, cols] = size(image);
10 num_blocks = 4; % Mre a 4x4 blokova
11 block_rows = rows / num_blocks;
12 block_cols = cols / num_blocks;
13
14 % Dimenzije vodenog iga u skladu s blokovima
15 watermark_rows = 100;
16 watermark_cols = 100;
17 block_size = block_rows / watermark_rows;
18
19 % Ekstrakcija vodenog iga iz svakog bloka
20 extracted_watermark = zeros(watermark_rows, watermark_cols);
21
22 % Prolaz kroz sve blokove
23 for row_block = 1:num_blocks
24     for col_block = 1:num_blocks
25         % Izolacija trenutnog bloka
26         r_start = (row_block - 1) * block_rows + 1;
27         r_end = row_block * block_rows;
28         c_start = (col_block - 1) * block_cols + 1;
29         c_end = col_block * block_cols;
30         block = image(r_start:r_end, c_start:c_end);
31
32         % Ekstrakcija vodenog iga iz trenutnog bloka
33         for i = 1:block_size:block_rows
34             for j = 1:block_size:block_cols
35                 sub_block = block(i:i+block_size-1, j:j+block_size
-1);
36                 [U, S, V] = svd(sub_block);
37                 d = abs(S(1, 1)); % Ekstrakcija najve e singularne
vrijednosti
38
39         % Izra un pozicije u vodenom igu
40         row_index = ceil(i / block_size);
41         col_index = ceil(j / block_size);
42
43         % Kvantizacija za rekonstrukciju
44         low = floor(d);
45         high = ceil(d);
46         mid = (low + high) / 2;
47
48         if abs(d - mid) < 36
49             extracted_watermark(row_index, col_index) =
50             extracted_watermark(row_index, col_index) + 0; % Nema iga
51         else
52             extracted_watermark(row_index, col_index) =
53             extracted_watermark(row_index, col_index) + 1; % ig prisutan
54         end
55     end
56 end

```

5 Zaključak

SVD (Singular Value Decomposition) metoda za umetanje i ekstrakciju vodenih žigova pokazala se kao vrlo moćna i robustna. Međutim, ključni faktor u njezinoj učinkovitosti je prilagodba parametara, kao što su veličina blokova i intenzitet žiga, specifičnostima same slike. Ukoliko slika pretrpi ozbiljna oštećenja, poput efekta "spinning globe", gdje se slika deformira ili rotira, metoda postaje manje robustna i može doći do gubitka informacija o žigu. Ipak, u slučajevima gdje slika ostaje dovoljno netaknuta, SVD metoda pruža visoku sigurnost i efikasnost, omogućujući očuvanje integriteta vodenog žiga uz minimalan utjecaj na vizualni izgled slike.

6 Literatura

Literatura

- [1] B. Chandra Mohan, S. Srinivas Kumar, *A Robust Image Watermarking Scheme using Singular Value Decomposition*, JNTU College of Engineering, ECE Department, Kakinada, India
- [2] Ruizhen Liu, Tieniu Tan, *A SVD-Based Watermarking Scheme for Protecting Rightful Ownership*, National Lab of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, P. O. Box 2728, Beijing, 100080, P. R. China
- [3] Zecheng Kuang, *Singular-Value Decomposition and its Applications*, Department of Mathematics, University of California San Diego, La Jolla, CA.
- [4] Naš Github repozitorij sa kodom: ([https://github.com/NikolaKasnar/
Image-watermarking-using-SVD](https://github.com/NikolaKasnar/Image-watermarking-using-SVD))