Detekcija ključnih točk

David Rubin (david.rubin@student.um.si)

22. avgust 2021

1 Zahteve naloge

Za poročilo/zagovor se pričakujejo naslednje demonstracije:

- **Fast detektor**
  - hiter test
    Na sliki prikažite točke, ki jih potrdite s hitrim testom in točke, ki jih potrdite s celotnim testom. Za izbrano testno sliko preštejte in primerjajte število točk najdenih s hitrim testom z številom končnih točk.
  - odstranjevanje nemaksimalnih odzivov
    Prikažite točke pred in po odstranjevanju nemaksimalnih odzivov.
  - iskanje točk različnih velikosti
    Točke različnih velikosti ločeno prikažite na izvorni sliki. Prikažite rezulatate za vsaj tri velikosti.
- **BRIEF deskriptor**
  - izračun orientacije točke
    Za vsako FAST točko v sliki s črto prikažite orientacijo točke (črta je vektor smeri). Prikažite tudi rezultate za isto sliko, rotirano za 45 stopinj na kateri bodo vidne iste točke z ustrezno prilagojenimi orientacijami.
  - izračun orientacije opisa
    Za FAST točko v sliki izrišite izbrane BRIEF točke v njeni okolici (nekje 10 BRIEF točk). Prikažite iste točke poravnane z orientacijo točke v rahlo rotirani sliki (pozicije BRIEF točke bi se morale ohraniti relativno na orientacijo FAST točke).
  - primerjava točk
    Pokažite ujemajoče točke med sliko in njeno rahlo modificirano (translirano, rotirano, skalirano) različico. Večina točk se mora ustrezno ujemati

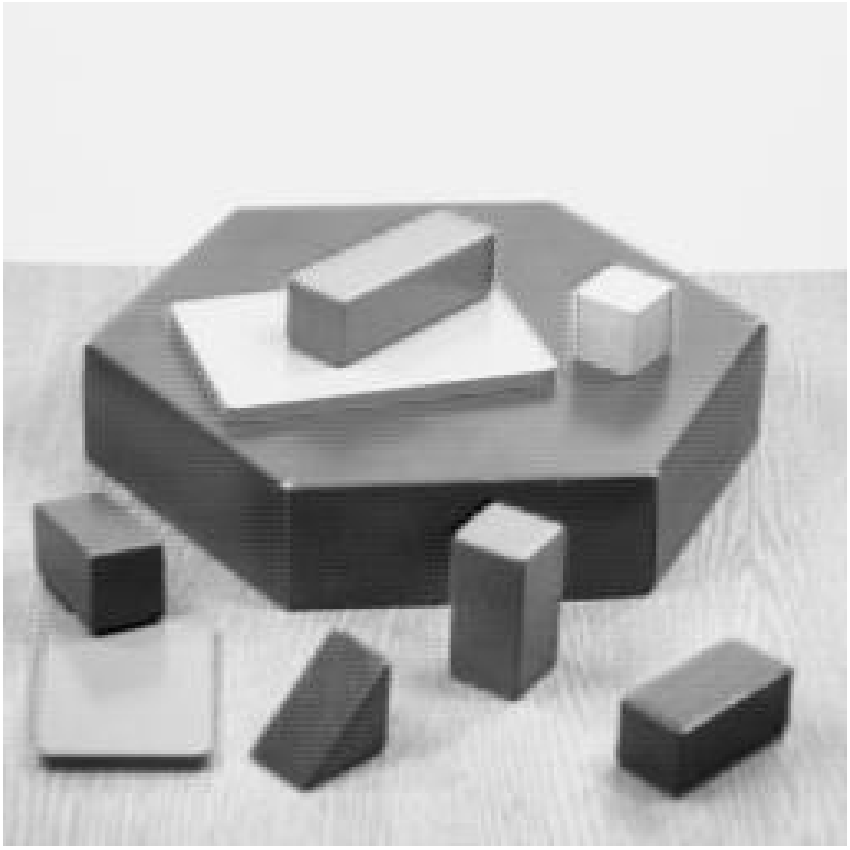
2 Implementacija

Nalogo sem implementiral v programsken jeziku Python, za delovanje so potrebne knjižnice *numpy* (za pospešene operacije nad seznami), *matplotlib* (za pomoč pri risanju grafov in slik),

in *open-cv* (za branje in pisanje slik). Pri risanju ujemajočih točk med dvema slikama sem še uporabil funkcijo *plot_matches* iz knjižnice *scikit-image*. Projekt je sestavljen iz 4 datotek:

- `fast.py` vsebuje implementacijo detektorja ključnih točk FAST,
- `brief.py` vsebuje implemetacijo deskriptorja BRIEF,
- `orb.py` povezuje prej omenjeni datoteki in nudi metodo, ki skuša povezati ključne točke na 2 *podobnih* slikah,
- `utils.py` pa vsebuje pomožne metode, kot so recimo izračun Hammingove razdalje ali pa branje slik iz datotek.

Za demonstracijo delovanja algoritma FAST smo uporabili sliko 1.



Slika 1: Slika blox. Vir: http://52.55.172.202/testes/opencv-3.2.0/samples/data/

2.1 Hitri test vs. poln

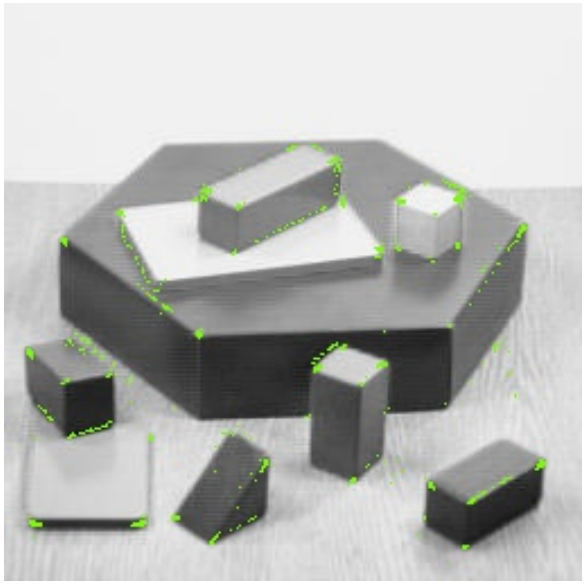
Pri hitrem testu preverjam le 4 skrajne točke v krogu okoli opazovanega oglišča (skrajno severno, južno, vzodno in zahodno). Kadar so vrednosti v vsaj 3 izmed teh ogliščih višje oziroma nižje

od praga. Torej veljati mora

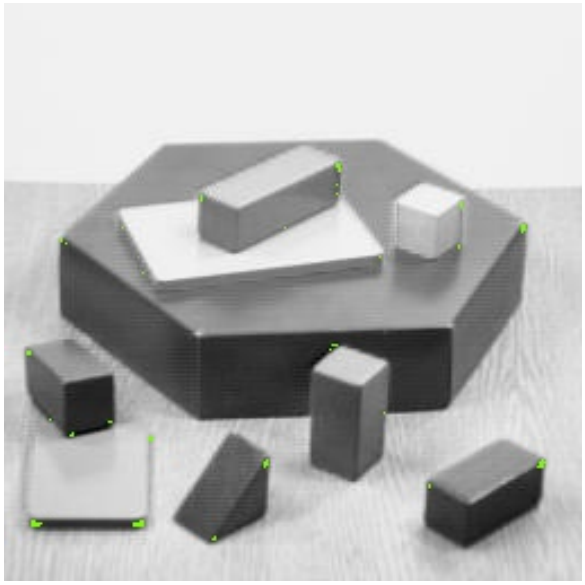
$$p_x > p_{center} + t$$
$$\text{ali} \tag{1}$$
$$p_x < p_{center} - t,$$

pri čemer p_x predstavlja vrednost piksla v enem izmed 4 pozicij, p_{center} predstavlja opazovan piksel (kandidat za ključno točko), t pa je uporabniško določena pragovna vrednost. V kolikor je hitri test pokazal morebitno ogljišče nad to lokacijo preverim še vseh 16 pikslov, ki jo obkrožajo, pri temu pa mora veljati, da je vsaj 12 zaporednih, ki ustrezajo (izključno) enemu izmed pogojev 1.

Nad testno sliko 1 je bila pognana priložena implementacija FAST algoritma s pragovno vrednostjo 17 in brez odstranjevanja kandidatov s pomočjo nemaksimalnih odzivov (angl. *nonmax supression*). Rezultati so vidni na sliki 2.



(a) Kandidati za ključne točke (oglišča) pridobljeni s pomočjo hitrega testa.



(b) Ključne točke, ki smo jih potrdili kot oglišča.

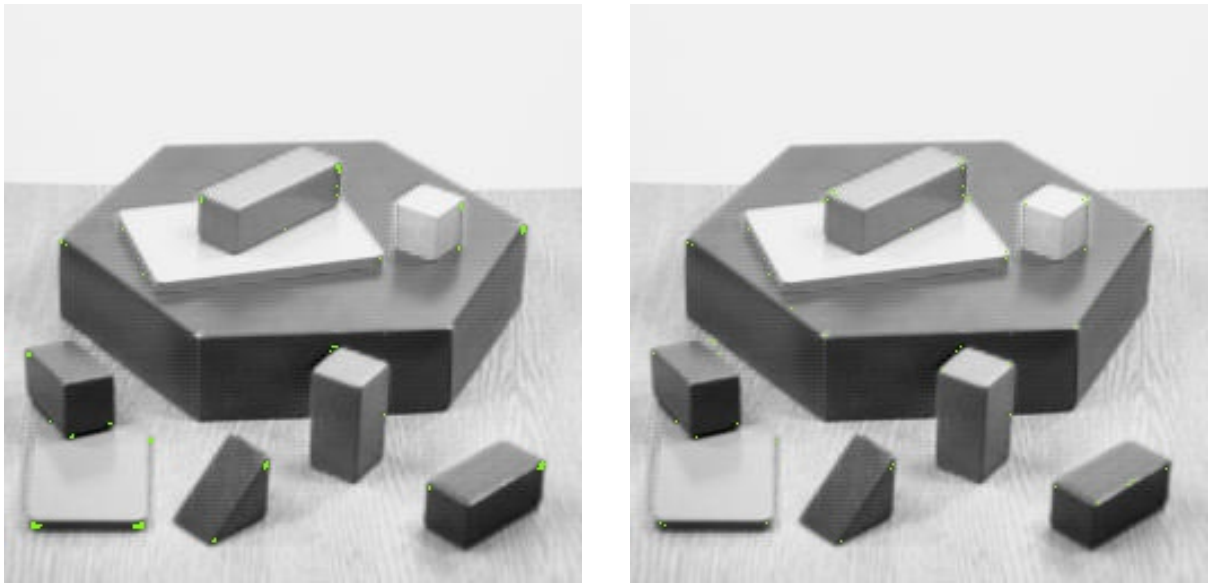
Slika 2: Ključne točke so označene z zeleno barvo. Pragovna vrednost je znašala 17 in opcija nonmax je bila izključena (`False`).

2.2 Odstranjevanje nemaksimalnih odzivov

Pri odstranjevanju nemaksimalnih odzivov skušamo omejiti vpliv pikslov, ki so blizu skupaj in sklepamo, da opisujejo isto ogljišče. Za vsako ključno točko poračunamo vrednost:

$$s = \sum_{n=1}^{16} p_n - p_{center} - t$$
$$\text{oziroma} \tag{2}$$
$$s = \sum_{n=1}^{16} p_{center} - t - p_n$$

Pri tem predstavlja p_{center} pikselno vrednost v središču, t je pragovna vrednost, p_n pa vrednost na krožnici okoli centra. Prva enačba velja, kadar smatramo ključtno točko svetlejšo od pragovne vrednosti, druga pa kadar je temnejša. V kolikor imamo takšno vrednost, lahko za vsako ključno točko preverimo sosednje piksle in odstranimo tiste, ki imajo nižjo vrednost. Rezultat primerjave detekcije točk z uporabe te tehnike in brez je predstavljen na sliki 3.

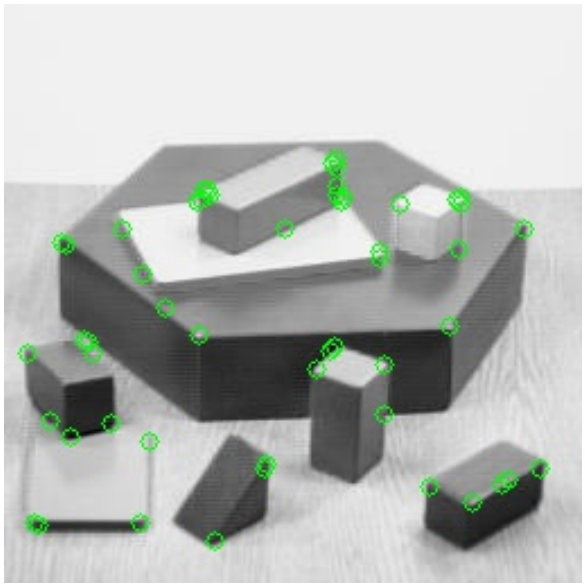


(a) Ogljišča brez uporabe nonmax. (b) Ogljišča po uporabi nonmax.

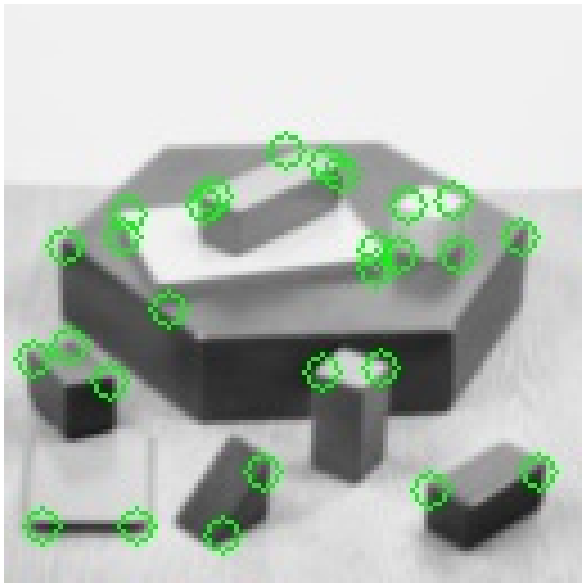
Slika 3: Piksli ključnih točk so označene z zeleno barvo. Pragovna vrednost je znašala 17.

2.3 Točke različnih velikosti

Pri iskanju oglišč v različnih velikostih sem vpleljal piramido slik, torej izvorno sliko pomanjšamo (v mojem primeru za faktor 2), kar pomeni, da izgubimo delež podrobnosti in posledično lahko iščemo le najizrazitejša ogljišča. Detekcija poteka na vsaki izmed pomanjšanih slik, na katere se tudi vrišejo zaznane ključne točke. Primerjava točk je vidna na sliki 4



(a) Ogljišča na izvorni sliki (velikost slike je 256×256).



(b) Ogljišča na prvem podnivoju (velikost slike je 128×128).

(c) Ogljišča na drugem podnivoju (velikost slike je 64×64).

Slika 4: Ključne točke so označene kot zeleni krogci s konstantnim polmerom 4 pikslov. Pragovna vrednost je znašala 17, uporabljen je bil tudi nonmax. Opazimo, da se med sliko 4a in 4b izgubi nekaj ogljišč (izrazitejša obstanejo), slednje pa opazimo tudi med slikama 4b in 4c (npr. zgornji levi kot na belem kvadru).

2.4 Izračun orientacije točk

Izračun orientacije točk poteka preko centroida intenzitete (angl. *intensity centroid*), pri čemer skušamo izračunati center intenzitete pikslov v krožnem območju okoli opazovane točke. Vektor smeri označen na slikah je potem daljica iz središča ključne točke do izračunanega centra na območju okoli točke. Na sliki 5 je prikazan primer orientacije ključnih točk zaznanih na rotiranih slikah.

(a) Ključne točke in njihove orientacije na testni sliki.

(b) Ključne točke in njihove orientacije na isti sliki rotirani za 45 stopinj.

Slika 5: Primerjava ohranitve orientacije ključnih točk (označene kot zelene puščice) ob rotiranju slike za kot 45 stopinj v obratni smeri urinega kazalca. Orientacije so se ohranile v večini od skupaj zaznanih točk. Uporabili smo okroglo okno velikosti 31 pisklov.

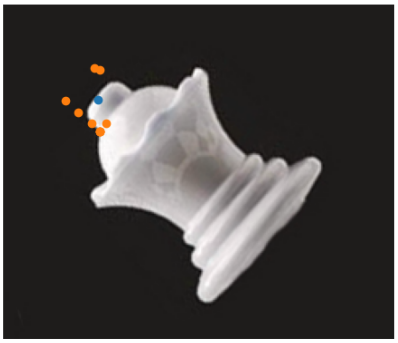
2.5 Izračun orientacije opisa

Za prikaz orientiranega opisa sem uporabil izrez iz slike šahovskih figur 8 in jo večkrat rotiral. Pri obeh slikah sem ročno poiskal ujemajoče ključne točke z algoritmom FAST, in zanje nato poračunal orientacijo nekaj naključnih BRIEF točk. BRIEF točke se generirajo naključno, rezultati dveh zagonov so vidni na slikah 6 in 7.

(a) Naključne BRIEF točke poravnane glede na orientacijo FAST točke.

(b) Identične BRIEF točke iz slike 6a poravnane na orientacijo ključne točke v rotirani sliki.

Slika 6: Primer naključnih BRIEF točk (obarvane v oranžnem odtenku) poravnanih glede na orientacijo ključne točke (obarvane modro) na izseku vrha dame rotiranega za 30 in 35 stopinj.



(a) Naključne BRIEF točke poravnane glede na orientacijo FAST točke.

(b) Identične BRIEF točke iz slike 7a poravnane na orientacijo ključne točke v rotirani sliki.

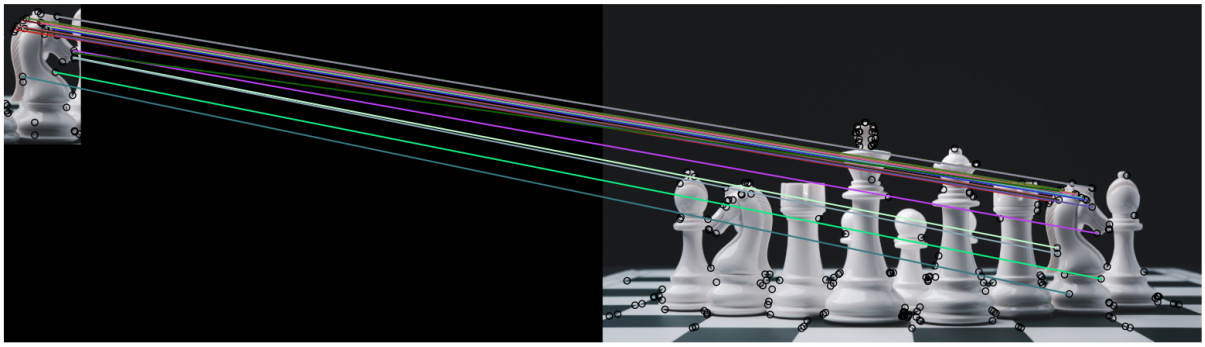
Slika 7: Ponoven primer naključnih BRIEF točk (obarvane v oranžnem odtenku) poravnanih glede na orientacijo ključne točke (obarvane modro) izseku vrha dame rotiranega za 30 in 50 stopinj.

2.6 Primerjava točk

Pri primerjavi točk sem uporabil sliko šahovskih figur 8, iz katere sem nato izrezal odseke z urejevalnikom slik in jih z implementiranim algoritmom ORB skušal umestiti nazaj v izvorno sliko. Za učinkovitejše delovanje je priporočljivo, da se kot prva slika vzame izsek, saj algoritem primerja vsako ključno točko iz prve slike in jih išče najbližjo primerjavo v drugi sliki. Algoritem je prav tako zasnovan, da pri primerjavi točk upošteva prag najvišjega razlikovanja, ki je uporabniško določen. V kolikor najmanjša (Hammingova) razdalja med opazovano ključno točko na prvi sliki in vsemi ključnimi točkami na drugi sliki preseže ta prag, se smatra, da ključna točka nima ustreznega zadetka. Na slikah 9, 10 in 11 so vidni rezultati poskusov. Za vsak poskus so bili uporabljeni parametri: velikost okna 31, velikost deskriptorja 256, FAST prag 25, največja razdalja za ujemanje 35.

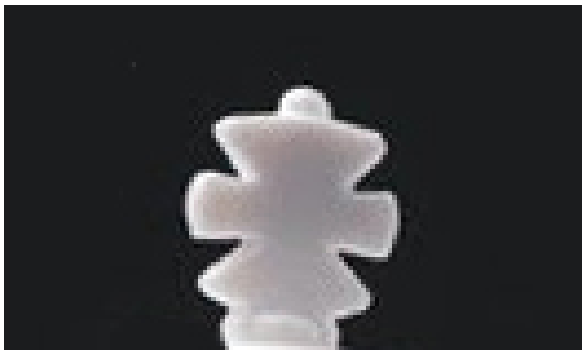
Slika 8: Slika šahovskih figur, osnova za primerjavo ključnih točk.

(a) Izrezan in rotiran vrh kraljevske figure.

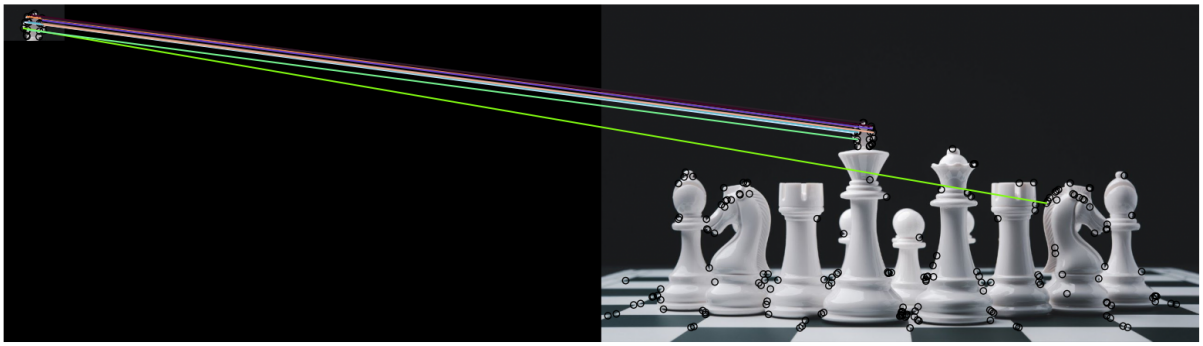


(b) Ujemanje konja s prvotno sliko.

Slika 9: Iz prvotne slike izrezan konj in njegovo ujemanje ključnih točk. Ključne točke so se preslikale na isti objekt, vendar z nekoliko zamaknjenimi pozicijami.



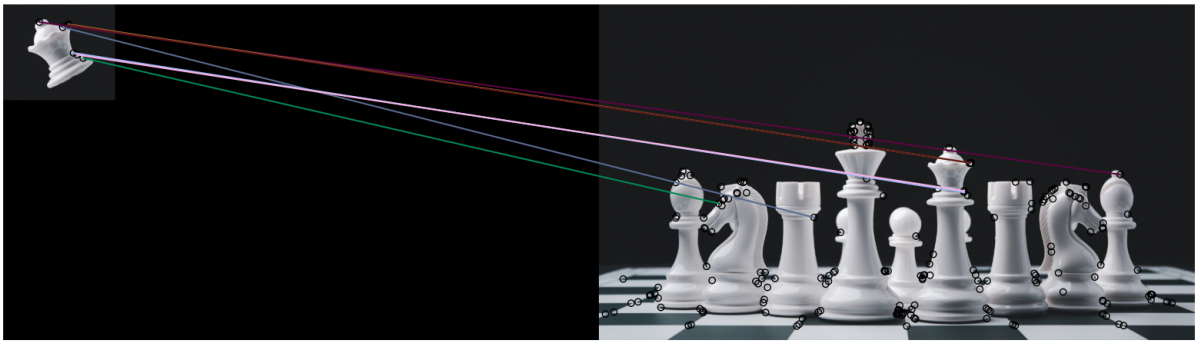
(a) Izrezan in rotiran vrh kraljevske figure.



(b) Ujemanje modificiranega vrha s prvotno sliko.

Slika 10: Primerjava ključnih točk na izrezani in malo rotirani kroni iz kraljevske figure. Ena izmed točk se je preslikala na konja, ostale pa so zadele prvotni objekt.

(a) Izrezan, rotiran in povečan vrh dame.



(b) Ujemanje modificiranega vrha s prvotno sliko.

Slika 11: Primerjava ključnih točk na izrezanem, rotiranem (30 stopinj) in skaliranem (10% povečava) vrhu dame. Točke so zadele damo, preslikale pa so se tudi na top, tekača in konja. Sklepam, da so ključne točke, ki so se napačno preslikale precej podobne, saj so šahovske figure stilsko usklajene.