Univerzitet u Nišu
Prirodno-Matematički fakultet
Departman za računarske nauke

Faruk B. Selimović

PRIMENA VORONOI-DELONE TRIANGULACIJA I KATALANOVIH OBJEKATA U ZAŠTITI PODATAKA

Doktorska disertacija

Niš, 2021.

University of Niš
Faculty of Sciences and Mathematics
Department of computer science

Faruk B. Selimović

APPLICATION OF VORONOI-DELAUNAY TRIANGULATION AND CATALAN OBJECTS IN DATA PROTECTION

PhD thesis

Niš, 2021.

Imam posebnu čast i zadovoljstvo da se zahvalim mentoru prof. dr Predragu Stanimiroviću, redovnom profesoru Prirodno-matematičkog fakulteta u Nišu, na nesebičnoj pomoći i brojnim korisnim savetima kojima je doprineo boljem kvalitetu ove disertacije.

Zahvaljujem se svim profesorima PMF-a na korektnoj saradnji u toku doktorskih studija a posebno prof. dr Predragu Krtolici na izradi zajedničkih naučnih radova.

Veliku zahvalnost dugujem i prof. dr Muzaferu Saračeviću, redovnom profesoru fakulteta za primenjeni menadžment, ekonomiju i finansije (MEF) u Beogradu i prorektoru za nauku Univerziteta u Novom Pazaru na iskrenoj podršci, dobrim idejama i savetima, naročito u presudnim momentima izrade i objavljivanja naučnih radova.

Zahvalnost dugujem i prof. dr Danijeli Milošević, redovnom profesoru Fakulteta tehničkih nauka u Čačku, Univerziteta u Kragujevcu, na nesebičnoj saradnji i pomoći tokom doktorskih studija.

Posebno se zahvaljujem članovima moje porodice na podršci, razumevanju i ohrabrenjima, koji su bili glavni razlog mog istrajanja u radu.

Подаци о докторској дисертацији

Ментор:	Проф. Др Предраг С. Станимировић, редовни професор, Универзитет у Нишу, Природно-математички факултет
Наслов:	Примена Воронои - Делоне триангулација и Каталанових објеката у заштити података
Резиме:	Ова докторска дисертација је из области рачунарске геометрије у комбинацији са методама за заштиту података, са акцентом на имплементацију апликација техникама објектно оријентисаног програмирања. Представљене су нове методе за заштиту података помоћу Воронои - Делоне триангулација. Прва метода заснована је на енкрипцији 3Д равни у ГИС-у. Друга метода је заснована на аутентификацији корисника (клијената) банака, енкрипцијом слике помоћу инкременталног алгоритма Воронои - Делоне триангулација и Каталанових објеката. Трећа метода, је метода стеганографије која се темељи на комбинацији Каталанових објеката и Воронои - Делоне триангулације. Четврта метода се односи на предлог примене једног алгоритма рачунарске геометрије у поступку генерисања Криптографских кључева из једног сегмента 3Д слике.
Научна област:	Рачунарске науке
Научна дисциплина:	Рачунарска геометрија, Објектно - оријентисано програмирање
Кључне речи:	Воронои - Делоне триангулација полигона, Јава програмирање, криптографија, стеганографија
УДК:	004.4, 004.415.24/.26, 004.415.3
CERIF класификација	P150, P170
Тип лиценце Креативне заједнице	**CC BY-NC-ND**

Data on Doctoral Dissertation

Doctoral Supervisor:	Predrag S. Stanimirović, Ph.D., full professor at Faculty of Sciences and Mathematics, University of Niš
Title:	Application of Voronoi-Delaunay triangulation and Catalan objects in data protection
Abstract:	This doctoral dissertation is in the field of computer geometry in combination with data protection methods, with an emphasis on the implementation of applications using object-oriented programming techniques. New methods for data protection using Voronoi - Delaunay triangulation were presented. The first method is based on 3D plane encryption in GIS. The second method is based on the authentication of users (clients) of banks, by encrypting the image using the incremental algorithm Voronoi - Delone triangulation and Catalan objects. The third method is the method of steganography, which is based on a combination of Catalan objects and Voronoi - Delone triangulation. The fourth method refers to the proposal of application of one algorithm of computer geometry in the procedure of generating Cryptographic keys from one segment of 3D image.
Scientific Field:	Computer science
Scientific Discipline:	Computer geometry, Object - oriented programming
Key Words:	Voronoi - Delaunay polygon triangulation, Java programming, cryptography, steganography
UDC:	004.4, 004.415.24/.26, 004.415.3
CERIF Classification:	P150, P170
Creative Commons License Type:	**CC BY-NC-ND**

Sadržaj

Spisak algoritama

Spisak tabela

Spisak slika

Predgovor

Ova doktorska disertacija je iz oblasti računarske geometrije u kombinaciji sa metodama za zaštitu podataka, sa akcentnom na implementaciju tehnikama objektno-orijentisanog programiranja.

Disertacija se sastoji od novih metoda za zaštitu podatka pomoću Delone triangulacije poligona i Katalanovih objekata. Programski jezik u kome su implementirane navedene metode je Java, koji je uprkos pojavi novih modernih programskih jezika, i dalje u vrhu liste najzastupljenijih.

Disertacija je organizovana u 5 poglavlja. U nastavku biće predstavljen kratak opis svakog od poglavlja.

1. **Prvo poglavlje** sastoji se od metodološkog pristupa istraživanju (ciljevima, problemu, tehnikama istraživanja). Predstavljen je pojam Katalanovih brojeva, odnosno, razlika između Katalanovog broja i objekta. Takođe, predstavljen je u kratikim crtama Voronoi dijagram i Delone triangulacija poligona. Urađena je analiza već postojećeg *inkrementalnog algoritma* Delone triangulacije koji je kasnije modifikovan u predstavljenim metodama zaštite podataka.

2. **Drugo poglavlje** je rezervisano za svojstva Katalanovih brojeva i njihovih objekata i kombinatornih problema u šifrovanju podataka. Objašnjen je pojam bit-balansiranosti kao krucijalnog atributa Katalanovih objekata. Analizirana je metoda kretanja kroz celobrojnu rešetku *Lattice Path* sa primerom šifrovanja koordinata (x,y) temena Delone triangulacije. Razmotrena je i metoda *stek permutacije* bitova takođe upotrebom Katalanovih objekata. Pored navedenih svojstava predstavljene su i metode *ballot notacije* i uparenih zagrada *(eng. Balanced Parentheses)* takođe na primeru šifrovanja koordinata (x,y) temena Delone triangulacije.

3. **Treće poglavlje** obrađuje *Java* programski jezik. Izučavane su prednosti i mogućnosti objektno-orijentisanog programiranja kroz primenu postojećih biblioteka i programskih interfejsa kao što su (*Java 2D, 3D, Triangulation, Pnt, Triangle, Logic*). Aplikacije koje su navedene u disertaciji karakteristične su po tome što su korišćene klase iz navedenih *Java* klasa, odnosno iz *AWT* i *Swing* paketa. Pored navedenih paketa i interfejsa predstavljena je i *JavaFX* programska platforma koja je naslednica *Swing* paketa. Navedene su neke prednosti *JavaFX* platforme. Na primer, napisan

kao *Java API*, *JavaFX* aplikacioni kod, može se pozivati na *API-je* iz bilo koje Java biblioteke. Drugim rečima *JavaFX* programi mogu koristiti *Java API* biblioteke za pristup izvornim mogućnostima sistema i povezivanje sa serverskim aplikacijama.

4. **Četvrto poglavlje** predstavlja opisane nove metode zaštite podataka pomoću Voronoi - Delone triangulacije i Katalanovih objekata. Prva metoda temelji se na enkripciji 3D ravni u GIS-u. Druga metoda je zasnovana na autentifikaciji korisnika (klijenata) banaka, enkripcijom slike pomoću inkremantalnog algoritma Voronoi - Delone triangulacijije i Katalanovih objekata. Treća metoda, je metoda steganografije koja se temelji na kombinaciji Katalanovih objekata i Voronoi-Delone triangulacije. Četvrta metoda odnosi se na predlog primene jednog algoritma računarske geometrije u postupku generisanja Kriptografskih ključeva iz jednog segmenta 3D slike. Objavljeni naučni radovi autora disertacije na kojima se baziraju navedene metode su [10, 32, 47, 65]

5. **Peto poglavlje** odnosi se na zaključna razmatranja i kratak pregled rezultata izloženih poglavlja, kao i budući pravci istraživanja. Za tri navedena metoda razvijene su aplikacije čiji ključni delovi koda su dati u prilogu disertacije.Takođe, na kraju disertacije su navedeni spiskovi svih algoritama, tabela i slika.

Poglavlje 1

Uvodna razmatranja

Moderno elektronsko doba i uređaji koji ga prate doneli su za sobom mnogobrojne mogućnosti. Jedna od njihovih glavnih mogućnosti jeste skladištenje podataka i njihova zaštita od neovlašćenog pristupa. Nauka koja se bavi metodama zaštite podataka naziva se kriptografija. Uporedo sa kriptografijom razvila se i kriptoanaliza, koja oduvek teži da sazna tajnu poruku dobijenu nekom od kriptografskih metoda. U suštini kriptografija se zasniva na matematičkim modelima algoritama.

Primena operacija takvih modela algoritama nad početnom (javno dostupnom) informacijom naziva se *"šifrovanje"*. Ovim postupkom dobijamo modifikovanu (šifrat) informaciju koja nije razumljiva. Obrnut, inverzan proces, kada se od šifrata ponovo dobija razumljiva informacija, naziva se *"dešifrovanje"*. Jedan od nezaobilaznih ulaznih parametara u izabranom algoritmu šifrovanja jeste kriptografski ključ. On predstavlja binarni niz nula i jedinica čija dužina zavisi od samog kriptografskog algoritma koji se koristi [10]. Obzirom na sve veću ulogu GIS-a *(Geografski Inforamcioni Sistemi)* u obradi i analizi prostornih podataka, kao i u komandnim i kontrolnim sistemima odbrane i javne bezbednosti, *Delanay trangulacija* predstavlja osnovni model za kreiranje mreže nepravilnih trouglova TIN *(Triangulated Inregular Network)* u procesu dobijanja digitalnog modela terena (DMT) [1]. To je u stvari organizovani skup podataka o visinama terena zapisan u digitalnom obliku.

U današnjoj eri modernih tehnologija javlja se sve veća potreba za kreiranjem sigurnih tj. pouzdanih sistema autentifikacije korisnika. Poseban akcenat se odnosi na bankarske transakcije i sisteme koji su često meta hakerskih napada. Pored analize i obrade prostornih podataka u GIS-u odnosno, kriptovanju TIN modela, u ovoj disertaciji predstavljena je i metoda enkripcije slike Voronoy - Delanay triagulacijom i Katalanovim objektima u procesu autentifikacije korisnika na bankarski sistem. Predstavljen je scenario u kome će biti uključen i potencijalni napadač. Glavni doprinos ove metode šifrovanja je to što koristi inkrementalni algoritam Delone triangulacije i Katalanovih objekata. Predložena metoda je kombinacija računarske geometrije i kriptografije. Ova metoda predstavlja novi korak prema kodiranju koordinata temena trougla pomoću Katalanovog ključa. Dati su konkretni predlozi za primenu ove metode u proveri autentičnosti za klijente banke šifrovanjem slike.

Mnogo je prednosti ove vrste implementacije scenarija za autentifikaciju korisnika dodeljenim Katalanovim objektom i metodom stek permutacije.

Kada se govori o zaštiti privatnosti podataka na slici, odnosno skrivanju poruke unutar slike, može se reći da je to novi izazov u Cyber prostoru. Slika kao nosilac tajne poruke i njen dizajn, šema šifrovanja, stvaranje algoritama i metoda privukli su pažnju mnogih istraživača. Stoga su u [25] autori pomenuli mnoge tehnike korišćene u šifrovanju slike sa kojima se napadači susreću. Neke od tih tehnika su Fridrichove sheme šifrovanja slike, difuzija bez okvira i tako dalje. Za razliku od spomenutih tehnika, predložena tehnika temelji se na kombinaciji Delone triangulacije binarnog zapisa slike, Katalanovih objekata (kao ključa) i stek permutacije kao metode šifrovanja.

U okviru disertacije, predstavljena je i metoda kojom se sprečava potencijalni napadač da pristupi tajnoj poruci, tako što je nosilac tajne poruke slika koja neće promeniti svoj izvorni oblik nakon instaliranja tajne poruke. Glavna motivacija autora bila je kako sakriti poruku na slici i ne menjati joj oblik. Zatim, predstavljen je postupak rastavljanja stego ključa kako bi napadač bio zbunjen prilikom kreiranja metode napada. U analizi se razmatra kako napadač može dobiti tajnu poruku čak i ako u nekim slučajevima uspe povezati delove u stego ključu. Metoda koja je predstavljena ima za cilj korišćenje slike za prenos skrivenih podataka željenom korisniku putem interneta. Ovaj algoritam steganografije, u procesu šifrovanja, temelji se na kodiranju slike u binarni zapis, pretvaranju tajne poruke (skrivenih podataka) u binarni zapis i stvaranju Delone triangulacije binarnog zapisa slike čija su temena u stvari nosači bita tajne poruke.

Pored navedenih metoda u ovoj disertaciji je takođe predložena metoda primene jednostavnog algoritma triangulacije poligona u procesu generisanja skrivenih kriptografskih ključeva iz jednog segmenta 3D slike. Ova metoda utvrđuje važnost računarske geometrije (poligonske triangulacije) i Katalanovih objekata koji postoje u kriptografiji, pre svega u razvoju algoritama za generisanje binarnih sekvenci potrebnih za generisanje ključeva. Tema ove disertacije je multidisciplinarna što znači da dotiče oblasti računarske grafike, geometrije, programiranja i zaštite podataka.

Predmet istraživanja disertacije jesu mogućnosti, svojstva i primena Katalanovih brojeva i Katalanovih objekata u procesu generisanja ključeva u enkripciji 3D ravni triangulisanoj sa Voronoy - Delone triagulacijom kao i primena Katalanovih objekata u procesu enkripcije Voronoi - Delone triangulisane slike. Takođe predmet istraživanja je bila i metoda skrivanja informacija u slici primenom Voronoi - Delone triangulacije binarnog zapisa slike.

Cilj ove disertacije je razmatranje i obrazloženje postojećih znanja o primeni Katalanovih brojeva i kombinatornih problema u postupku šifrovanja i dešifrovanja TIN mreže 3D ravni i u procesu autentifikacije na bankarski sistem kao i skrivanje (steganizaciji) informacije u slici. Cilj je i da se primene neke metode, poput dekompozicije Katalanovih brojeva i šifrovanja na osnovu problema puteva u celobrojnoj mreži (LatticePath).

1.1 Pojam Katalanovih brojeva i Katalanovih objekata

Katalanovi brojevi (C_n) predstavljaju niz brojeva koji se najčešće koriste u geometriji. Takođe, pojavljuju se i kao rešenje velikog broja kombinatornih problema. Leonard Ojler (*Leonhard Euler*, 1707-1783) je prvi pronašao ove brojeve tražeći opšte rešenje za broj različitih načina na koji se može podeliti jedan mnogougao na trouglove. Pritom je trebalo voditi računa da se dijagonale mnogougla koje se međusobno seku ne koriste [53].

Belgijski matematičar Ežen Šarl Katalan (*Eugene Charles Catalan*, 1814–1894) je prvi koji je otkrio vezu između ovih brojeva i problema korektnih nizova n parova zagrada, tako da su ovi brojevi upravo po njemu dobili ime.

Katalanovi brojevi se izračunavaju po sledećoj formuli [2]:

$$\begin{aligned} C_n &= \frac{(2n)!}{(n+1)!n!} \\ &= \frac{1}{n+1}\begin{pmatrix} 2n \\ n \end{pmatrix}, n \geq 0 \end{aligned} \tag{1.1}$$

gde važi da je n broj trouglova na koje se može podeliti dati poligon.

Često se koristi i sledeći alternativni izraz za definiciju Katalanovih brojeva:

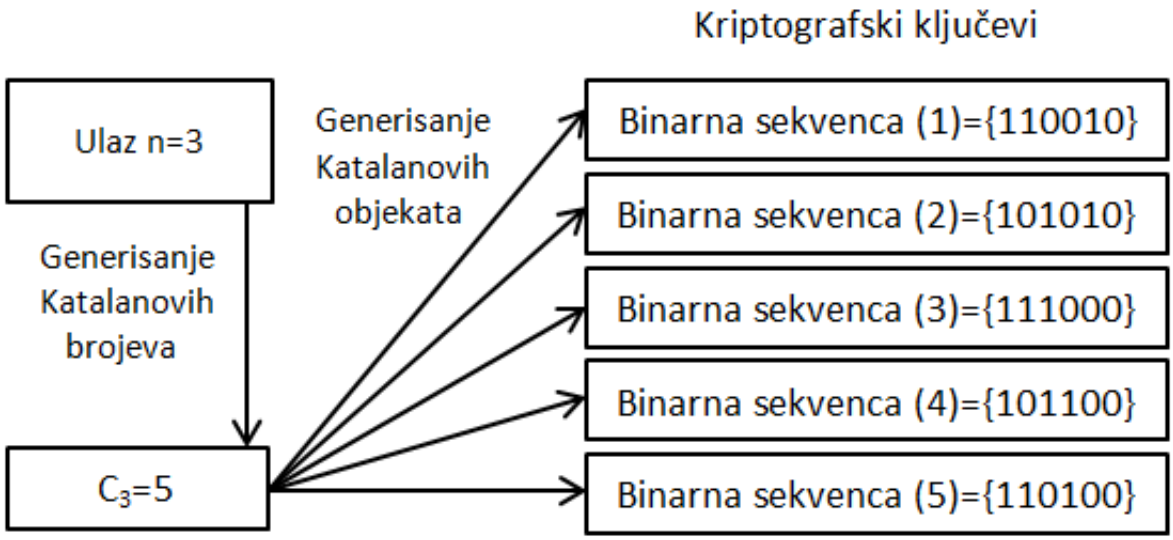
$$\begin{aligned} \begin{pmatrix} 2n \\ n \end{pmatrix} - \begin{pmatrix} 2n \\ n+1 \end{pmatrix} &= \frac{(2n)!}{(n!)^2} - \frac{(2n)!}{(n-1)!(n+1)!} \\ &= \frac{(2n)!}{n!(n+1)!} \\ &= C_n \end{aligned} \tag{1.2}$$

Tabela 1.1 sadrži Katalanove brojeve za vrednosti $n \in \{1, 2, ..., 30\}$, koji se izračunavaju po formuli (1.1) ili (1.2).

Tabela 1.1: Vrednosti prvih 30 Katalanovih brojeva

n	C_n	n	C_n	n	C_n
1	1	**11**	58,786	**21**	24,466,267,020
2	2	**12**	208,012	**22**	91,482,563,640
3	5	**13**	742,900	**23**	343,059,613,650
4	14	**14**	2,674,440	**24**	1,289,904,147,324
5	42	**15**	9,694,845	**25**	4,861,946,401,452
6	132	**16**	35,357,670	**26**	18,367,353,072,152
7	429	**17**	129,644,790	**27**	69,533,550,916,004
8	1,430	**18**	477,638,700	**28**	263,747,951,750,360
9	4,862	**19**	1,767,263,190	**29**	1,002,242,216,651,360
10	16,796	**20**	6,564,120,420	**30**	3,814,986,502,092,300

U ovoj disertaciji Katalanove brojeve koristićemo u svojstvu generatora ključeva za šifrovanje i dešifrovanje koordinata tačaka 3D ravni u GIS-u, kako i šifrovanju i dešifrovanju koordinata bilo koje slike triangulisane Delone triangulacijom u procesu autentifikacje korisnika sa bankarskim sistemom. Na osnovu Tabele 1.1 možemo primetiti da povećanjem osnove, raste i prostor ključeva. Ako uzmemo u obzir da za svaki Katalanov broj dobijamo odredjeni broj ključeva, u nastavku rada za svaki ključ koristićemo terimin Katalanov objekat. Sledeća Slika 1.1 ilustruje razliku između Katalanovog broja i objekta.



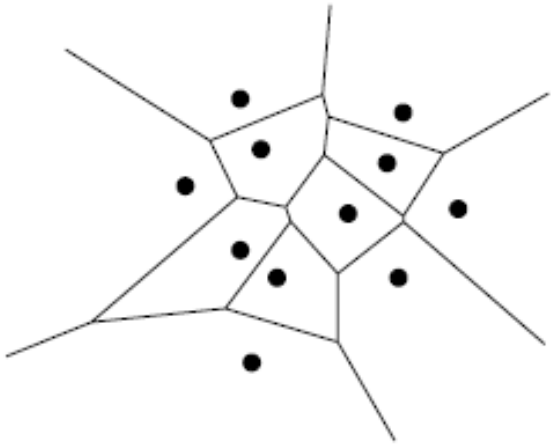
Slika 1.1: Generisanje Katalanovih ključeva (objekata) $n = 3$

1.2 Pojam Voronoi dijagrama i Delone triangulacije

Voronoi dijagram je geometrijska struktura koja je često korišćena i u čijoj je osnovi pitanje bliskih tačaka datoj tački. Drugim rečima, Voronoi dijagram podrazumeva podelu ravni na ćelije, tako da je svakoj tački unutar jedne ćelije najbliži onaj centar koji se nalazi u toj ćeliji. Mnoge prirodne i društvene pojave objašnjavaju se primenom ovog dijagrama.

Najpoznatiji problem u geometriji koji se može rešiti primenom Voronoi dijagrama je tzv. *poštanski problem*. Predpostavimo da postoji n pošta u gradu tako da svaki korisnik koristi onu koja mu je najbliža. Postavlja se pitanje kako izgleda oblast koju pokriva jedna pošta? Takođe, ako postoji potreba da se otvori nova pošta, gde bi najbolje bilo da se ona nalazi? Na Slici 1.2. je predstavljen Voronoi dijagram tako da svaka ćelija ima svoj centar (poštu).

Predpostavimo da je dat skup tačaka u ravni: $\{p_1, p_2, ..., p_n\}$ koje nazivamo "sedišta" ili centri. Potrebno je odrediti za svaku tačku ravni koje joj je sedište najbliže. Rastojanje između dve tačke je obično euklidsko d , tako da važi sledeća funkcija.



Slika 1.2: Voronoi diagram

$$d(p,q) = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2} \tag{1.3}$$

Oblast čini skup tačaka kojima je jedno isto sedište najbliže tj, relacija : "dve tačke su u relaciji ako i samo ako im je isto sedište najbliže" je relacija ekvivalencije tako da deli ravan na skup oblasti. Granice oblasti (ćelije) čine tačke koje imaju više od jednog najbližeg sedišta. Za funkciju koja svakoj tački ravni pridružuje najbliže sedište, kažemo da je funkcija Voronoi dodele.

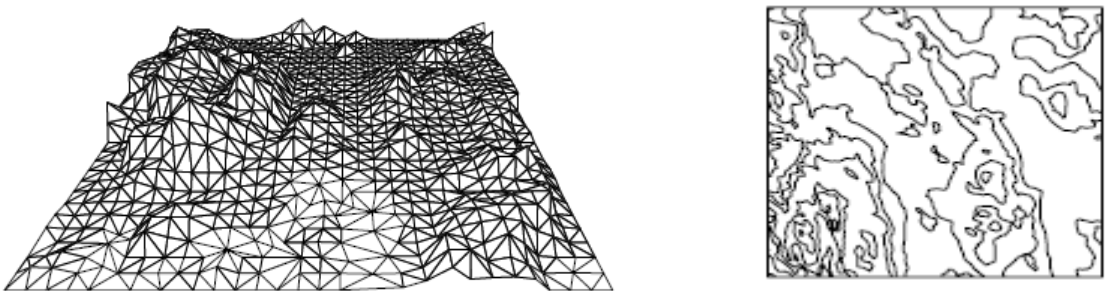
Def. 1 (Voroni dijagram). Voronoi dijagram Vor(P) skupa tačaka P= $\{p_1, p_2, ..., p_n\}$ je podela ravni na oblasti (regione ili ćelije) takve da tačka X pripada oblasti tačke p_i ako i samo ako važi:

$$d(X,\ p_i) < d(X,\ p_y) \text{ za svako } i \neq j$$

Oblast tačke p_i se naziva Voronoi ćelija tačke p_i i označava se sa V(p_i)

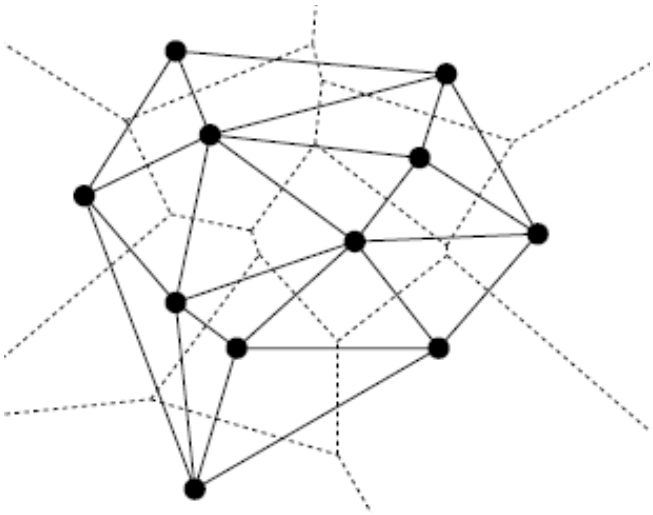
U nastavku vam predstavljamo Delanay dijagram (triangulaciju) koji je dualan Voroni dijagramu [5].

U prethodnoj sekciji predstavljen je Voronoi dijagram čije su ćelije predstavljene u 2D dimenziji. Međutim, ako modelujemo površinu zemlje, kao što je interpolacija terena u GIS-u, neizostavno je da se uključi treća dimenzija tj. *visina* tačaka, odnosno centara. Karakteristično za modelovanje terena je to što svaka vertikalna linija preseca teren najviše u jednoj tački. To je u stvari graf funkcije *f:* $A \subset R^2 \rightarrow R$ koja dodeljuje visinu *f(p)* svakoj tački iz domena A. Na Slici 1.3. predstavljamo vizuelizaciju terena perspektivnim crtanjem ili izohipsama jednake visine.



Slika 1.3: Pogled iz perspektive terena

Kao što se vidi sa slike, predstavljanje terena perspektivno je u stvari mreža nepravilnih nepreklapajućih trouglova. Za postupak kreiranja ovakve mreže najčešće se koristi Delone triangulacija. Na Slici 1.4. je predstavljen Delone dijagram. Dobio je ime po matematičaru Borisu Nikolajavichu Deloneu.

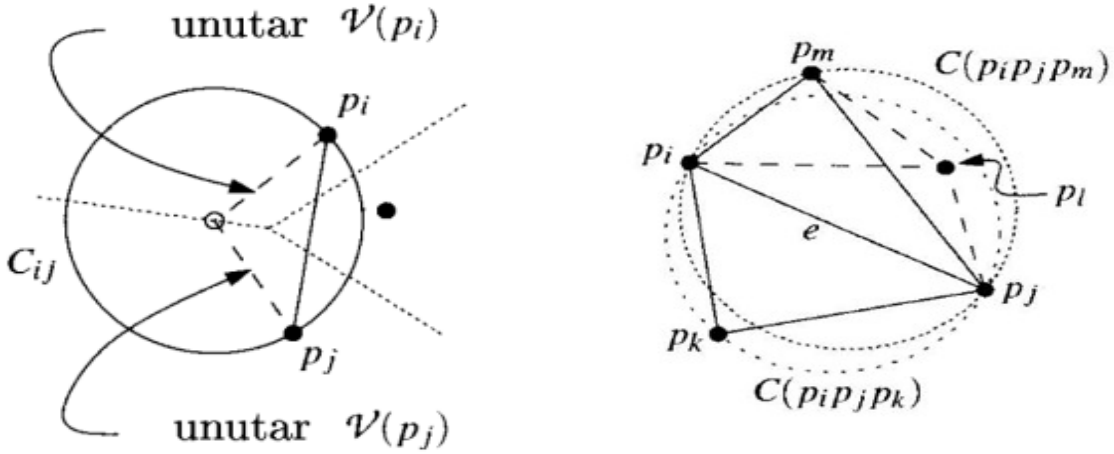


Slika 1.4: Delone dijagram

U procesu predstavljanja način kreiranja Delone triangulacije objasnimo pojam, ivice Delone dijagrama. Za ivicu $\overline{p_i p_j}$ kažemo da pripada Delauny dijagramu ako postoji kružnica C_{ij} kojoj pripadaju p_i i p_j i nijedan drugi centar is P ne leži unutar kružnice, a centar kružnice C_{ij} leži na ivici Voronoi dujagrama definisanoj sa $V(p_i)$ i $V(p_j)$. Na Slici 1.5. je detaljnije objašnjeno.

Neka je P skup tačka u ravni. Tri tačke p_i,p_j,p_k su čvorovi Delone dijagrama skupa P ako i samo ako kružnica kroz tačke p_i,p_j i p_j ne sadrži druge tačke iz P unutar kruga. Iz ove tvrdnje važi sledeća definicija Delone triangulacije.

Def 2. Za triangulaciju $\mathcal{T}$ skupa tačaka u ravni P, kažemo da je Delone triangulacija ako i samo ako opisani krug bilo kog trougla u $\mathcal{T}$, ne sadrži tačke iz P unutar kruga. Ovako definisana triangulacija se još i naziva *legalna Delone triangulacija*



(a) Ivica Delone dijagrama

(b) Delone triangulacija

Slika 1.5: Delone ivica i triangulacija

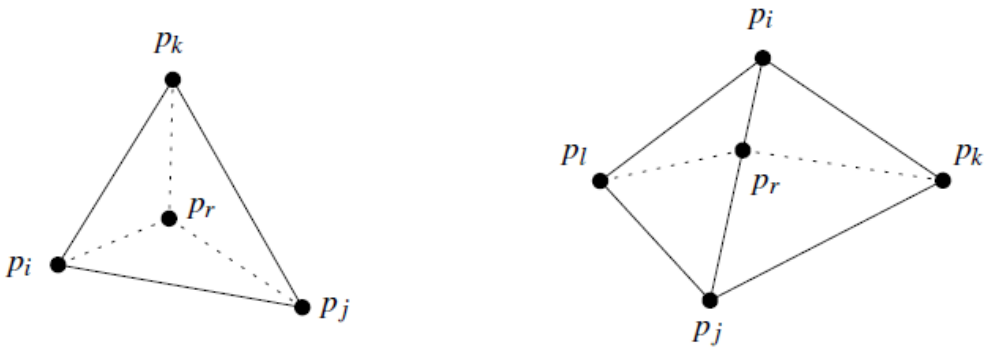
Na Slici 1.5. je predstavljena Delone triagulacija. Može se primetiti da se unutar kružnica $C(p_i p_j p_m)$ i $C(p_i p_j p_k)$ formiraju legalne triangulacije $\triangle p_i p_j p_m$ i $\triangle p_i p_j p_k$. Ukoliko se desi da se u procesu kreiranja Delone triangulacije slučajno pojavi neka tačka (teme) kao u ovom slučaju teme p_l (Slika 1.5.b), onda bi ove triangulacije bile nelegalne Delone triangulacije. Ukoliko ih želimo legalizovati potrebno je da se ukloni duž $\overline{p_m p_j}$ i da se formira nova $\overline{p_l p_i}$ [6]. Ovaj postupak legalizacije najbolje opisuje Inkrementalni algoritam konstrukcije Delone triagulacije o kome će biti više reči u narednoj sekciji.

1.3 Konstrukcija Delone triangulacije - inkrementalni algoritam

Inkrementalni algoritam Delone triangulacije skupa P započinje formiranjem velikog trougla $p_{-1}\ p_{-2}\ p_{-3}$ koji sadrži sve tačke skupa P. Na osnovu navedenog znači da Delone triangulaciju računamo za skup $P \cup \Omega$ gde je $\Omega{:=}\{p_{-1},\ p_{-2},\ p_{-3}\}$. Bitno je napomenuti da se tačke skupa Ω naknadno odbacuju, kao i ivice koje ih spajaju. Potrebno je tačke p_{-1}, p_{-2}, p_{-3} odabrati dovoljno daleko tako da se ne promeni nijedan trougao Delauny triangulacije skupa T. Tačke se dobijaju slučajnim izborom, što predstavlja glavnu prednost u procesu interpolacije reljefa, a algoritam odražava Delone triangulaciju trenutnog skupa tačaka. U suštini postoje dva slučaja pojavljivanja tačaka unutar "najvećeg trougla" [7].

U prvom slučaju tačka p_r se pojavljuje unutar trougla tako da se, nakon pronalaska trougla trenutne triangulacije koji sadrži tačku p_r, dodajemo ivice od p_r ka temenima trougla. Drugi slučaj jeste kada se tačka p_r pojavljuje na ivici već formiranog trougla. U tom slučaju dodaju se ivice od tačke p_r ka suprotnim temenima od ivice gde se p_r pojavila. Na Slici 1.6. su predstavljena dva pomenuta slučaja pojavljivanja tačaka.

U prethodnoj sekciji smo pomenuli pojam "nelegalne triangulacije". Podsećanja radi



Slika 1.6: Slučajevi pojavljivanja tačaka

to je sutuacija kada se unutar kružnice (Slika 1.5), za koji postoji trougao optimalne triangulacije, pojavi tačka tako da takva Delone triangulacija postaje nelegalna. U procesu konstrukcije inkrementalnog algoritma koristićemo algoritam *LegalizujIvicu* koji prebacivanjem, (tj. uklanjanjem postojećih) stranica trouglova, kreira nove stranice, koje se povlače iz novonastalog temena. Na taj način svaka triangulacija postaje legalna. U nastavku predstavljamo algoritam *LegalizujIvicu*.

Algoritam 1.3.1 LegalizujIvicu (p_r, $\overline{p_ip_j}$, $\mathcal{T}$)

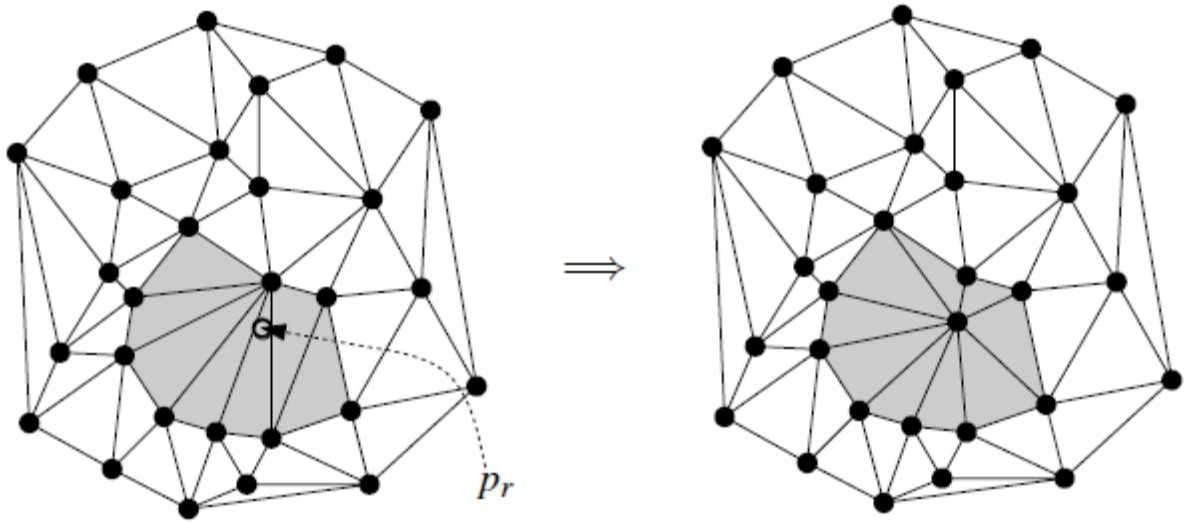
Ulaz: (Taka p_r je tačka koja se ubacuje, a $\overline{p_ip_j}$ ivica triangulacije $\mathcal{T}$, koja bi možda mogla biti prebačena.)

1: **if** je ivica $\overline{p_ip_j}$ ilegalna
then Neka je $p_ip_jp_k$, trougao susedan trouglu $p_ip_jp_r$ duž ivice $\overline{p_ip_j}$.
(Prebaci ivicu $\overline{p_ip_j}$) Zameni je sa $\overline{p_rp_k}$.
LegalizujIvicu (p_r, $\overline{p_ip_k}$, $\mathcal{T}$)
LegalizujIvicu (p_r, $\overline{p_kp_j}$, $\mathcal{T}$)

Ako detaljnije analiziramo rad ovog algoritma, primetićemo da se on rekurzivno poziva i proverava legalnost svake ivice susednog trougla u odnosu na tačku p_r koja se slučajno pojavila. Drugim rečima potrebno je da se obezbedi da svaka ivica bude legalna. Rezultat rada ovog algoritma predstavljen je na Slici 1.7.

Na osnovu strukture stabla locira se tačka p_r tako što se počinje od korena $\mathcal{D}$ (Slika 1.8) koji odgovara trouglu p_{-1}, p_{-2} i p_{-3}. Proverava se u kojem od tri deteta leži p_r dok ne dođemo do lista stabla $\mathcal{D}$. Što se tiče lociranja tačaka p_{-1}, p_{-2} i p_{-3}, one se lociraju dovoljno daleko tako da ne utiču na Delone triangulaciju skupa P.

Ove tačke se biraju na sledeći način, p_{-1}:=(3M, 0), p_{-2}:=(0,3M) i p_{-3}:=(-3M,-3M), gde je M maksimalna apsolutna vrednost bilo koje koordinate tačaka u P. Ovim je osigurano da je P sadržan unutar trougla $p_{-1}p_{-2}p_{-3}$. Broj trouglova kreiranih algoritmom Delone triangulacije je najviše $9n + 1$, a vremenska složenost ovog algoritma je $\mathcal{O}(n \log n)$ koristeći



Slika 1.7: Legalizovanje susednih ivica tački p_r

$\mathcal{O}(n)$ memorije. U nastavku je izložen detaljan opis rada inkrementalnog algoritma sa kreiranje Delone triangulacija [5].

Algoritam 1.3.2 Delone triangulacija (P)

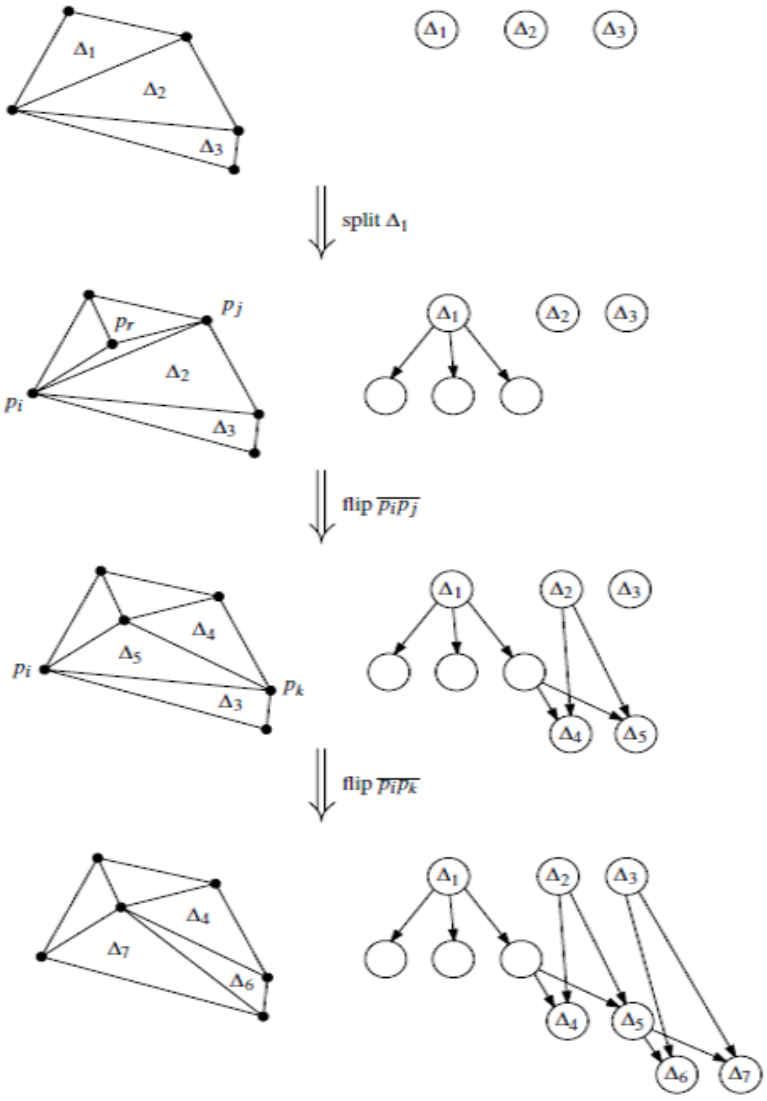
Ulaz: Skup P od n tačaka u ravni. Neka su p_{-1}, p_{-2} i p_{-3} tri tačke trougla unutar koga su sadržane sve ostale tačke skupa P .

1: Početna triangulacija $\mathcal{T}$ sastoji se od trougla p_{-1}, p_{-2} i p_{-3}
2: **for** $r = 1$ to n **do** (Ubaci p_r u $\mathcal{T}$)
3: Pronađi trougao $p_ip_jp_k \in \mathcal{T}$, koji sadrži p_r.
4: **if** p_r leži u unutar trougla $p_ip_jp_k$
then Dodaju se ivice iz p_r ka temenima trougla $p_ip_jp_k$
LegalizujIvicu (p_r, $\overline{p_ip_j}$, $\mathcal{T}$)
LegalizujIvicu (p_r, $\overline{p_ip_k}$, $\mathcal{T}$)
LegalizujIvicu (p_r, $\overline{p_kp_i}$, $\mathcal{T}$)
else (p_r leži na ivici trougla $p_ip_jp_k$, npr. na ivici $\overline{p_ip_j}$). Dodaje se ivica iz p_r ka temenu p_k i prema temenu p_l drugog trougla koji deli ivicu $\overline{p_ip_j}$ sa trouglom $p_ip_kp_j$.
LegalizujIvicu (p_r, $\overline{p_ip_l}$, $\mathcal{T}$)
LegalizujIvicu (p_r, $\overline{p_lp_j}$, $\mathcal{T}$)
LegalizujIvicu (p_r, $\overline{p_jp_k}$, $\mathcal{T}$)
LegalizujIvicu (p_r, $\overline{p_kp_i}$, $\mathcal{T}$)
5: Odbacivanje temena p_{-1}, p_{-2} i p_{-3} i njihove ivice iz triangulacije $\mathcal{T}$
6: **return** $\mathcal{T}$
7: $Izlaz$: Delone triangulacija skupa P.

Na kraju je još potrebno opisati kako se pronalazi trougao koji sadrži tačku p_r i kako se lociraju tačke p_{-1}, p_{-2} i p_{-3} u početnom trouglu. Da bi smo bliže objasnili pronalaženje

trougla koji sadrži tačku p_r, paralelno sa građenjem Delone triangulacije gradićemo i strukturu stabla pozicije tačaka $\mathcal{D}$.

Inicijalno stanje strukture $\mathcal{D}$ je jedan list koji odgovara trouglu p_{-1}, p_{-2} i p_{-3}. Shodno primeru koji je već objašnjen, tačka p_r se slučajno pojavljuje u prvom listu $\triangle_1$ (Slika 1.8). Dodaju se sada u listu nova tri trougla tj, legalizuje se ivica $\overline{p_j p_i}$ [7]. U narednom koraku se još legalizuje ivica $\overline{p_i p_k}$ i na taj način se obezbeđuje da sve triangulacije unutar strukture budu legalne. Na Slici 1.8. je detaljnije objašnjen sam postupak.



Slika 1.8: Struktura stabla $\mathcal{D}$

Poglavlje 2

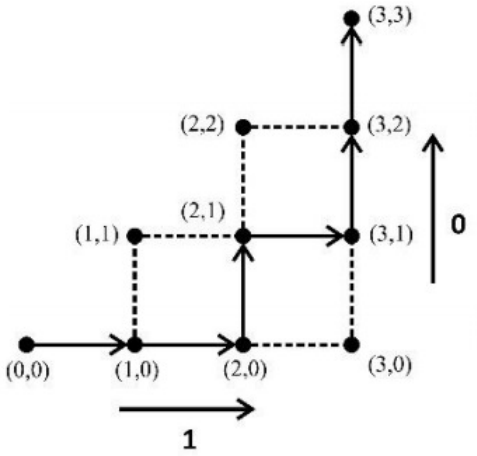
Svojstva Katalanovih brojeva

U ovom poglavlju su razmatrana svojstva Katalanovih brojeva i Katalanovih objekata i kombinatornih problema u šifrovanju podataka. Objašnjen je pojam bit-balansiranosti kao krucijalnog atributa Katalanovih objekata. Analizirana je metoda kretanja kroz celobrojnu rešetku *Lattice Path* sa primerom šifrovanja koordinata (x,y) temena Delone triangulacije. Razmotrena je i metoda *stek permutacije* bitova takođe upotrebom Katalanovih objekata. Pored navedenih svojstava predstavljene su i metode *ballot notacije* i uparenih zagrada *(eng. Balanced Parentheses)* takođe na primeru šifrovanja koordinata (x,y) temena Delone triangulacije [54].

2.1 Primena Katalanovih brojeva i kombinatornih problema u šifrovanju podataka

Krucijalni atribut Katalanovih objekata jeste *bit-balansiranost*. Drugim rečima, svaki Katalanov objekat u svom binarnom zapisu ima isti broj bitova "0" i bitova "1". Takođe, mora počinjati sa bitom "1". Jedan od načina predstavljanja binarne notacije Katalnovog objekta jeste i putem diskretne rešetke ili *LatticePath*, koju čine određeni broj tačaka, u Dekartovom koordinantnom sistemu. Govoreći o problemu diskretne rešetke, možemo reći da se on odnosi na broj izračunavanja puteva kretanja kroz nju. Broj validnih puteva u mreži jednak je Katalanovom broju C_n. Putevi počinju tačkom *(0,0)* i završavaju se *(n,n)* tačkom, a sastoje se od *2n* koraka. Prilikom predstavljanja binarnog zapisa Katalanovog objekta pomoću *Lattice Path* treba voditi računa da se bit 1 predstavlja kao pomeraj udesno, dok se bit 0 predstavlja nagore. Na Slici 2.1 je predstavljen primer jedne kombinacije puta kroz *Lattice Path* za Katalanov objekat *K3=110100*.

Sa slike se može primetiti da se svaka putanja u mreži može kodirati određenim redosledom vektora tako što se pomera udesno sa (1,0) i vektora pomaka na gore (0,1). Određivanje puta u celobrojnoj mreži je u stvari izbor pozicija pomaka udesno (to je od ukupnog broja koraka *2n* oznaka 1) zajedno sa preostalim pozicijama koje predstavljaju pomake na gore (oznaka 0). Drugim rečima, ako se primeni validan Katalanov objekat, navedeno kretanje



Slika 2.1: Kombinacija puta kroz *Lattice Path 3×3* za *K3=110100*

će se obaviti tačno *2n* puta, tako što će početi u tački *(0,0)* i završiti u tački *(n,n)*. Važno je istaći da put nikada ne prelazi njenu dijagonalu i da je glavni uslov da se sa svakim narednim korakom mora biti bliže poslednjoj tački. U slučaju da je $n \geq 0$ broj puteva se može izračunati kao:

$$\begin{pmatrix} n \\ \frac{n+m}{2} \end{pmatrix} - \begin{pmatrix} n \\ \frac{n+m}{2}+1 \end{pmatrix} = \frac{m+1}{\frac{n+m}{2}+1} \begin{pmatrix} n \\ \frac{n+m}{2} \end{pmatrix} \tag{2.1}$$

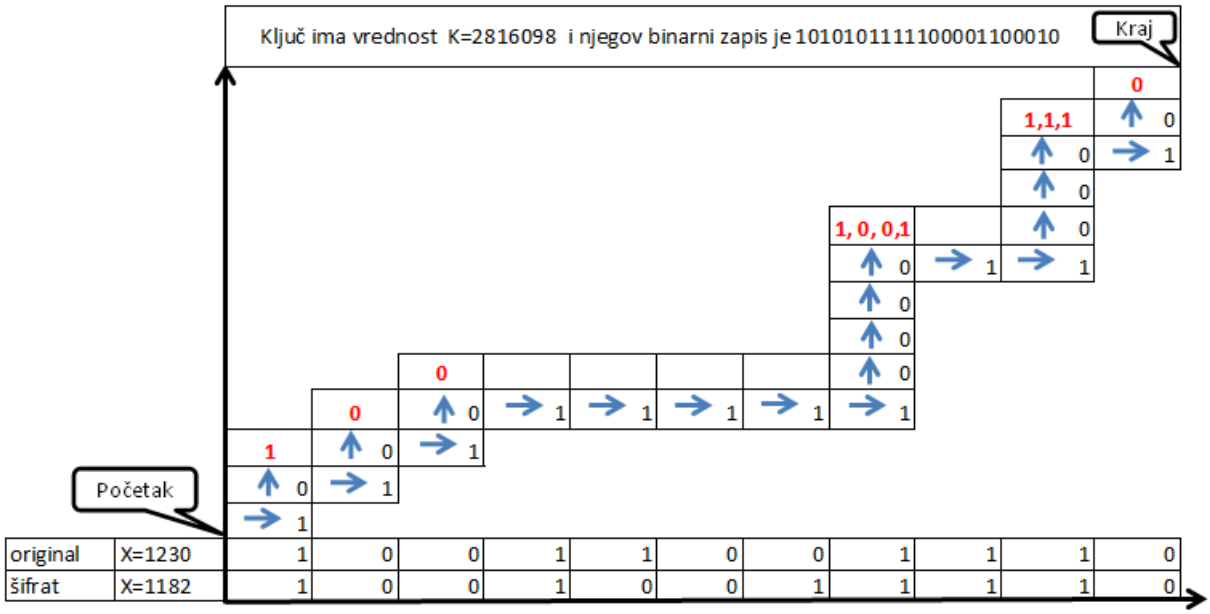
U navedenoj formuli m je broj nepravilnih putanji u mreži. U slučaju da je vrednost $m=0$ dobija se broj pravilnih putanji, odnosno Katalanov broj C_n [13]. Na osnovu prethodno izloženog može se zaključiti da je u mreži dimenzije n broj mogućih puteva definisan Katalanovim brojem C_n. Na osnovu prikazanog postupka kretanja kroz celobrojnu mrežu pomoću binarnog zapisa ključa došli smo na ideju da ispitamo šifrovanje koordinata temena Delone trinagulacije.

Posmatrajući raspored bitova u binarnom zapisu ključa možemo primetiti da elementi imaju dva moguća stanja i to: *Slobodan element* - karakter koji još uvek nije šifrovan, drugim rečima, nije prenet u šifrat i uslovljen je pojavom bita 1 koji čeka na svoj par, a to je bit 0 i *Zauzet element* - to je onaj element koji je šifrovan i koji je prenet u šifrat. Drugim rečima, on je pronašao svoj par i uslovljen je pojavom bita 0. U nastvaku pogledajte deo Java koda (metoda "Binarno Sifrovanje Koordinata") koji realizuje kretanje kroz celobrojnu mrežu.

```
public String Binarno_Sifrovanje_Kordinata
(String Binarna_Vrednost_Koordinate, String Katalan_Kljuc)
  {
  String Vrednost_Sifrovane_Koordinate="";
   //Varijabla za predstavljanje Sifrovane kordinate.
 final int Broj_bita_Konrdinate=11;
```

```
    char[] K = new char[22];
  int n=0, i, slobodan, zauzet, doCilja ;
  char[] Katalan_K=Katalan_Kljuc.toCharArray();
  char[] Koordinata_Realna =
          Binarna_Vrednost_Koordinate.toCharArray();
  char[] Pomocni_Niz = new char[Broj_bita_Konrdinate];
  char[] Sivrovana_Krodinata= new char[Broj_bita_Konrdinate];
       for (int j=0; j<Katalan_Kljuc.length(); j++)
          {
             K[n]=Katalan_K[j];
                 if (n!=21)
                 {
                     n++;
                 }
           }
    slobodan = zauzet =  0;       // Pomocne promeljive
      doCilja=0;
      for (i=0; i<22; i++)
           {
              if (K[i]=='1')
             {
     Pomocni_Niz[slobodan] = Koordinata_Realna[doCilja];
                 slobodan++;
                 doCilja++;
              }
              else
              {
     Sivrovana_Koordinata[zauzet] = Pomocni_Niz[slobodan-1];
                 slobodan--;
                 zauzet++;
               }
           }
Vrednost_Sifrovane_Koordinate= new String(Sivrovana_Krodinata);
      return Vrednost_Sifrovane_Koordinate;
   }
```

Radi boljeg razumevanja moze se posmatrati ključ *K=2816098*. Njegov binarni zapis je 2816098_{10}=1010101111100001100010_2 koji ima *2n=22* bita. Takodje, primer koordinate koju želimo šifrovati je *x=1230*. Njen binarni zapis je 1230_{10}=10011001110_2 koji ima *n=11* bitova. Kretanjem kroz celobrojnu rešetku dobijamo šifrovanu koordinatu *x=1182*, a njen šifrovan binarni zapis je 1182_{10}=10010011110_2 [54].



Slika 2.2: Šifrovanje koordinata kroz *Lattice Path*

Posmatrajući navedenu Sliku 2.2, možemo primetiti da se bitovi x koordinate koju želimo šifrovati uzimaju i smeštaju u šifrat onog trenutka kada se dobije uređen par 1 i 0. Drugim rečima bit ima status "Slobodan" sve dok se ne dobije uređen par 1 i 0. U tom trenutku se bit prenosi u šifrat i samim tim dobija status "Zauzet". Na osnovu predstavljenog na slici mogu se izvesti dva osnovna zaključka: prvi je da kretanje kroz celobrojnu mrežu ne prelazi njenu dijagonalu nikada, dok je drugi činjenica da svakim korakom moramo biti dalji od izvorišta (početka) a bliži odredištu (kraju). U naradnoj Tabeli 2.1 predstavljamo stanja bitova koordinate x=1230_{10} i njenog šifrata.

U slučaju da želimo dešifrovati dobijenu koordinatu x, potrebno je da u obrnutom redosledu čitamo ključ i enkriptovanu koordinatu. U ovom slučaju umesto para (1,0) važi par (0,1), drugim rečima pojava bita 0 označava otvoren par, dok 1 predstavlja zatvoren par.

U cilju dokazivanja validnosti Katalanovog objekta tj. njegove bitbalansiranosti u procesu šifrovanja koordinata temena Delone triangulacije može se uzeti primer ključa koji nema svojstvo Katalanovog objekta tj. nema isti broj bita 1 i bita 0 u svom sastavu. To je ključ *K=2816199* a njegov binarni zapis je 2816199_{10}=$10101011111000110001111_2$. Prilikom kretanja kroz celobrojnu rešetku nailazimo na problem u 20-om koraku, tako što bit iz 20-og koraka nema svoj par. Takođe, biti iz koraka 7 i nadalje 21 i 22 nemaju svoje parove. Na taj način se ne može uspešno završiti kretanje po celobrojnoj rešetki i samim tim se ne može šifrovati koordinata x. U nastavku pogledajte sliku neuspešnog završetka kretanja kroz celobrojnu rešetku za vrednosti ključa koji nije Katalanov objekat (videti Sliku 2.3).

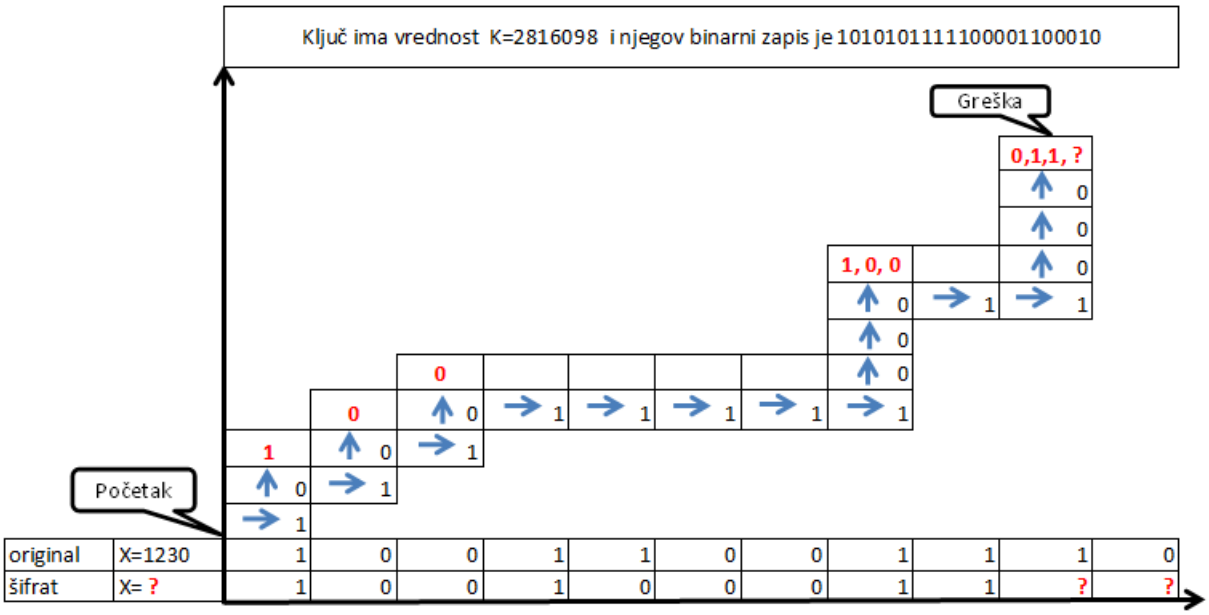
Drugi slučaj je kada se uzme ključ čijim vrednostima se završava put kretanja kroz mrežu, ali se izlazi van opsega dijagonale. Dakle, to je ključ *K=2816099*, a njegov binarni zapis je

Bit u ključu	Koordinata X koja se šifruje (pročitan bit - pojava bita 1)	Šifrat koordiante X (prenet - zauzet bit, pojava bita 0)	Proces brojanja - parametri u metodi "Binarno_Sifrovanje_Koordinata"
1	1		doCilja=21, slobodan=1, zauzet=0
0		1	doCilja=20, slobodan=0, zauzet=1
1	0	1	doCilja=19, slobodan=1, zauzet=1
0		1,0	doCilja=18, slobodan=0, zauzet=2
1	0	1,0	doCilja=17, slobodan=1, zauzet=2
0		1,0,0	doCilja=16, slobodan=0, zauzet=3
1	1	1,0,0	doCilja=15, slobodan=1, zauzet=3
1	1,1	1,0,0	doCilja=14, slobodan=2, zauzet=3
1	1,1,0	1,0,0	doCilja=13, slobodan=3, zauzet=3
1	1,1,0,0	1,0,0	doCilja=12, slobodan=4, zauzet=3
1	1,1,0,0,1	1,0,0	doCilja=11, slobodan=5, zauzet=3
0	1,1,0,0	1,0,0,1	doCilja=10, slobodan=4, zauzet=4
0	1,1,0	1,0,0,1,0	doCilja=9, slobodan=3, zauzet=5
0	1,1	1,0,0,1,0,0	doCilja=8, slobodan=2, zauzet=6
0	1	1,0,0,1,0,0,1	doCilja=7, slobodan=1, zauzet=7
1	1,1	1,0,0,1,0,0,1	doCilja=6, slobodan=2, zauzet=7
1	1,1,1	1,0,0,1,0,0,1	doCilja=5, slobodan=3, zauzet=7
0	1,1	1,0,0,1,0,0,1,1	doCilja=4, slobodan=2, zauzet=8
0	1	1,0,0,1,0,0,1,1,1	doCilja=3, slobodan=1, zauzet=9
0		1,0,0,1,0,0,1,1,1,1	doCilja=2, slobodan=0, zauzet=10
1	0	1,0,0,1,0,0,1,1,1,1	doCilja=1, slobodan=1, zauzet=10
0		1,0,0,1,0,0,1,1,1,1,0	doCilja=0, slobodan=0, zauzet=11

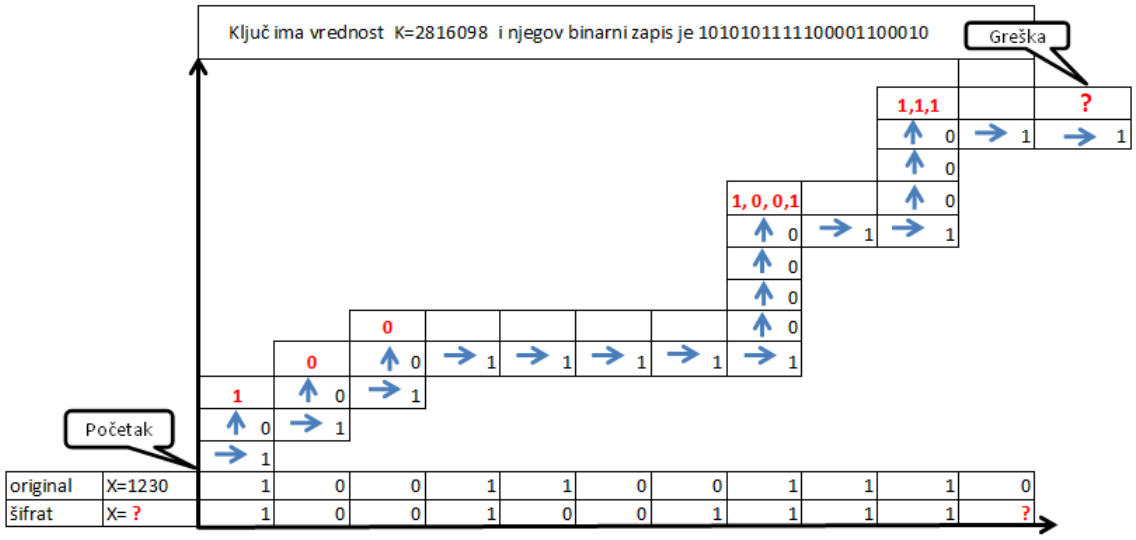
Tabela 2.1: Stanja bita po principu kretanja kroz diskretnu rešetku.

2816099_{10}=1010101111100001100011_2. Ako pokušamo da kretanjem kroz diskretnu rešetku šifrujemo koordinatu X=1230_{10} dobićemo, pored izlaska van dijagonale, takođe problem u koraku 21, jer bit u 21-om koraku nema svoj par. Na Slici 2.4 pogledajte ilustrovano vandijagonalno kretanje kroz rešetku.

Ovaj neispunjen uslov je prouzrokovao da se ne ispune druga dva osnovna uslova na kojima se zasniva kretanje kroz celobrojnu rešetku. Posmatrajući prvi slučaj, primećujemo da nije ispunjen uslov, da se svakim narednim korakom približavamo odredišnoj tački. Ako pogledamo drugi slučaj primetićemo da nije ispunjen uslov uvođenja restrikcije na mrežu veličine $n \times n$, tako da je kretanje van mreže nedefinisano, jer je tačno definisano koliko ima najkraćih puteva u celobrojnoj mreži, a to je broj C_n. Možemo izvesti zaključak da ključ koji nije bitbalansiran, tj. nije Katalanov objekat ne može biti funkcionalan u kretanju kroz celobrojnu mrežu.



Slika 2.3: Neuspešno šifrovanje sa ključem koji nije Katalanov objekat.



Slika 2.4: Primer neuspešnog šifrovanja sa ključem koji nije Katalanov objekat.

2.2 Metoda ballot notacije, stek permutacije i uparenih zagrada

Nakon predstavljanja metode kretanja kroz Dekartov koordinatni sistem u nastavku je predstavljeno još jedno svojstvo Katalanovih objekata kroz metodu *ballot* notacije, poznate kao problem glasanja. Ako se prezentuje to kretanje u Dekartovom koordinatnom sistemu onda

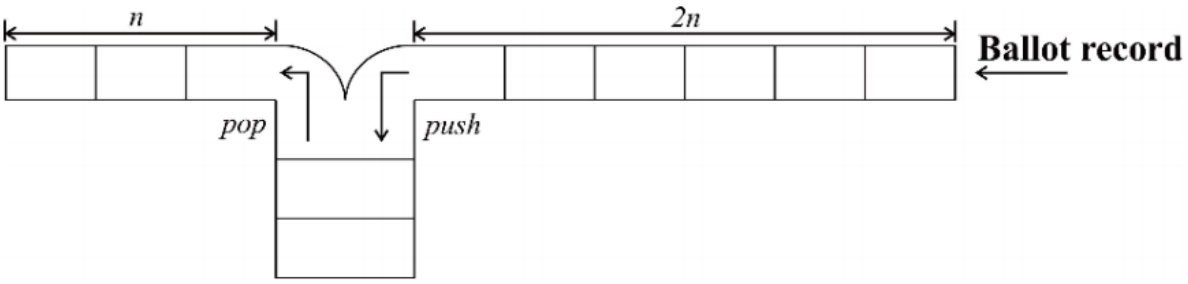
bilo koji pomeraj nagore odgovara glasu B, dok bilo koji pomeraj nadesno odgovara glasu A. Drugim rečima, svaka putanja koja se sastoji od $2n$ pomeraja u $n \times n$ mreži odgovara *ballot* zapisu dužine $2n$ (videti Sliku 2.1).

Primer 2.2.1. *Tabela 2.2 je formirana od niza glasanja redosleda* ***AABABB****. Ovaj ballot zapis odgovara kretanju kroz putanju mreže sa Slike 2.1. Celi brojevi redom označavaju do tada izbrojan broj pojava specifičnog glasa [53].*

Tabela 2.2: Redosled glasanja u *ballot* zapisu $AABABB$

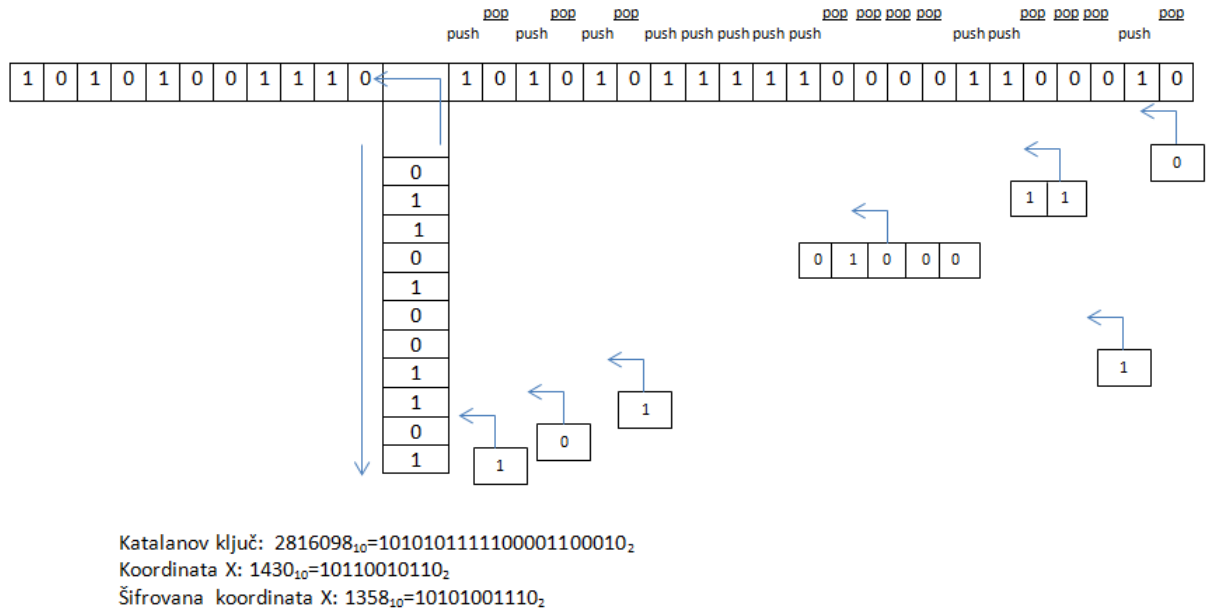
Kandidat	***A***	***A***	***B***	***A***	***B***	***B***
A	1	2	2	3	3	3
B	0	0	1	1	2	3

Stek predstavlja strukturu podataka koja je zasnovana na principu *LIFO(last in, frst out)* i na dvema osnovnim operacijama *push i pop*. Stek permutacije, kao metod za rešavanje kombinatornih problema, takođe mogu biti generisane na osnovu svojstva Katalanovih objekata. Ukoliko je trenutni bit u binarnom zapisu ključa "1", poziva se operacija *push* stavlja se broj pojavljivanja ovog bita u zapisu sa leve strane. Ukoliko se pojavljuje vrednost bita "0", poziva se operacija *pop* i izbacuje se bit iz steka. U radu [3] predstavljeno je da broj permutacija koje zadovoljavaju navedene uslove odgovara Katalanovom objektu. Na osnovu toga može se reći da se primenom steka može preslikati svaki binarni zapis (ili ekvivalentni ballot zapis) dužine *2n* na jednu od permutacija skupa $\{1, 2, ..., n\}$. Na ulazu steka se očekuje binarni zapis, tako da operacija *push* snima samo broj pojavljivanja bitova "1". Na taj način se dobija dva puta kraći izlaz. Na Slici 2.5. je predstavljen odnos ulaza i izlaza steka.



Slika 2.5: Proces obrade ulaznih podataka preko stek strukture.

U nastavku je predstavljen primer šifrovanja jedne od koordinata *(x,y,z)* 3D ravni primenom stek permutacija. Koordinata x ima vrednost *x=1430*. Njen binarni zapis je 1430_{10}=10110010110_2 koji ima *n=11* bitova. Vrednost Katalanovog objekta (u nastavku



Slika 2.6: Primer šifrovanja koordinata po pricipu stek permutacije

ključa) je *K=2816098*. Njegov binarni zapis je 2816098_{10}=$10101011111000011000010_2$ koji ima *2n=22* bita. Slika 2.6 opisuje detalje.

Kada se govori o procesu dešifrovanja, on je u suštini isti kao i proces šifrovanja samo što se enkriptovana koordinata i Katalan ključ (objekat) čitaju obrnutim redosledom. Ekvivalentna stek permutaciji je i metoda **Uparenih zagrada** *(eng. Balanced Parentheses)* [4]. Slika 2.7. predstavlja proces šifrovanja.

Ključ	1	0	1	0	1	0	1	1	1	1	1	0	0	0	0	1	1	0	0	0	1	0
Uparene zagrade	(	)	(	)	(	)	(	(	(	(	(	)	)	)	)	(	(	)	)	)	(	)
Koordinata X	1		0		1		1	0	0	1	0					1	1				0	
Šifrovana koordinata X		1		0		1						0	1	0	0			1	1	1		0

Slika 2.7: Primer šifrovanja koordinata po pricipu uparenih zagrada

Poglavlje 3

Računarska geometrija u *Javi* - Implementacija algoritama

U ovom poglavlju je predstavljena *Java* kao objektno orijentisani programski jezik, odnosno njene prednosti u procesu kreiranja algoritama računarske geometrije. Upotrebom postojećih klasa i programskih interfejsa (*Java 2D, 3D, Triangulation, Pnt, Triangle, Logic*) predstavljene su mogućnosti *Jave* u kreiranju inkrementalnog algoritma Delone triangulacije i modifikacji ovog algoritma shodno metodama koje će biti predstavljene.

3.1 Objektno - orijentisano programiranje i njegove prednosti

Objektno-orijentisano programiranje obiluje različitim mogućnostima poput jednostavnog korišćenja delova *izvornog koda*, fleksibilnost u kreiranju modularnih programa itd. *Java* kao jedan od najznačajnih objektno-orijentisanih jezika raspolaže mehanizmima poput klasa, objekata, polimorfizma i nasleđivanja.

U odnosu na druge programske jezike, *Java* podržava osnovne biblioteke klasa koje omogu- ćavaju rad sa osnovim tipovima podataka, Internet protokolima, kreiranje interfejsa za korisnike kao i rad sa ulazom i izlazom. Ako uzmemo u obzir statistiku sa portala *TIOBE Programming Community Index*[1] *Java* je i dalje jedan od najpopularnijih programskih jezika poslednjih godina. (Tabela 3.1).

Ako pogledamo statistiku navedenog portala primetićemo da Java zauzima drugo mesto, dok je u nekom ranijem periodu ubedljivo zauzimala prvo mesto. To nam govori da je ona sveobuhvatno gledano najviše zastupljen programski jezik.

Jednostavnost sintakse i prenosivost kreiranih programa u Javi su još neke njene prednosti. Ove sposobnosti Javi omogućuje njena instalirana virtuelna mašina (JVM - *Java Virtual Machine*). Drugim rečima, ovo znači da program jednom napisan u Javi, moguće je da se pokreće na različitim računarskim platformama pod uslovom da je JVM instalirana.

[1] *http : //www.tiobe.com/index.php/content/paperinfo/tpci/index.html*

Tabela 3.1: Statistika popularnosti programskih jezika (OOP jezici, May 2021-2020)

Objektno-orijentisani programski jezici	**Period posmatranja** **Maj-2021**	**2020**
C	**1**	**1**
Python	2	2
Java	3	3
C++	5	4
C#	5	4
(Visual) Basic	6	5
JavaScript	10	9
Assembly language	10	9
PHP	4	6
SQL	7	7
Ruby	11	8

Tabela 3.2: Statistika popularnosti programskih jezika (svi prog. jezici, 1996-2021)

Programski	**Rangiranje**				
jezici	**2021**	**2016**	**2011**	**2006**	**2001**
C	1	2	2	2	1
Java	**2**	**1**	**1**	**1**	**3**
Python	3	5	6	8	27
C++	4	3	3	3	2
C#	5	4	5	7	13
Visual Basic	6	13	-	-	-
JavaScript	7	8	10	9	10
PHP	8	6	4	4	11
SQL	9	-	-	-	-

Na primer, pokazivači nisu eksplicitno navedeni i nizovi su realni objekti. Pored toga, upravljanje memorijom je automatski, odnosno nakon kreiranja novog objekta *Java* automatski dodeljuje potrebnu količinu memorije za njegovo skladištenje. Nakon prestanka važnosti objekta, oslobađanje memorije se vrši pomoću mehanizma "garbage collector". Ovaj mehanizan recimo ne postoji u jeziku C++, tako da je za razliku od njega upravljanje memorijom u Javi znatno olakšano. Višenitno programiranje, kao i rad u mrežnom okruženju su takođe neke od prednosti Jave u odnosu na neke druge programske jezike. [14, 15]

3.2 Primena programskog jezika *Java* za računarsku geometriju i GUI

U okviru algoritama računarske geometrije, pored ostalih, spadaju i neki prosti algoritmi za ispitivanje kolinearnosti tačaka, pozicije među njima i složeni kao što su generisanje Voronoi dijagrama ili Delone triangulacije poligona i pronalazak najmanjeg konveksnog omotača skupa tačaka. Nezaobilazna stvar je efikasnost tih algoritama. Da bi efikasnost bila adekvatna, neophodno je odabrati pogodne strukture podataka za prezentaciju segmenata, tačaka, skupa tačaka kao i ostalih geometrijskih objekata. U radovima [16, 17, 18] su predstavljeni neki od algoritama računarske geometrije.

3.2.1 Korišćeni paketi (*AWT*, *Swing*, *JavaFX*) u kreiranju aplikacija

Za predložene metode u disertaciji, razvijene su konkretne Java aplikacije: Enkripcija 3D ravni (TIN model), zatim za autentifikaciju utemeljenu na šifrovanju slike pomoću Delone triangulacije i Katalanovih objekata i primena Delone triangulacije i Katalanovih objekata u steganografiji. U kreiranim aplikacijama korišćeni su paketi iz *AWT* i *Swing* biblioteke kao i JavaFX softverska platforma, odnosno njihovi osnovni elementi. U svojoj početnoj verziji *Java* programski jezik je posedovao biblioteku komponenata zvanu *Abstract Window Toolkit (AWT)* koja je bila potrebna za izgradnju grafičkog korisničkog interfejsa (GUI). Ova biblioteka se zasniva na korišćenju komponenata korisničkog interfejsa na platformi programa koji se pokreće. Drugim rečima za svaki operativni sistem, implementacija *AWT* komponenata je različita.

Za razliku od *AWT*, *Swing* je lagano Java grafičko korisničko okruženje (GUI) koje se koristi za kreiranje različitih aplikacija. Swing ima komponente koje su nezavisne od platformi. Omogućuje korisniku kreiranje dugmadi i linije traka (Scrollbar) za pomeranje. *Swing* uključuje pakete za kreiranje desktop aplikacija na Javi. Ove komponente su napisane na jeziku Java. Dio je Java Foundation Classes (JFC) [70].

Unutar ove *Swing* biblioteke korišćen je paket `event` pomoću koga su uzimane vrednosti koordinata (x,y) tačaka odnosno, temena trouglova prilikom kreiranja Delone triangulacije na panelu. Ako posmatramo direktnu vezu između *AWT* i *Swing* paketa, primetićemo da je većina klasa *Swing* paketa samo unapređena verzija klasa unutar *AWT* paketa. U pogledu fleksibilnosti *Swing* klase su fleksibilnije od sličnih klasa u *AWT* jer su u potpunosti implmentirane u *Javi* [19].

Pored navedenih paketa *AWT*-a i *Swing* za triangulaciju slike korišćena je JavaFX softverska platforma za kreiranje i isporuku aplikacija za radnu površinu, kao i bogatih internetskih aplikacija (RIA). *JavaFX* je zamenjen Swingom kao standardnom GUI bibliotekom za Java SE. JavaFX ima podršku za desktop aplikacije kao i za web u sistemu Microsoft Windows, Linux i Mac OS X. [63]. Drugim rečima, JavaFX predstavlja skup grafičkih

paketa, koji služe programerima za kreiranje, dizajniranje, uklanjanje grešaka kod bogatih klijentskih aplikacija, uz mogućnost da rade na različitim platformama [64].

U okviru disertacije razvijene su tri autorske aplikacije u *Javi* za tri predložene metode. Korišćeno je *Java NetBeans IDE* (*Integrated Development Environment*) okruženje. Pored ostalih pogodnosti ovo okruženje je pogodno i za implementaciju alogoritama Delone triangulacije poligona, enkripciju koordinata temena trouglova kao i steganografiju (skrivanje podataka u nekom medijumu) u formi i *Java apleta*. Takođe, u okviru ove platforme nalaze se komponente za kreiranje GUI-a. Ova platforma nudi uključivanje korisničkih modula preko mehanizama u cilju funkcionalnosti celokupne platforme [20].

3.2.2 *OpenGL* biblioteka u *Javi*

Ova biblioteka je kreirana tako da može da pristupi *OpenGL API*-ju, odnosno dizajnirana je tako da se pomoću nje mogu kreirati 2D i 3D grafičke aplikcije programirane u Javi. Ona je otvorenog koda i u početku su je razvili bivši studenti MIT-a Ken Russell i Chris Kline. Kasnije je usvojila grupacija Sun Microsystems i sada je integrisana u Java i služi u oblastima grafike, zvuka i procesiranja (JOGAMP). Funkcioniše na različitim operativnim sistemima kao što su Windows, Solaris, Mac OS X i Linux (na x 86) [67].

Drugim rečima, ovo je biblioteka koja je namenjena za rad sa geometrijskim oblicima i unutar nje se nalaze svi paketi i klase za realizaciju programerskih zadataka takvog tipa. Pored navedenih integracija ona podržava i rad sa *Swing* komponentama [21].

Posmatrajući Java 2D API sistem, primetićemo da je on skup klasa za naprednu 2D grafiku i slike. Obuhvata linijsko crtanje, tekst i slike u jednom opsežnom modelu. Takođe API pruža bogat skup operatora za obradu slika orijentisanih na prikaz. Ove klase se nalaze i `java.awt` i `java.awt.image` [22, 68]. Za razliku od Java 2D API sistem-a, Java 3D API omogućuje kreiranje trodimenzionalnih grafičkih aplikacija i 3D apleta baziranih na Web okruženju tj. Internetu. Pruža konstrukcije na visokom nivou za kreiranje i manipulaciju 3D geometrijom i izgradnju struktura korišćenih u prikazivanju te iste geometrije [69].

Implementacija *Java 3D API*-a sastoji od tri paketa: *OpenGL*, *DirectX* i *JOpenGL* [23, 24]. Sveukupno posmatrano, *Java OpenGL* u stvari fokusirana je na 3D prikaz, tako da sa pomoćnim bibliotekama olakšava kreiranje geometrijskih oblika.

3.2.3 Klase i paketi za kreiranje inkrementalnog algoritma Delone triangulacije poligona

U ovoj sekciji predstavljen je postupak kreiranja Delone triangulacije. Samim tim predstavljene su mogućnosti *Jave* u procesu rada sa računarskom geometrijom, tačnije u procesu kreiranja Delone triangulacije. Kao što je već pomenuto, standardni paketi koji se koriste u računarskoj geometriji su *java.awt*, *javax.swing* i *javafx.geometry*. U sledećoj tabeli su predstavaljene klase i metode iz pomenutih paketa koje su korišćene za kreiranje inkrementalnog

algoritma Delone triangulacije.

Tabela 3.3: Korišćene klase i medode za realizaciju *Inkrementalnog* algoritma

KLASE	**METODE**

DelaunayAp	`main(), init(), run()`
	`actionPerformed(), mouseEntered()`
	`mousePressed(), mouseClicked()`

Triangulation	`DelaunayPlace (), update ()` `locate (), contains ()`

Triangle	`isNeighbor (), facetOpposite ()`
	`toString(), getCircumcenter ()`
	`add (), iterator ()`

Pnt	`coord (), dimension ()`
	`dimCheck (), extend()`
	`add (), angle ()`

DelaunayPanel	`addSite(), clear()`
	`getColor(), paintComponent()`
	`drawAllDelaunay, drawAllVoronoi()`

Logic	`createRandomPoints(), findFirstPoint()`
	`findSecondPoint(), findThirdPoint()`
	`createTriangulation(), flipAdjacentTriangles()`
	`colorTriangulation()`

Glavna klasa je `DelaunayAp` i ona nasleđuje klasu *javax.swing.JApplet*. U okviru ove klase postoji izvšna metoda `main()` koja se prva pokreće. Takođe, tu su i `init()` metoda za inicijalizaciju `DelaunayAp` objekta, kao i `run()` metoda za dodavanje svih grafičkih elemenata kao i osluškivača događaja. Nakon pokretanja applikacije, odnosno apleta, kreće se sa kreiranjem Delone triangulacije poligona. Kao što je već pomenuto za njenu konstrukciju koristi se inkrementalni algoritam koji je objašnjen u prvoj sekciji (Uvodna razmatranja). Samim pokretanjem aplikacije u metodi `run()` se poziva objekat `DelaunayPanel` koji u svom konstruktoru kreira inicijalni trougao sa koordinatama p_{-1}:=($3M$, 0), p_{-2}:=(0,$3M$) i p_{-3}:=(-$3M$,-$3M$). Deo koda koji inicijalizuje početni trougao dat je u nastavku.

```
public DelaunayPanel (DelaunayAp controller) {
      this.controller = controller;
      initialTriangle = new Triangle(
              new Pnt(-initialSize, -initialSize),
              new Pnt( initialSize, -initialSize),
              new Pnt(           0,  initialSize));
      dt = new Triangulation(initialTriangle);
      colorTable = new HashMap<Object, Color>();
  }
```

Kao što se može primetiti, unutar ove metode kreira se objekat `initialTriangle` klase `Triangle` koji je argument objekta `dt` klase `Triangulation`. Nakon kreiranja inicijalnog trougla, kreće se sa slučajnim odabirom tačaka odnosno temena. Ova akcija se realizuje u metodi `mousePressed(MouseEvent e)`, gde se kreira objekat `point` klase `Pnt`. Ovaj objekat uzima koordinate x,y pomoću metoda `e.getX(), e.getY()`. Nakon toga se tačka odnosno teme postavlja na panel sa metodom `addSite(point)` i poziva metoda `repaint()`. Segment koda metode `mousePressed(MouseEvent e)` dat je u nastavku.

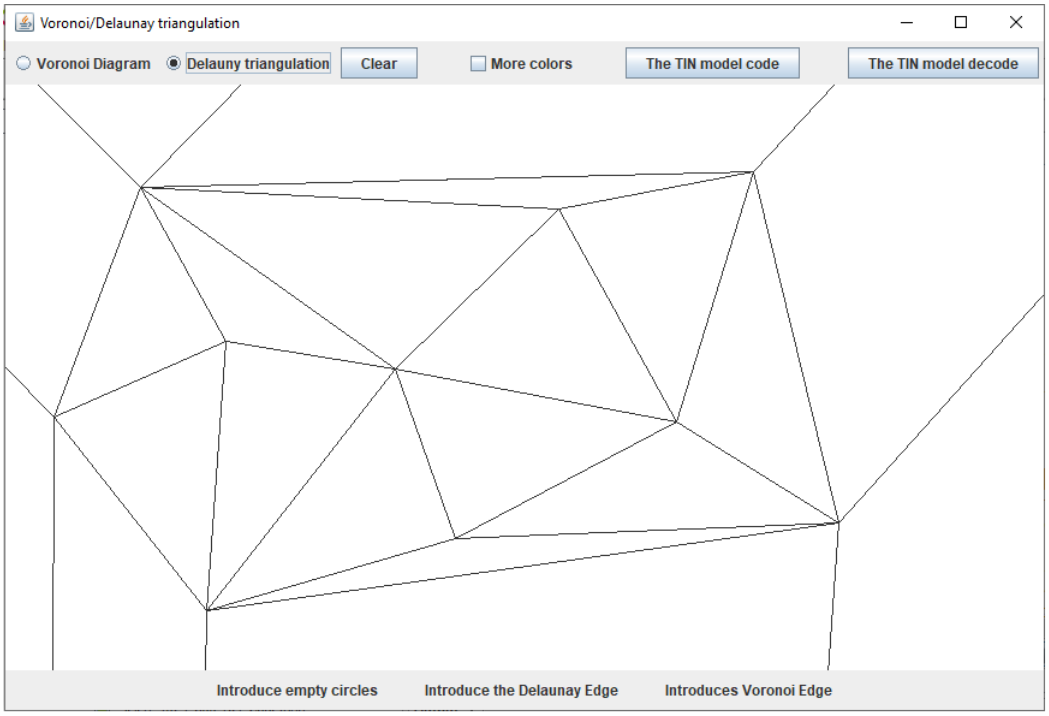
```
public void mousePressed(MouseEvent e) {
        if (e.getSource() != DelaunayPanel) return;
        Pnt point = new Pnt(e.getX(), e.getY());
        if (debug ) System.out.println("Click " + point);
        DelaunayPanel.addSite(point);
        DelaunayPanel.repaint();
    }
```

Važno je istaći da se rad metode `repaint()` u stvari zasniva na crtanju trouglova unutar panela, i to na taj način što se za prvu unetu tačku iscrtaju linije ka temenima inicijalnog trougla, zatim svaka sledeća tačka koja se pojavi, od nje se iscrtavaju linije ka najbližim susednim temenima. Proveravaju se ivice, da li su legalne kako je objašnjeno u algoritmu *LegalizujIvicu* i na taj način se iscrtava legalna Delone triangulacija. Metoda `repaint()` autoamatski poziva metodu `paintComponent (Graphics g)`, koja zapravo iscrtava Delone triangulaciju. Segment koda koji obašnjava rad ove metode dat je u nastavku.

```
public void paintComponent (Graphics g) {
        super.paintComponent(g);
        this.g = g;
        Color temp = g.getColor();
        if (!controller.isVoronoi()) g.setColor(DelaunayColor);
        else if (dt.contains(initialTriangle))
              g.setColor(this.getBackground());
        else g.setColor(voronoiColor);
        g.fillRect(0, 0, this.getWidth(), this.getHeight());
        g.setColor(temp);
        if (!controller.isColorful()) colorTable.clear();
        if (controller.isVoronoi())
             drawAllVoronoi(controller.isColorful(), true);
        else drawAllDelaunay(controller.isColorful());
        temp = g.getColor();
        g.setColor(Color.white);
        if (controller.showingCircles()) drawAllCircles();
        if (controller.showingDelaunay()) drawAllDelaunay(false);
        if (controller.showingVoronoi())
```

```
            drawAllVoronoi(false, false);
        g.setColor(temp);
    }
```

Pored opisanog načina rada, odnosno kreiranja Delone aplikacije kod JavaFX aplikacije za kreiranje triangulacije slike, glavna klasa je `Logic` koja u sebi ima sedam metoda. Kreiranje triangulacije je takođe zamišljeno da se odvija u sedam koraka. U prilozima će biti predstavljeni glavni segmenti koda pomenute klase. Rezultat rada gore pomenutih klasa i metoda dat je na Slici 3.1.



Slika 3.1: Primer Delone triangulacije $\mathcal{D}$

Poglavlje 4

Metode zaštite podataka pomoću Voronoi-Delone triangulacije i Katalanovih objekata

Kao što je već predstavljeno u uvodnom delu, u ovoj disertaciji razvijeno je nekoliko metoda zaštite podataka pomoću Voronoi-Delone triangulacije i Katalanovih objekata. U nastavku rada, dataljnije je analiziran rad ovih metoda. Prva opisana metoda je enkripcija 3D ravni u GIS-u.

4.1 Metoda enkripcije 3D ravni u GIS-u pomoću Voronoi-Delone triangulacija i Katalanovih objekata

U modernom informatičkom dobu podaci predstavljaju kodovane činjenice koje počinju da egzistiraju onda kada bivaju registrovane, odnosno zabeležene određenim simbolima. Drugim rečima, podaci se sastoje od brojeva, reči, slika, teksta ili simbola. Podatke koji predstavljaju bilo koji deo Zemljine površine ili neke njene osobine nazivamo *geografskim podacima*. Bitno je istaći da geografski podaci nisu samo namenjeni za geografe, naučnike ili poslovne krugove već se oni mogu naći svuda, od baza podataka do poslovnih aplikacija. Kada su u pitanju poslovne aplikacije, centralno mesto zauzimaju odgovori na pitanja: Gde se to nalazi? Šta se tu nalazi, odnosno kakvih je svojstava?

Sa aspekta praktične upotrebe vrednost geografskih informacija je u posedovanju podataka o: lokacijama, adresama, rastojanjima, granicama i mogućnostima upravljanja prirodnim i društvenim pojavama. Najbitniji podaci u svakoj GIS proceduri i aplikaciji su prostorni podaci, koji su georeferencirani njihovom lokacijom na površini zemlje. Geo-Referenciranost podrazumeva tačno zabeleženu lokaciju u određenom koordinantnom sistemu. Obzirom da je GPS sistem okosnica u lociranju i praćenju ciljeva na površini zemlje, bezbednost, odnosno zaštita GPS signala, koji se šalju zemaljskim stanicama, je od velike

važnosti u procesu kreiranja poslovnih aplikacija za navigaciju. U tom pogledu, enkripcija 3D ravni primenom Katalanovih objekata i Delone triangulacije, je samo jedan od modela zaštite geografskih podataka.

4.1.1 Daljinska detekcija - GPS sistem za globalno pozicioniranje

Metoda kojom se vrši prikupljanje i interpretacija informacija o udaljenim objektima bez fizičkog dodira sa objektom naziva se *daljinska detekcija*. Uobičajene platforme za opažanja u daljinskim istraživanjima su: avioni, svemirske sonde i sateliti. Ova metoda najčešće se fokusira na dve uže oblasti: teledetekcija i fotogrametrija. Daljinsko istraživanje u kome se prikupljaju informacije o zemljinoj površini, uz pomoć uređaja smeštenim u satelitima, i interpretaciju tako dobijenih informacija nazivamo *Teledetekcijom*[11]. Druga oblast je *Fotogrametrija* tj. tehnika merenja kojom se oblik, veličina i položaj snimljenog predmeta izvodi na osnovu fotografskih snimaka. Kao što je rečeno, sateliti predstavljaju najzastupljeniju platformu u teledetekciji.

Slika 4.1: Globalni sistem za pozicioniranje - GPS

Globalni sistem za pozicioniranje (eng. *Global Positioning System - GPS*) je satelitski navigacioni sistem, jedinstven u svetu pomoću koga u svakom trenutku možemo odrediti tačan položaj tačke (prijemnika) na zemlji. U osnovi GPS sateliti šalju signale prijemnicima

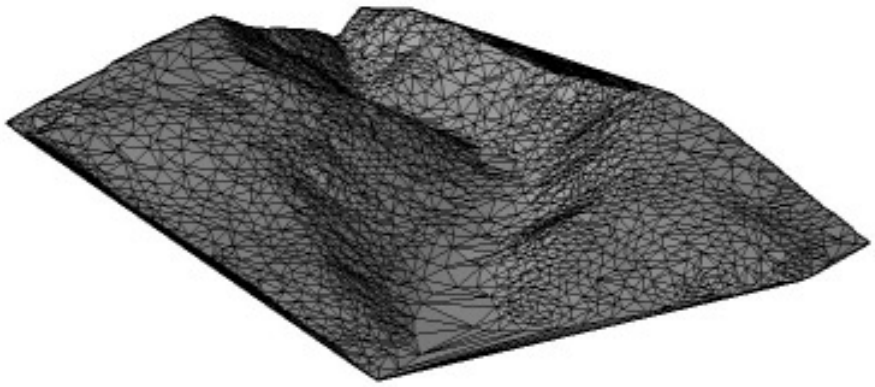
o njihovoj geografskoj širini, dužini i visini, tj. šalju signale za tri koordinate (x, y, z). Postupak za dobijanje navedenih koordinata zasniva se na principu preseka (*trilateracije*) sfera koje emituju tri satelita. Primena GPS-a je mnogostruka [8].

Prvo je razvijen za vojne potrebe, da bi kasnije 80-ih, počeo da se koristi i u civilne svrhe. Navigacija aviona, brodova, automobila je bez GPS-a nezamisliva. U procesu zaštite signala tj. koordinata tačaka (pozicije prijemnika) potrebno je da u satelitu postoji mehanizam (algoritam) enkripcije. U narednom poglavlju detaljnije je predstavljen ovaj algoritam. Dakle, prijemniku se šalje kriptovan signal i ključ enkripcije. Sa druge strane, prijemnik treba da poseduje mehanizam (algoritam) dekripcije koji je u stanju da tako dobijeni signal (kriptovan sa ključem) vrati u prvobitnu vrednost.

4.1.2 Modeliranje 3D ravni - TIN model

Kada govorimo o TIN modelu, važno je istaći da je on u stvari pravilna mreža nepravilnih trouglova *(Triangulated Irregular Network - TIN)*. Ovaj model formira se na osnovu poznatih pozicija tačaka i njihovih visina tj. koordinata tačaka (x, y, z). Za proces formiranja mreže nepravilih trouglova koristi se inkrementalni algoritam Delone triangulacije.

Na osnovu TIN modela mogu se izvoditi svi željeni proračuni: vrednost nagiba u zadatoj tački, visina za datu poziciju u horizontalnom smislu, pravac maksimalnog nagiba, krivina (zakrivljenost) površi u zadatoj tački, vizuelizacija modela terena, geostatistička analiza i drugo. Upotreba TIN modela je mnogostruka. Koristi se u projektovanju saobraćajnica, hidrogradnje, podzemnih objekata, u vojno geografskoj analizi, itd [9]. Na Slici 4.2. je predstavljen (TIN) model 3D ravni.



Slika 4.2: TIN mreža nepravilnih trougolova

Obzirom na široku primenu TIN modela, potrebno je omogućiti enkripciju koordinata tačaka u trenucima prenosa elektronskim putem (naročito je bitno ukoliko je TIN model

poverljiva informacija), kao i prilikom njegovog skladištenja na nekom prenosnom medijumu. Uopšteno, algoritam predstavljen u narednom poglavlju za rezultat daje takođe TIN model ali sa drugim (šifrovanim) vrednostima koordinata.

4.1.3 Implementacija algoritma enkripcije 3D ravni u Java - Net Beans okruženju

Programski jezik Java predstavlja jezik opšte korisničke i poslovne namene, relativno je jednostavan, ipak dosta moćan ali i platformski nezavistan. Platformska nezavisnost se ogleda u tome da se programi pisani u Javi prevode u tzv. bajt-kod, koji nije mašinski jezik nijednog konkretnog računara, već se izvršava na JVM. Java Virtuelna Mašina predstavlja virtuelni računar koji može biti simuliran na bilo kom računaru. Java takođe ima mnogobrojne funkcionalnosti koje podržavaju kriptografske algoritme. Pomenute kriptografske funkcionalnosti Java obezbeđuje kroz dva API-ja i to. *Java Cryptography Architecture (JCA)* i *Java Cryptography Extension (JCE).* Klase oba pomenuta API-ja obezbeđuju rad za hešovanje poruka, generisanje ključeva kao i rad sa digitalnim potpisima.

4.1.4 Proces šifrovanja i dešifrovanja koordinata

Proces šifrovanja koordinta započinje prvo generisanjem niza od ukupnog broja slučajno odabranih temena trouglova TIN-modela. Nakon toga se primeni inkrementalni algoritam Delone triangulacije. U drugom koraku se svaka decimalna vrednost koordinata *(x,y,z)* temena formirane triangulacije pamti u celobrojni niz. Potom se vrši njihova konverzija iz decimalnog u binarni zapis jer je to potrebno uraditi iz razloga što je Katalanov objekat zadat u binarnom odbliku. Tako dobijeni binarni zapis koordinata se metodom *stek permutacije* konvertuje u neki drugi šifrovani zapis, koji nakon ponovne konverzije u decimalni zapis u stvari predstavlja enkriptovanu koodinatu. U nastavku je detaljnije opisan princip rada algoritma. 4.1.1.

U algoritmu 4.1.1 je zbog jasnoće predstavljeno šifrovanje tačaka ravni. Međutim to šifrovanje tačaka podrazumeva šifrovanje njihovih koordinata (x, y, z).

Za realizaciju navedenih koraka u algoritmu, pored metoda za Delone triangulaciju, korišćene su metode: $Konvertuj\_U\_Binarni\_Zapis$, $Binarno\_Sifrovanje\_Kordinata$, $Konvertuj\_Bibary\_u\_Integer$ klase *DelaunayAp.ja-va.* Način šifrovanja treće koordinate z je isti kao i *x,y.* U nastavku je detaljnije predstavljen rad ovog algoritma.

4.1.5 Struktura Java izvornog koda

Pokretanje Java GUI aplikacije, pokreće se izvršna metoda *main()* klase *DelaunayAp.java.* Odmah na početku je potrebno da se unese broj n tačaka u ravni u skupu P. Nakon toga se klikom na panel aplikacije slučajnim izborom dodeljuju koordinate (x, y), u velikom početnom trouglu $p_i p_j p_k$, i na taj način se određuje položaj tačke p_r u ravni. Proces

Algoritam 4.1.1 Enkripcija TIN modela u ravni

Ulaz: Skup P od n tačaka u ravni. Neka su p_{-1}, p_{-2} i p_{-3} tri tačke inicijalnog truugla unutar koga su sadržane sve ostale tačke skupa P .

1: Inicijalna triangulucija $\mathcal{T}$ koja sadrži tačke p_{-1}, p_{-2} i p_{-3}
2: **for** $r = 1$ to n **do** (Unesi p_r u $\mathcal{T}$)
3: Pronađi trougao $p_i p_j p_k \in \mathcal{T}$, koji sadrži p_r.
4: Unesi p_r u celobrojni niz K.
5: **vrati** $\mathcal{T}$
6: **for** $r = 1$ to n **do** (Pristupi p_r u nizu K)
7: Konvertuj p_r u binardni zapis
8: Enkriptuj p_r **Stek permutacije** metodom baziranom na **Katalanovom ključu** iz C_n
9: Konvertuj p_s u celobrojni niz (Nakon permutacije bita p_r, ovo teme postaje p_s)
10: Unesi p_s u niz K_s
11: **for** $s = 1$ to n **do** (Unesi p_s u $\mathcal{T}_s$)
12: Pronađi trougao $p_i p_j p_k \in \mathcal{T}_s$, koji sadrži p_s.
13: **vrati** $\mathcal{T}_s$.
14: $Izlaz$: Enkriptovana n temena (tačke) iz skupa P (Enkriptovan TIN model u ravni)

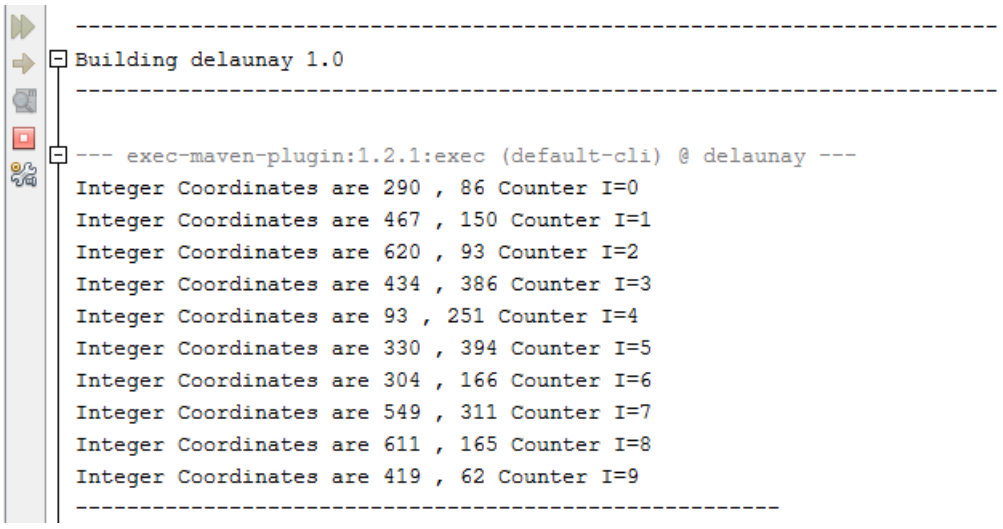
se ponavlja sve dok se ne unese i poslednja tačka. Ovo se postiže pozivanjem događaja $mouseClicked(MouseEvente)$. Napomene radi, ova motoda je modifikovana na osnovu one iste koja je predstavljena u Sekciji 3.2.3 na taj način što se pozicije `e.getX()`, `e.getY()` dodeljuju celobrojnim nizovima $S\_x\_kordinata[i]$, $S\_y\_kordinata[i]$. Kasnije se koordinate ovih nizova pretvaraju u binarni zapis radi enkripcije. U nastavku je predstavljen *Java* izvorni kod navedene metode.

```
public void mouseClicked(MouseEvent e) {

x_cor = e.getX();   // Dodeljivanje vrednosti Koordinata
                                promenljivoj x_cor
y_cor = e.getY();   // Dodeljivanje vrednosti Koordinata
                                promenljivoj y_cor
  if ((pom_x!=x_cor) && (pom_y!=y_cor))
   {
    S_x_kordinata[i]=x_cor;   //Dodela Nizu K vrednost
                                        koordinate X
    S_y_kordinata[i]=y_cor;   //Dodela Nizu K vrednost
                                        koordinate Y
      pom_x=x_cor;
       pom_y=y_cor;
      System.out.println("Integer Koordinate su " +
       S_x_kordinata[i]+ " , "    +S_y_kordinata[i] +"
        brojac I="+i);
```

```
            if (i==broj_temena-1)
             {
            JOptionPane.showMessageDialog(null,"
             Uneli ste broj "+  broj_temena+" pre
             dvidjenih tacaka ","Obavestenje",
JOptionPane.PLAIN_MESSAGE);
             }
             i++;
          }
}
```

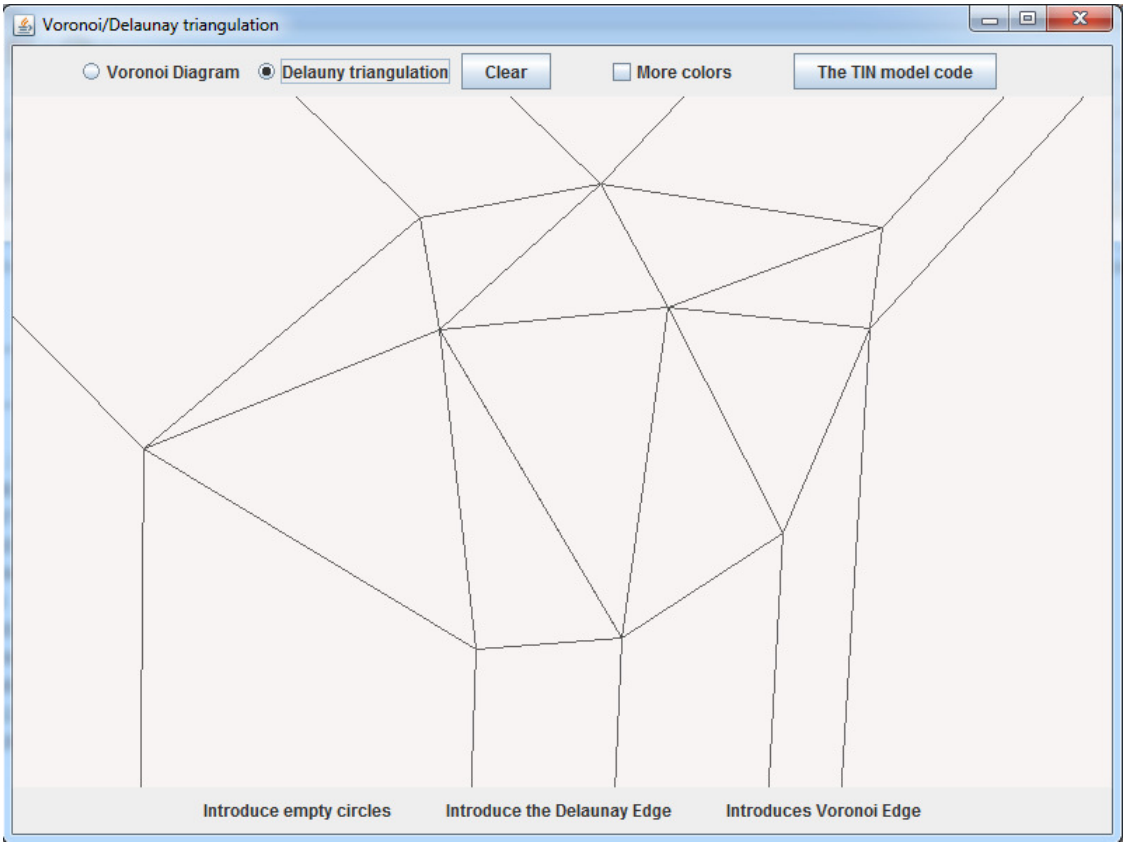
U pozadini se pozivaju metode za rad inkrementalnog algoritma Delone triangulacije tako da, određivanjem položaja koordinata *(x,y)* svih tačaka iz nekog skupa u ravni, kreira se TIN mreža trouglova koju treba šifrovati. Na Slici 4.3. su predstavljene decimalne vrednosti koordinata (x, y), dok je na Slici 4.4. predstavljen TIN model ravni sa unesenim tačkama (teminima) trouglova zadatim u decimalnom zapisu.



Slika 4.3: Vrednosti unetih koordinata (x, y) temena trogulova

Prethodno opisani događaji odgovaraju algoritmu do koraka 6 gde je dobijen niz K sa unetim koridinatama (x, y) tačaka. Klikom na "TIN model Code" poziva se metoda $Sifruj\_X\_Y\_Koordinate()$. Prva koja se poziva je metoda $Konvertuj\ \_U\_Binarni\_Zapis$ (), gde se svakoj koordinati tačke u nizu K pristupa i vrši njena konverzija iz decimalnog u binarni zapis (korak 8 u algoritmu 4.1.1).

Nakon konverzije pokreće se metoda $Binarno\_Sifrovanje\_Kordinata()$. Rad ove metode je detaljnije objašnjen u Sekciji 2.1.1. Rezultat rada ove metode jesu koraci od 9 do 14 u algoritmu. Treba naglasiti da je rezolucija monitora u ovakvom okruženju ograničavajući faktor. Broj bitova koordiante je exponent broja 2 i mora uvek da je unutar opsega u odnosu



Slika 4.4: TIN mreža nepravilnih trougolova

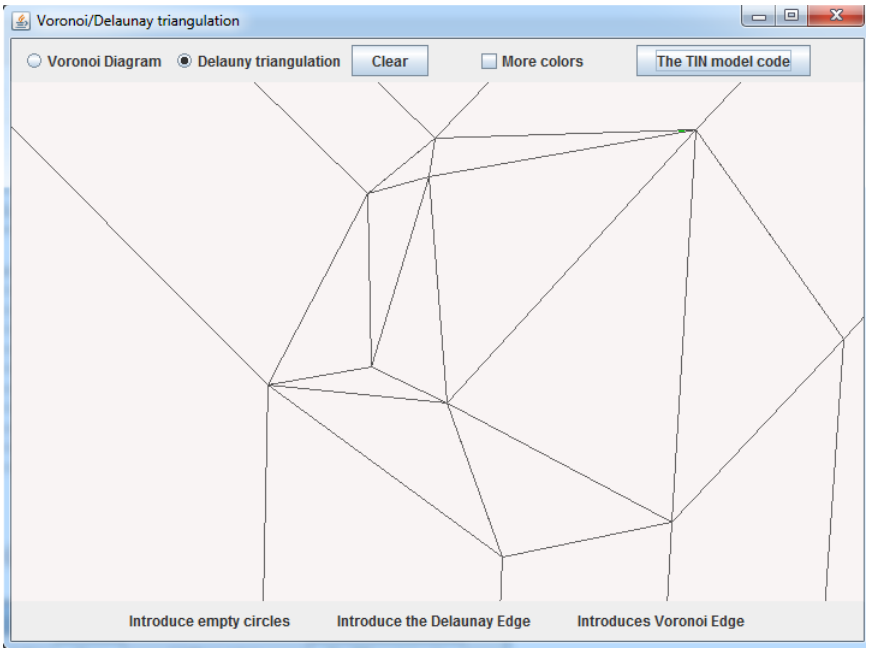
na rezoluciju monitora. Na primer ako je rezolicuja 1440 x 900 piksela, broj 1440 prelazi vrednost od 1024 (2^{10}=1024) eksponent mora biti 11 da bi se vrednosti koordinata koje su veće od 1024 mogle predstaviti.

Takođe, broj bitova Katalanovog ključa je uvek $2n$ gde n predstavlja broj bitova koordinate. U ovom slučaju je to 22 bita. Kada se radi o prostoru i vrednostima koordinata koje npr. šalju GPS sateliti zemaljskim stanicama, ovaj uslov ne važi. U tom slučaju se nakon konverzije decimalne vrednosti koordinate u binarnu vrednost, bira Katalanov ključ po modelu $2n$. Rezultat rada ove metode dat je na Slici 4.5. Dok je na Slici 4.6. predstavljen šifrovan TIN model ravni.

Ako se uzme u obzir proces dešifrovanja, može se konstatovati da se on odvija na način da originalna i enkriptovana koordinata menjaju mesto. Katalanov objekat ostaje isti stim što se čitanje bita ključa i šifrata vrši sa desna na levo tj. obrnutim redosledom od šifrovanja. Na Slici 4.7. je predstavljen rezultat rada metode $Binarno\_Desifrovanje\_Kordinata()$, a dobijeni dešifrovani TIN model odgovara originalu kao na Slici 4.4.

```
Integer Cooordinate: X = 611, Y =165
Catalan Key for N=11: 1010101111100001100010
Binary Coordinate: X = 01001100011 ,Y = 00010100101
Encrypted Binary Coordinates: X = 01000111001 Y = 00000100111
Encrypted Binary Coordinates: X = 569 Y = 39
------------------------------------------------------------------------
Integer Cooordinate: X = 419, Y =62
Catalan Key for N=11: 1010101111100001100010
Binary Coordinate: X = 00110100011 ,Y = 00000111110
Encrypted Binary Coordinates: X = 00100101011 Y = 00011101100
Encrypted Binary Coordinates: X = 299 Y = 236
------------------------------------------------------------------------
End of Encryption Coordinates-----------------------------------------
 Encrypted Coordinates X = 296 Y= 92
 Encrypted Coordinates X = 347 Y= 78
 Encrypted Coordinates X = 692 Y= 213
 Encrypted Coordinates X = 362 Y= 266
 Encrypted Coordinates X = 213 Y= 251
 Encrypted Coordinates X = 408 Y= 394
 Encrypted Coordinates X = 352 Y= 46
 Encrypted Coordinates X = 549 Y= 365
 Encrypted Coordinates X = 569 Y= 39
 Encrypted Coordinates X = 299 Y= 236
Execution time: 20 millisecond
```

Slika 4.5: Binarne vrednosti koordinata i nihovi šifrati

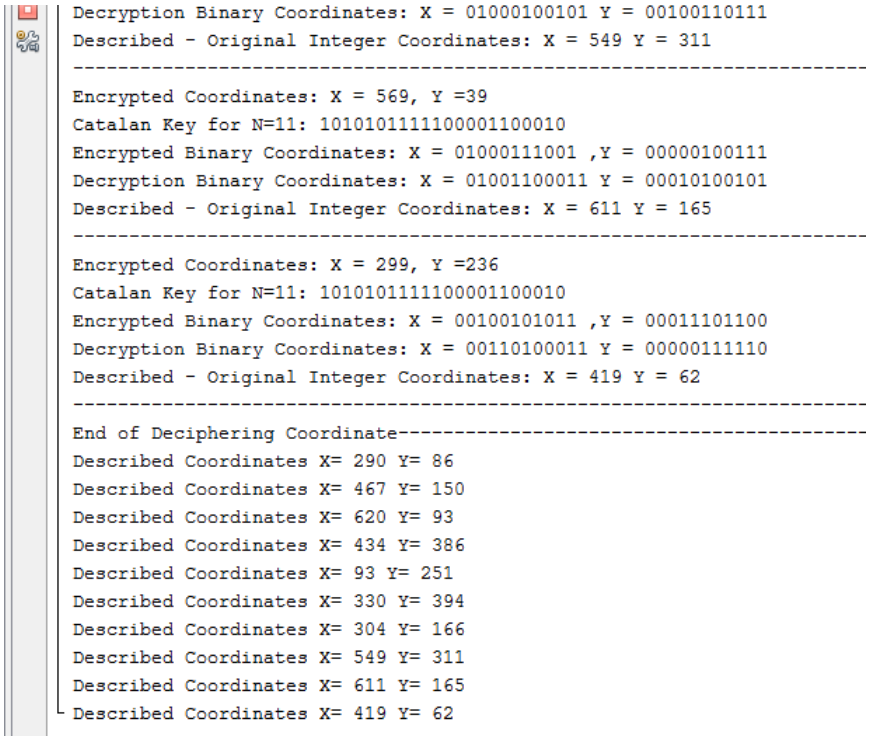


Slika 4.6: Šifrovan TIN model

4.1.6 Eksperimentalni rezultati

Vreme enkripcije je testirano sa brojevima temena iz skupa $N = \{5,10,20,40,100,200,400\}$. Budući da se u *JavaNetBeans* okruženju aplikacija istovremeno kompiluje i interpretira, ovo okruženje je odabrano kao pogodno da se ispitaju mogućnosti našeg algoritma.

Ako se ovi podaci predstave i grafički, može se primetiti da vreme šifrovanja nije direk-



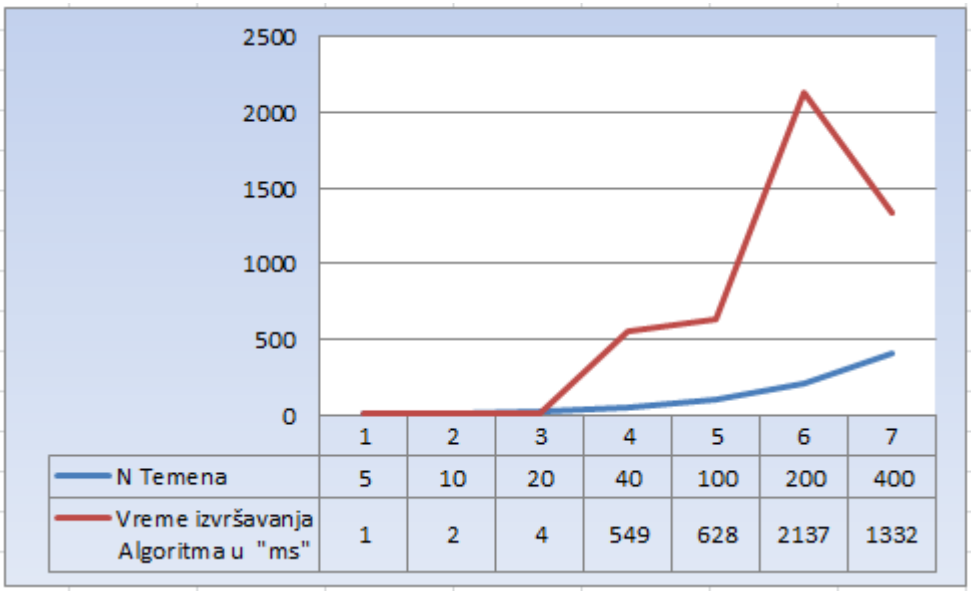
Slika 4.7: Vredsnoti dešifrovanih koordinata

Tabela 4.1: Vreme izvršavanja algoritma u ms

N Temena	**Vreme enkripcije temena**
5	1
10	2
20	4
40	549
100	628
200	2137
400	1332

tno proporcionalno broju temena trouglova. Ova činjenica je dobar pokazatelj, jer vreme šifrovanja ne raste shodno povećanju broja temena.

Uzimajući u obzir ovako malo vreme za šifrovanje, šifrovane koordinate mogu se pohraniti u neku bazu podataka, što će dodatno povećati učinkovitost ovog algoritma. Testiranje je izvršeno na računaru sledećih performansi: *Intel Core i5-CPU 2.6 GHz, RAM -4 Giga-Bytes, Operating system: Windows 7 Microsoft -64 bits.* Predložena metoda je kombinacija računarske geometrije, geografskih informacionih sistema i kriptografije. Kao što je već pomenuto ovo je nova metoda za kodovanje koordinata pomoću Katalanovog objekta. Objavljeni naučni rad autora na kome se bazira data metoda je [47].



Slika 4.8: Vreme enkripcije koordinata (x, y)

4.2 Metoda provere autentičnosti na temelju enkripcije slike pomoću Delone triangulacije i Katalanovih objekata

Ubrzanom ekspanzijom informacionih tehnologija bezbednost u protoku informacija putem interneta je sve više ugrožena. Ugroženost u protoku informacija je posebno izražena kod bankarskih transakcija. U nastavku je predstavljen metod autentifikacije korisnika (klijenata) banaka enkripcijom slike pomoću inkrementalnog algoritma Voronoi - Delone triangulacije i Katalanovih objekata. Predstavljeni metod se sastoji iz pet faza. U današnjoj eri modernih tehnologija javlja se sve veća potreba za kreiranjem sigurnih tj. pouzdanih sistema autentifikacije korisnika. Poseban akcenat se odnosi na bankarske transakcije i sisteme koji su često meta hakerskih napada. U ovom delu data je Delone triangulacija i enkripcija slike upotrebom Katalanovog ključa (objekta) kao osnovna metoda autentifikacije. Predstavljen je scenario u kome je uključen i potencijalni napadač.

4.2.1 Voronoi - Delone triangulacija slike

Cilj Delone triangulacije slike je njena razgradnja u neukrštene trougaone elemente. Triangulacija je procedura koja se koristi za obradu tačaka koje imaju slučajnu raspodelu [26]. Tačke poligona Voronoi razdvajaju bilo koju tačku od najbližih tačaka suseda. Stranice Voronoi poligona sadrže simetrale segmenta koji se dobija spajanjem tačke sa susednim tačkama, gde se svaka tačka kombinuje sa susednim tačkama kako bi se dobila Delone triangulacija. Ideja autentifikacije zasniva se na Delone triangulaciji izabrane slike i bojenju trougolova

na osnovu zadate slike. Tako triangulisanu i obojenu sliku treba ponovo triangulisati na taj način što se koordinate trouglova koduju sa Katalanovim ključem. Navedena su neka od korisnih svojstava Dealunay triangulacije: jedinstvenost i nezavisnost od početne tačke, formirani trouglovi su u obliku jednakostraničnih trouglova, nema druge tačke u kružnicama trouglova (svojstvo kružnice), segment koji se dobija iz najbližeg para tačaka je u triangulaciji, segment koji se dobija iz tačke i njene najbliže tačke je strana trougla u triangulaciji. Na Slici 4.9 dat je primer jedne Delone triangulisane slike sa obojenim trouglovima. Scenario kodovanja će kasnije biti detaljno objašnjen.

(a) Originalna slika (b) Delone triangulisana slika

Slika 4.9: Delone triangulacija slike

4.2.2 Neka istraživanja na ovu temu

Autentifikacja putem enkriptovane slike nije nova ideja. U svom radu [27] Luan Guangyu predložio je novu šemu šifrovanja i autentifikacije asitmetričnih slika zasnovanu na jednakim modulima raspada u Fresnelovom transformacijskom domenu. Prednosti ovakve šeme su višestruke, prvo Fresnelov spektar otvorenog spektra je retko uzorkovan. Zatim je retka prezentacija Fresnelovog spektra podeljena u dve kompleksne vrednosti sa jednakim modulom raspada, od kojih su obe potrebne za dešifrovanje i autentifikaciju.

Lin Yuan u svom radu [28] je razvio šemu autentifikacije putem slike zasnovanu na enkripciji dvostrukih slika i delimičnom dešifrovanju faze u neparabilnom frakcijskom Fourierovom transformacijskom domenu. Samo deo informacije faze šifrovanog rezultata se čuva za dešifrovanje, dok se ostatak faze i sve druge informacije o amplitudama odbacuju. U radu [29] Huaqian Yang predlaže brzu enkripciju slike u procesu autentifikacije. Konkretno, ključna *hash* funkcija je uvedena da generiše 128-bitnu *hash* vrednost iz obične slike i tajnih ključeva. Vrednost hash-a igra ulogu ključa za enkripciju i dešifrovanje, dok se tajni ključevi hash-a koriste za proveru autentičnosti dešifrovane slike.

Kada je primena Delone triangulacije u pitanju postoje mnoge ideje autentifikacije triagulisanjem slike otisaka prstiju. U tom smislu u radu [30] Zanobya N. Khan je razvio ideju razdvajanje linija dlana. Zatim se određuju krajne tačke ovih linija i stvorena je veza između njih koristeći Delone triangulaciju čime se generiše izrazita topološka struktura svakog otiska dlana. Nakon toga se izdvajaju različite geometrijske i kvantitativne karakteristike iz trouglova Delone triangulacije koje pomažu u identifikaciji različitih pojedinaca. U radu [31] je razvijena tehnika Delone triangulacije slike otisaka prstiju na bazi podudaranja kako bi se izbegla greška autentičnosti uzrokovana netačnim upisima i slanjem OTP (eng. One-Time Pasword) i kako bi se osigurala maksimalna sigurnost u proveri autentičnosti. Ovaj metod triangulacije omogučava nesmetan pristup ovlašćenim korisnicima na bankomatima čak i sa izmenjenim otiscima prstiju, a isto tako i poboljšava sigurnost.

4.2.3 Konkretan opis metoda enkripcije Delone triangulisane slike i scenario autentifikacije

Proces pomoću koga se utvrđuje da li korisniku koji pristupa sistemu možemo dozvoliti pristup poznat je kao autentifikacija. Ona može biti lokalna i udaljena. Razlika je ta što prilikom procesa udaljene autentifikacije dolazi do prenosa korisničkih podataka kroz komunikacioni kanal dok se kod lokalne autentifikacije kompletan proces odvija u lokalnoj mreži. Udaljena autentifikacija korisnika sa bankarskim sistemom je u poslednje vreme najviše izložena hakerskim napadima.

Ovi napadi su često manifestovani upadom neovlašćenog NN lica u komunikacioni kanal u trenutku razmene lozinke i njene potrvde između korisnika u bankarskog sistema. Posebno su osetljive novčane transakcije gde napadač hvata i šalje sistemu potvrdnu poruku korisnika. Ova metoda enkripcije slike zasniva se na Delone triangulaciji izabrane slike, tj. kriptovanju koordinata slučajno izabrane slike i bojenju trouglova.

Kada je u pitanju Delone triangulacija slike, koristiće se inkrementalni algoritam koji je zasnovan na slučajnom izboru tačaka i međusobnoj proveri trouglova tj. da li svako teme trougla leži na kružnici, što predstavlja osnovu Delone triangulacije. Bojenje trouglova se odvija na taj način što se prepozna boja piksela na sredini trougla i tom bojom se oboji trougao triangulisane slike. Kao rezultat, dobija se nova slika koja je triangulisana i čije će koordinate temena trouglova predstavljati osnovu za enkripciju Katalanovim ključem (objektom).

U nastavku je detaljnije objašnjen algoritam šifrovanja slike. Ulazni elementi su slučajno odabrana temena, odnosno njihove (x, y) koordinate. Kreira se početna triangulacija $\mathcal{T}$ koja sadrži trougao $\triangle p_{-1}p_{-2}p_{-3}$. Ovo je zapravo pomoćni trougao od koga se polazi. On kasnije gubi na važnosti jer nije potreban. Drugim rečima, uklanjaju se njegova temena kao i sve stranice koje njegova tri temena sadrže.

U sledećem koraku formira se početni trougao slike $\triangle p_i p_j p_k$. Treba naglasiti da su koordinate (x, y) svakog temena ovog trougla, kao i sve naredne koje će se kreirati, u okviru

rezolucije slike. To je uslov potreban da bi se slika pravilno triangulisala. Proces kreiranja ovog trougla završava se nakon 3 brojačke petlje. U svakom brojanju, koordinate temena p_r dodaju se nizu K koji nam je potreban u daljem procesu šifrovanja koordinata. Nakon formiranja početnog trougla u procesu triangulacije, funkcija ***size()*** poziva se kako bi se pronašao piksel sa (x, y) koordinatama smeštenim u središtu trougla (cetntralni piksel trougla). Način na koji ova funkcija pronalazi spomenuti piksel predstavljen je standardnom formulom za izračunavanje koordinata od težišta trougla.

$$T = \left(\frac{x1 + x2 + x3}{3}, \frac{y1 + y2 + y3}{3} \right) \tag{4.1}$$

Treba napomenuti da se funkcija ***size()*** poziva za svaki novokreirani trougao slike. U okviru ove funkcije se u stvari poziva RGB funkcija koja boji trougao bojom središnjeg piksela trougla. Takođe treba naglasiti da se za sledeći unos slučajnog temena p_r mogu očekivati dva slučaja o njegovom položaju s obzirom na $\triangle p_i p_j p_k$ i unutar velikog trougla $\triangle p_{-1} p_{-2} p_{-3}$.

U prvom se slučaju p_r nalazi se unutar trougla $\triangle p_i p_j p_k$. U drugom slučaju, p_r je na ivici $\triangle p_i p_j p_k$. U svakom od ovih koraka upućuje se poziv prema LegalizeEdge algoritmu (predstavljenom kao LegalizujIvicu).

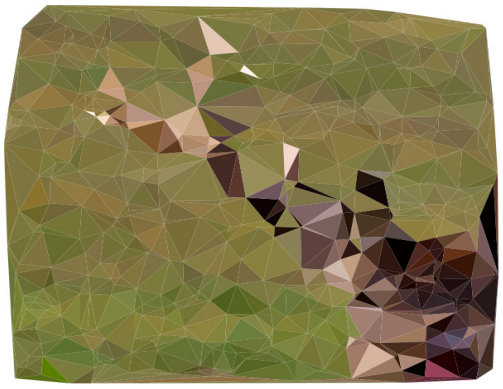
Nakon što su ivice legalizovane, odnosno nakon završetka postupka triangulacije, uklanjaju se temena i ivice velikog trougla $\triangle p_{-1} p_{-2} p_{-3}$ jer nam je veliki trougao trebao samo za kreiranje početne triangulacije slika $\triangle p_i p_j p_k$. U sledećem koraku, kao povratnu vrednost, dobijamo triangulisanu sliku $\mathcal{D}$, odnosno $\mathcal{D}$ je skup svih trouglova dobijenih postupkom triangulacije.

Sada treba naglasiti da je $\mathcal{D}$ (triangulisana slika) zapravo skup toruglova koji je ustvari podskup skupa $\mathcal{T}$ (veliki trougao) pomenut u definiciji 2. Izbacivanjem velikog trougla $\mathcal{T}$ uklanjaju se delovi slike koji su izvan konveksnog omotača tj. izvan rubnih ivica $\mathcal{D}$.

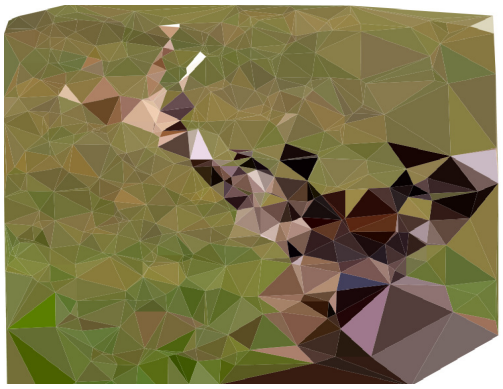
Sada se elementi niza K, odnosno vrednosti koordinata (x, y) temena, pretvaraju u binarni zapis. Ovo je uslov da se pokrene metoda *Stack Permutation*. Zatim odabere se odgovarajući Katalanov objekat i pomoću metode *Stack Permutation* šifruju se koordinate i smeštaju se u niz K_s. Od sada se postupak triangulacije slike ponavlja kao u prethodnim koracima, osim što se oznake mijenjaju (dodato je "s") zbog jasnoće. Rezltat rada opisanog srenarija je Slika 4.10

Proces dekripcije tj. vraćanje slike u prvobitno triangulisan oblik dobija se tako što šifrovane koordinate temena menjaju mesto sa originalnim (prvobitno slučajno izabranim) u metodama pri pozivu za dešifrovanje. Na taj način se dobija prvobitno triangulisana slika. Sa Slike 4.10 se jasno može uočiti razlka između originala i šifrata.

Scenario autentifikacije je zamišljen tako da se za svakog korisnika, pored korisničkog imena i lozinke (deo lozinke mora biti Katalanov ključ (objekat) koji dodeljuje sistem i koji znaju samo korisnik i sistem atutentifikacije), u tabeli čuvaju originalna slika, Dalaunay



(a) Delone triangululacija slike



(b) Delone triangulisana-enkriptovana slika

Slika 4.10: Delone trinagualcija i Delone enkriptovana-triangulisana slika

triangulisana slika, i Delone triangulisana-enkriptovana slika. Zapravo čuvaju se nizovi sa koordinatama temena trouglova triangulisane i enkriptovane triangulisane slike. Ovi nizovi su nam potrebni jer na osnovu njihovih indeksa se vrši kasnije provera korisnika. Drugim rečima potvrđuje se njegova autentifikacija.

Tabela 4.2: Korisnička tabela autentifikacije korisnika Bankarskog sistema

Id Korisnika	Korisničko ime	Lozinka	Katalan objekat (ključ)	Delauany triangulacija slike	Enkriptovana - Delaunay triangulacija slike	Koordinate temena Delaunay triangulisane slike (Niz "K")	Koordinate temena Enkriptovane - Delaunay triangulisane slike (Niz "Ks")	Vrednosti koordinata u datim indeksima niza "Ks"
1	pera	peric2816098	2816098=10101011 11100001	image.jpg	del_image.jpg	teme 1: X=124, Y=439 teme 2: X=397, Y=114 teme 3: X=26, Y=2 teme 4: X=144, Y=510 teme 5: X=408, Y=265 teme 6: X=491, Y=194 teme 7: X=322, Y=14 teme 8: X=344, Y=26 teme 9: X=268, Y=157 teme 10: X=349, Y=477	teme 1: X=244, Y=367 teme 2: X=391, Y=120 teme 3: X=200, Y=8 teme 4: X=66, Y=510 teme 5: X=450, Y=385 teme 6: X=443, Y=26 teme 7: X=280, Y=140 teme 8: X=464, Y=200 teme 9: X=388, Y=199 teme 10: X=469, Y=471	teme 4: X=66, Y=510 teme 7: X=280, Y=140 teme 2: X=391, Y=120

U nastavku su data dva algoritma. Algoritam 4.2.1 kreira triangulaciju slike, dok Algoritam 4.2.2 kriputuje triangulisanu sliku.

Algoritam 4.2.1 Delone triangulacija slike $(p_r, \overline{p_i p_j}, \mathcal{T})$

Ulaz: Slučajno izbrana slika i skup temena (tačaka) P.
1: Kreiraj inicijalnu truaingulaciju $\mathcal{T}$ koja je u stvari $\triangle p_{-1}$, p_{-2}, p_{-3}.
2: **for** $r = 1$ to n **do** (Unesi p_r u $\mathcal{T}$)
Pronađi $\triangle p_i p_j p_k \in \mathcal{T}$, koji sadrži p_r
Unesi p_r u niz K.
if p_r u unutrašnjosti $\triangle p_i p_j p_k$
then
LegalizeEdge(p_r, $\overline{p_i p_j}$, $\mathcal{T}$)
Pozovi **size()** funkciju za pronalazak centralnog piksela $\triangle p_r p_i p_j$
Pozovil **RGB** funkciju za bojenje $\triangle p_r p_i p_j$
LegalizeEdge(p_r, $\overline{p_j p_k}$, $\mathcal{T}$)
Pozovi **size()** funkciju za pronalazak centralnog piksela $\triangle p_r p_j p_k$
Pozovi **RGB** funkciju za bojenje $\triangle p_r p_j p_k$
LegalizeEdge(p_r, $\overline{p_k p_i}$, $\mathcal{T}$)
Pozovi **size()** funkciju za pronalazak centralnog piksela $\triangle p_r p_k p_i$
Pozovi **RGB** funkciju za bojenje $\triangle p_r p_k p_i$
else (p_r na ivici $p_i p_j p_k$, kaže se na ivici $\overline{p_i p_j}$)
LegalizeEdge(p_r, $\overline{p_i p_l}$, $\mathcal{T}$)
Pozovi **size()** funkciju za pronalazak centralnog piksela $\triangle p_r p_i p_l$
Pozovi **RGB** funkciju za bojenje $\triangle p_r p_i p_l$
LegalizeEdge(p_r, $\overline{p_l p_j}$, $\mathcal{T}$)
Pozovi **size()** funkciju za pronalazak centralnog piksela $\triangle p_r p_l p_j$
Pozovi **RGB** funkciju za bojenje $\triangle p_r p_l p_j$
LegalizeEdge(p_r, $\overline{p_j p_k}$, $\mathcal{T}$)
Pozovi **size()** funkciju za pronalazak centralnog piksela $\triangle p_r p_j p_k$
Pozovi **RGB** funkciju za bojenje $\triangle p_r p_j p_k$
LegalizeEdge(p_r, $\overline{p_k p_i}$, $\mathcal{T}$)
Pozovi **size()** funkciju za pronalazak centralnog piksela $\triangle p_r p_k p_i$
Pozovi **RGB** funkciju za bojenje $\triangle p_r p_k p_i$
3: Odbaci temena p_{-1}, p_{-2} i p_{-3} uključujući i stranice koje su sadržane u $\mathcal{T}$.
4: **Izlaz:** $\mathcal{D}$ ($\mathcal{D}$ je triangulisana slika ili podskup trouglova sadržan u $\mathcal{T}$).

Treba naglasiti da je rezultat algoritma 4.2.1 prikazan na Slici 4.10 (a). Nakon triangulacije slike, pokreće se drugi algoritam.

Rezultat rada Algoritma 4.2.2 predstavljen je na Slici 4.10 (b). Slika 4.11 predstavlja postupak provere autentičnosti u 5 koraka koji će u nastavku biti detaljno objašnjeni.

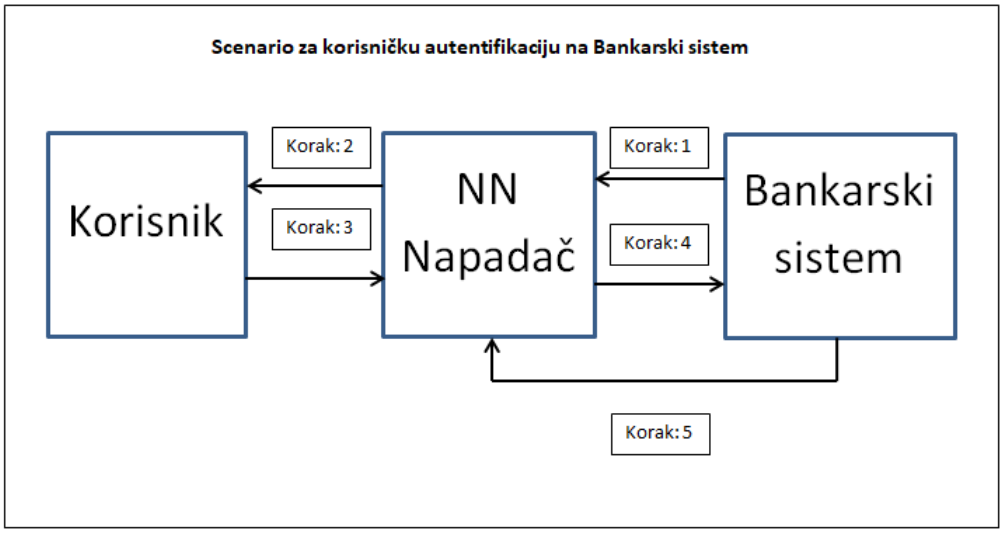
Korak 1: U prvom koraku, nakon zahteva korisnika za autentifikacijom, bankarski sistem prvo identifikuje korisnika sa slučajno dodeljenim Katalanovim ključem (numerički deo lozinke, npr. ako je korisničko ime: *"pera"* a lozinka:*"peric2816098"* Katalanov ključ je 2816098 u dekadnom sistemu dok je u binarnom 10101011111000011000010) zatim slučajno

Algoritam 4.2.2 Enkripcija Delone triangulisane slike $(p_r, \overline{p_i p_j}, \mathcal{T})$

Ulaz: Triangulacija $\mathcal{D}$ kao rezultat rada Algoritma 1.3.1 i niz K.

1: **for** $r = 1$ to n **do** (Pristupi p_r u nizu K)
2: Konvertuj p_r u binarni zapis
3: Pozovi **Stack permutation** metodu i izabran **Catalan ključ (objekat)** .
4: Konvertuj p_s u decimalni zpis (nakon permutacije bita p_r postaje p_s)
5: Unesi p_s u niz K_s
6: Kreiraj inicijalnu triangulaciju $\mathcal{T}_s$ koja sadrži $\triangle p_{-1}$, p_{-2} i p_{-3}.
7: **for** $s = 1$ to n **do** (Ubazi p_s u $\mathcal{T}_s$)
Pronađi $\triangle p_i p_j p_k \in \mathcal{T}_s$, koji sadrži p_s
if p_s u unutrašnjosti $\triangle p_i p_j p_k$
then
LegalizeEdge(p_s, $\overline{p_i p_j}$, $\mathcal{T}_s$)
Pozovi **size()** funkciju za pronalazak centralnog piksela $\triangle p_s p_i p_j$
Pozovi **RGB** funkciju za bojenje $\triangle p_s p_i p_j$
LegalizeEdge(p_s, $\overline{p_j p_k}$, $\mathcal{T}_s$)
Pozovi **size()** funkciju za pronalazak centralnog piksela $\triangle p_s p_j p_k$
Pozovi **RGB** funkciju za bojenje $\triangle p_s p_j p_k$
LegalizeEdge(p_s, $\overline{p_k p_i}$, $\mathcal{T}_s$)
Pozovi **size()** funkciju za pronalazak centralnog piksela $\triangle p_s p_k p_i$
Pozovi **RGB** funkciju za bojenje $\triangle p_s p_k p_i$
else (p_s je na ivici $p_i p_j p_k$, kaže se na ivici $\overline{p_i p_j}$)
LegalizeEdge(p_s, $\overline{p_i p_l}$, $\mathcal{T}_s$)
Pozovi **size()** funkciju za pronalazak centralnog piksela $\triangle p_s p_i p_l$
Pozovi **RGB** funkciju za bojenje $\triangle p_s p_i p_l$
LegalizeEdge(p_s, $\overline{p_l p_j}$, $\mathcal{T}_s$)
Pozovi **size()** funkciju za pronalazak centralnog piksela $\triangle p_s p_l p_j$
Pozovi **RGB** funkciju za bojenje $\triangle p_s p_l p_j$
LegalizeEdge(p_s, $\overline{p_j p_k}$, $\mathcal{T}_s$)
Pozovi **size()** funkciju za pronalazak centralnog piksela $\triangle p_s p_j p_k$
Pozovi **RGB** funkciju za bojenje $\triangle p_s p_j p_k$
LegalizeEdge(p_s, $\overline{p_k p_i}$, $\mathcal{T}_s$)
Pozovi **size()** funkciju za pronalazak centralnog piksela $\triangle p_s p_k p_i$
Pozovi **RGB** funkciju za bojenje $\triangle p_s p_k p_i$
8: Odbaci temena p_{-1}, p_{-2} i p_{-3} sa stranicama koje ona sadrže unutar $\mathcal{T}_s$.
9: **Izlaz:** $\mathcal{D}_s$ ($\mathcal{D}_s$ je enkriptovana triangulisana slika ili podskup trouglova u skupu $\mathcal{T}_s$.

izabere neku sliku i broj tačaka tj. temena trouglova za triangulaciju. Nakon toga se inkrementalnim algoritmom vrši Delone triangulacija slike i koordinate temena se čuvaju u nizu. Na primer ako je izabrana neka slika i triangulisana sa 10 slučajnih tačaka, koordinate temena koje bi se čuvale u Delone nizu bi bile: $p_1 = (124, 439)$, $p_2 = (397, 114)$, $p_3 = (26, 2)$, $p_4 = (144, 510)$, $p_5 = (408, 265)$, $p_6 = (491, 194)$, $p_7 = (322, 14)$, $p_8 = (344, 26)$, $p_9 = (268, 157)$, and $p_{10} = (349, 477)$.



Slika 4.11: Scenario korisničke autentifikacije na Bankarski sistem

Korak 2: U drugom koraku se tako triangulisana slika kriptuje numeričkim delom lozinke, koji je ujedno i Katalanov ključ, metodom *stek permutacije* objašnjenom u sekciiji 1.3. Dobijamo sada enkriptovanu triangulisanu sliku sa Delone nizom kriptovanih koordinata sa takođe 10 temena: $p_{s1} = (244, 367)$, $p_{s2} = (391, 120)$, $p_{s3} = (200, 8)$, $p_{s4} = (66, 510)$, $p_{s5} = (450, 385)$, $p_{s6} = (443, 26)$, $p_{s7} = (280, 140)$, $p_{s8} = (464, 200)$, $p_{s9} = (388, 199)$, and $p_{s10} = (469, 471)$. Ova slika i niz se takođe čuvaju u bazi jer su kasnije veoma važni u procesu potvrde autentifikacije korisnika. Sada prvobitno triangulisanu sliku, sa provobitnim nizom od 10 temena bankarski sistem šalje preko komunikacionog medijuma (Interneta) korisniku. U ovom trenutku nastupa potencijalni napadač koji ima priliku da uhvati datoteku sa triangulisanom slikom i nizom koordinata od 10 temena. Obzirom da se u ovom trenutku ne traži nikakva potvrda, napadač jednostavno propušta da datoteka stigne do korisnika.

Korak 3: U ovom koraku korisnik prihvata datoteku sa tringulisanom slikom, nizom koordinata od 10 temena i sa istim Katalanovim ključem (jer samo on i banka znaju da je numerički deo lozinke Katalanov ključ) enkriptuje dobijenu sliku i dobija istu Delone enkriptovanu sliku i enkriptovane koordinate temena kao u koraku 2. U ovom trenutku korisnik šalje enkriptovanu sliku bankarskom sistemu ali ne šalje niz sa enkriptovanim koordinatama. Takođe napadač hvata poslatu sliku i prosleđuje je bankarskom sistemu. Treba naglasiti da napadač nema informaciju o kriptovanim koordinatama temena.

Korak 4: Ovaj korak je rezervisan za proveru autentičnosti Delone triangulisane enkriptovane slike koju je poslao korisnik i one koja je pretnohdno enkriptovana i sačuvana od strane bankarskog sistema. Obzirom da su slike iste prelazi se na još jednu potrvdu autentičnosti korisnika.

Korak 5: Obzirom na činjenicu da napadač, i ako je uhvatio i prosledio enkriptovanu sliku bankarskom sistemu, nema informaciju o vrednostima kriptovanih koordinata,

bankarski sistem šalje zahtev za unosom koordinata 3 slučajna indeksa kriptovanog Delone niza npr. vrednosti (x, y) koordinata sa indeksima 4,7,2. U ovom slučaju bi napadač trebao da unese vrednosti: $p_{s4} = (66, 510)$, $p_{s7} = (280, 140)$, and $p_{s2} = (391, 120)$ što je nemoguće. Ukoliko sistem ne dobije povratnu informaciju o tačno unetim koordinatama onda on to razume da je napadač umešan u procesu autentifikacije i obustavlja dalje akcije. Ipak, ako dobije korektne vrednosti traženih koordinata onda se odobrava transakcija i to znači da napadač nije učestvovao u procesu autentifikacije.

4.2.4 Eksperimentalna istraživanja i rezultati

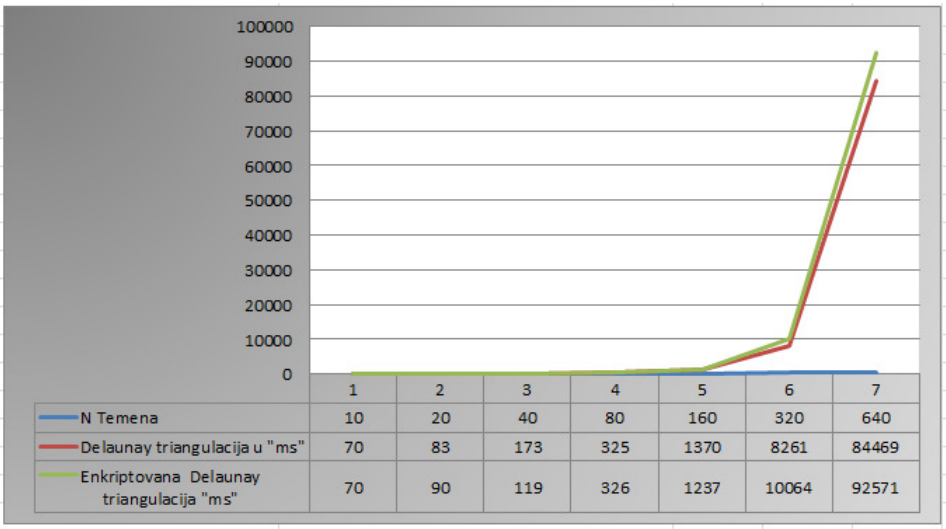
Vreme Delone triangulacije slike i njene enkripcije testirano je na temenima trouglova od *N=20,40,80,160,320,640*. Takodje, ovaj Delone inkrementalni algoritam, modifikovan enkripcijom *Stek permutacije* po principu *LIFO*, je implementiran u Java NetBeans okruženju.

Tabela 4.3: Vreme enkripcije temena u "ms"

N Temena	**Delone trinagulacija u "ms"**	**Enkriptovana Delone triangulacija u "ms"**
10	70	70
20	83	90
40	173	119
80	325	326
160	1370	1237
320	8261	10064
640	84469	92571

Ako se tabelarni rezultati predstave grafički, primetno je da vreme šifrovanja nije direktno proporcionalno broju temena trougla. Takođe, može se primetiti da se vreme enkripcije u velikoj meri ne razlikuje od vremena klasične Delone triangulacije što nam ukazuje na efikasnost metoda Stek permutacije. Testiranje je urađeno na računaru sledečih karakteristika: *Intel Core i5-CPU 2.6 GHz, RAM -8 GigaBytes, Operating system: Windows 7 Microsoft -64 bits.*

Kao što je već pomenuto, uporedo sa razvojem informacionih tehnologija, raste i mogućnost zloupotreba podataka koji se tim putem prenose. Razvoj hardvera i softvera, koji se koriste u procesu prenosa podataka, zahteva česte promene i usavršavanja istih. U svemu navedenom proizvođači nemaju vremena za detaljno testiranje opreme. Nedovoljno testiranje i predviđanje eventualnih propusta predstavlja osnovu za rad potencijalnih napadača. Shodno svemu navedenom, cilj ove metode jeste onemogućiti napadaća da pristupi bankarskom sistemu [32].

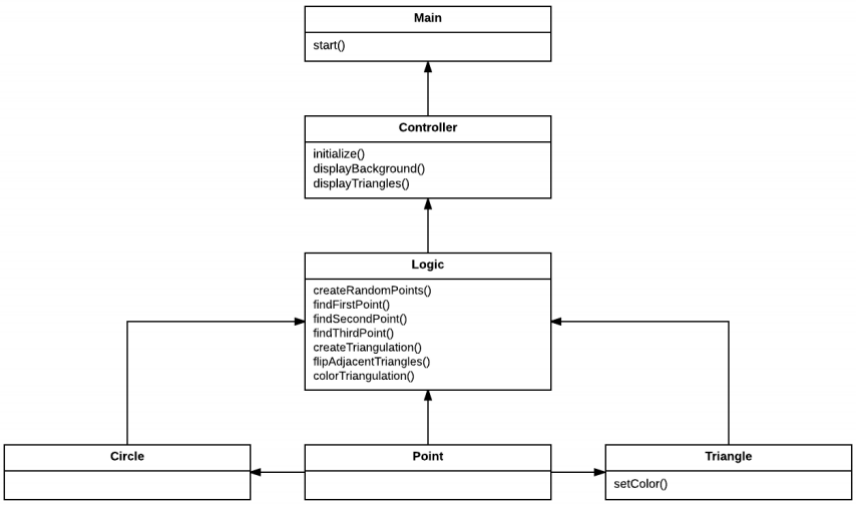


	1	2	3	4	5	6	7
N Temena	10	20	40	80	160	320	640
Delaunay triangulacija u "ms"	70	83	173	325	1370	8261	84469
Enkriptovana Delaunay triangulacija "ms"	70	90	119	326	1237	10064	92571

Slika 4.12: Grafička ilustracija vremena enkripcije

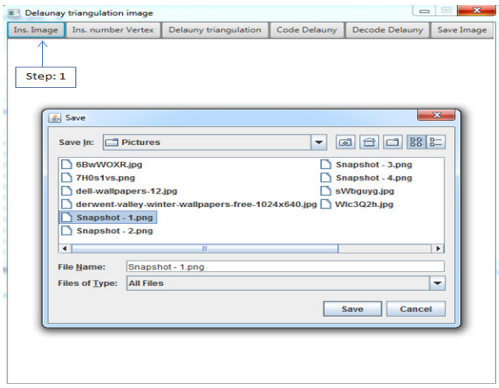
4.2.5 Implementacija predložene metode u Java - Net Beans okruženju

Ova aplikacija traba da transformiše originalnu sliku u novu sliku koja se sastoji od trouglova. Formirana triangulacija treba da bude Delone triangulacija, odnosno trouglovi obojeni na temelju centralnog piksela ulazne slike. Program je napravljen u Javi i koristi JavaFX za prikaz slike. Pre nego što se predstave koraci u radu aplikacije, date klase i metode koje su korišćene, tj. na Slici 4.13 predstavljen je dijagram klasa korišćenih u aplikaciji.

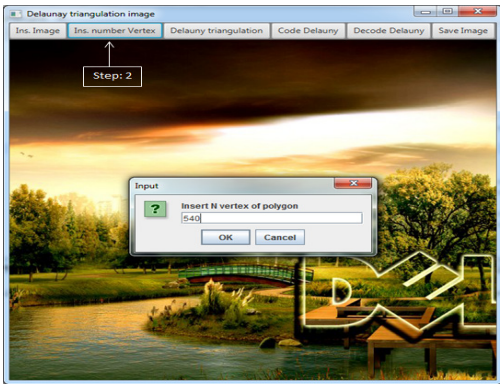


Slika 4.13: Dijagram klasa korišćenih u aplikaciji

Implemantacija predložene metode shodno prethodno objašnjenim algoritmima 4.2.1 i 4.2.2 trangulacije slike realizuje se u nekoliko koraka. Aplikacija se po strukturi klasa zasniva na [33], ali su te iste klase i metode modifikovane i prilagođene ovoj metodi enkripcije. Segmente pojedinih klasa moguće je videti u prilozima. Pokretanjem aplikcije, u pozadini se definiše Katalanov objekat. Dalje potrebno je da se u prvom koraku unese neka slučajna slika, zatim da se unese broj N temena. Na Slici 4.14 su prikazani koraci u procesu rada aplikacije.



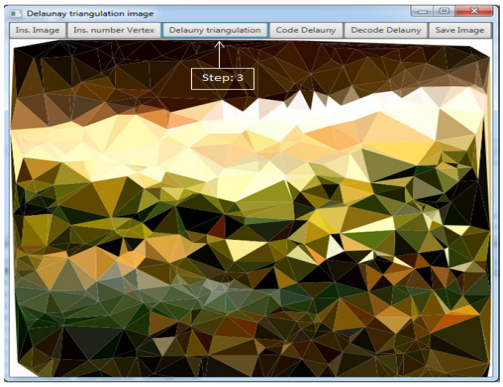
(a) Image selection



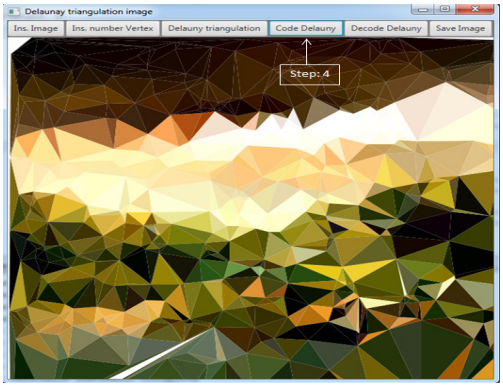
(b) Enter N number of vertex

Slika 4.14: Koraci 3 i 4 u procesu triangulacije slike

Nakon odabira slike i unosa odgovarajućeg broja temena, pokreće se proces Delone triangulacije slike, odnosno pokreće algoritam enkripcije koordinata (x, y) temena trouglova. Na kraju je zamišljeno da se u bazi čuvaju svi podaci: Delone triangulisana slika i prvobitno izabran niz sa koordinatama temena trouglova, zatim triangulisana i enkriptovana slika takođe sa enkriptovanim koordinatama.



(a) Delone triangulacija slike



(b) Enkriptovana Delone triangulacija slike

Slika 4.15: Koraci 3 i 4 u procesu triangulacije slike

Jedna od glavnih metoda je *createRandomPoints*() koja je modifikovana na način da svako slučajno izabrano teme se skladišti u nizove $S\_x\_Coordinate[i]$ i$S_y_Coordinate[i]$,

odnosno ovi nizovi čuvaju koordinate (x, y) slučajno izabranih temena. Segment koda date metode dat je u nastavku.

```
public void createRandomPoints()
    {
        if (encrypted==true)
        {
            for (int i=0; i<points.length; i++)
           {
              points[i]=new Point(D_x_coordinate[i],
                                  D_y_coordinate[i]);

                 while (unique(i)==false)
                   {
                      points[i]=new Point(random.nextInt(width),
                                          random.nextInt(height));
                   }
            }
        } else
           {
             for (int i=0; i<points.length; i++)
             {
                points[i]=new Point(random.nextInt(width),
                                    random.nextInt(height));
                S_x_coordinate[i]=(int) Math.round(points[i].x);
                S_y_coordinata[i]=(int) Math.round(points[i].y);
                 while (unique(i)==false)
               {
                 points[i]=new Point(random.nextInt(width),
                                     random.nextInt(height));
               }
             }
           }
    }
```

Takođe jedna od karakterističnih metoda koja je pomenuta u Algoritmima 4.2.1 i 4.2.2 jeste i metoda *size*() odnosno ona je u kodu nazvana kao *colorTriangulation*(). U nastavku pogledajte segment koda ove metode.

```
void colorTriangulation()
   {
      System.out.println("Bojenje trouglova (Triangulacije)");
        try
```

```
        {
         img=ImageIO.read(Selected_file_Del);
        }
      catch(IOException e)
        {
          e.printStackTrace();
        }
    System.out.println("Trinagles duzina"+triangles.size());
       for (int i=0; i<triangles.size(); i++)
       {
          int argb=img.getRGB((int)triangles.get(i).center.x,
                              (int) triangles.get(i).center.y);
         int red = (argb>>16) & 0xFF;
         int green = (argb>>8) & 0xFF;
         int blue = (argb) & 0xFF;
          Color color = Color.rgb(red, green, blue, 1);
          triangles.get(i).setColor(color);
}
```

4.3 Metoda primene Delone triangulacije i Katalanovih objekata u steganografiji

Metoda koja je predstavljena ima za cilj korišćenje slike za prenos skrivenih podataka željenom korisniku putem interneta. Ovaj algoritam steganografije, u procesu šifrovanja, temelji se na kodiranju slike u binarni zapis, pretvaranju tajne poruke (skrivenih podataka) u binarni zapis i kreiranju Delone triangulacije binarnog zapisa slike čija su temena nosači bita tajne poruke. Nakon toga, primenom metode stek permutacije i Katalanovih objekata nad koordinatama (x, y) Delone temena, dobija se potpuno nova šifrovana triangulacija čije su koordinate temena (x, y) smeštene u niz.

Izvorna slika, šifrovani niz sa koordinatama temena (x, y) u obliku $Base64$ koda i šifrovana Delone triangulacija, šalju se korisniku putem medijuma (interneta). Konačno, u procesu dešifrovanja, ponovnom primenom Katalanovog objekta i metode stek permutacije, stvara se izvorna Delone triangulacija binarnog zapisa slike i otkrivaju se izvorne informacije.

Takođe je predstavljen postupak rastavljanja stego ključa kako bi napadač bio zbunjen prilikom kreiranja metode napada. Osnovno pitanje je sledeće: kako napadač može dobiti tajnu poruku čak i ako u nekim slučajevima uspe povezati delove u stego ključu? Glavna motivacija za realizaciju ove metode bila je kako sakriti poruku na slici i ne menjati joj oblik.

4.3.1 Slična istraživanja na osnovu ove metode

Autori su radili u svojim prethodnim istraživanjima s Katalanovim brojevima i objektima na mnogim područjima, poput kriptografije, steganografije, računarske geometrije i kombinatorike. Nekoliko primena kombinatorike u steganografiji predstavljeno je u [11, 25, 35]. U [10] autori su proučavali svojstva Katalanovih brojeva i put rešetke, kao i njihovu primenu u kriptografiji. U radu [12] istraživane su primene Katalanovih brojeva u steganografiji. Dodatne tehnike skrivanja podataka utemeljene na binarnim (Dyck) rečima pojavile su se u radu [36].

U [37] metodi steganografije koriste se razgradnja vrednosti intenziteta piksela i dobro poznate sekvence brojeva (Fibonacci, Lucas i Catalan – Fibonacci). Drugim rečima, definisana je kao metoda koja koristi dva generatora slučajnih brojeva. Prvi je generator Katalonovih brojeva, dok drugi generiše Lucasove sekvence brojeva. Popis tehnika steganografije slika predstavljen je u [38]. Rezultati kombinatorike takođe se mogu koristiti u steganografiji kako je predstavljeno u [39]. U [40] predstavljena je metoda tajnog ključa koja se temelji na slučajnim brojevima. Vrhunska sigurnost glavna je prednost predložene tehnike jer su izvorna slika i stego slike nepromijenjene.

U radovima [41, 42, 43] je predložena metoda steganografije slika koja poboljšava sposobnost ugrađivanja u stego objekte. Ova metoda, koja se temelji na preklapanju piksela, koristi i funkciju modula i preklopljeni PVD. Broj bitova po pikselu, kao i vreme izvršavanja, upoređuju se sa postojećim metodama. Prikladna RS analiza pokazuje sigurnost metode. Osim toga, multi-stego metoda slike predstavljena u [44] koristi modificirano najmanje značajno podudaranje bitova kako bi se poboljšao kapacitet ugradnje kao i kvalitet slike. Tehnika izbacivanja pojedinih elemenata iz slike radi skrivanja podataka unutar nje predložena je u [45].

Nedavno su autori [25] analizirali mnoge tehnike steganografije i šifrovanja slike. Jedna od spomenutih tehnika je iterativna rekonstrukcija sheme za kompresiju šifrovanja binarnih slikovnih datoteka, koristeći slučajno Markovljevo polje za karakterizaciju odgovarajućih običnih slika u prostornom domenu. Uz to, u odnosu na Markovljeva slučajna polja, dekodiranje i dekompresija temelje se na metodologiji faktorskih grafova, koristeći dvodelni graf koji predstavlja faktorizaciju funkcija u nekim podfunkcijama [46].

4.3.2 Opis predložene metode steganizacije

Ova metoda se po svom funkcionisanju temelji na standardnom principu steganografskog sistema u odnosu na steganografski medij = skrivena poruka + nosilac poruke + steganografski ključ. Uslov svih uslova je da pošaljilac i primalac znaju vrednost Katalanovog objekta, jer je on glavni faktor u procesu zaštite tajne poruke od potencijalnog napadača. U nasatavku će biti predstavljeno kako sistem u suštini funkcioniše, kao i delimični opis stego metode [36].

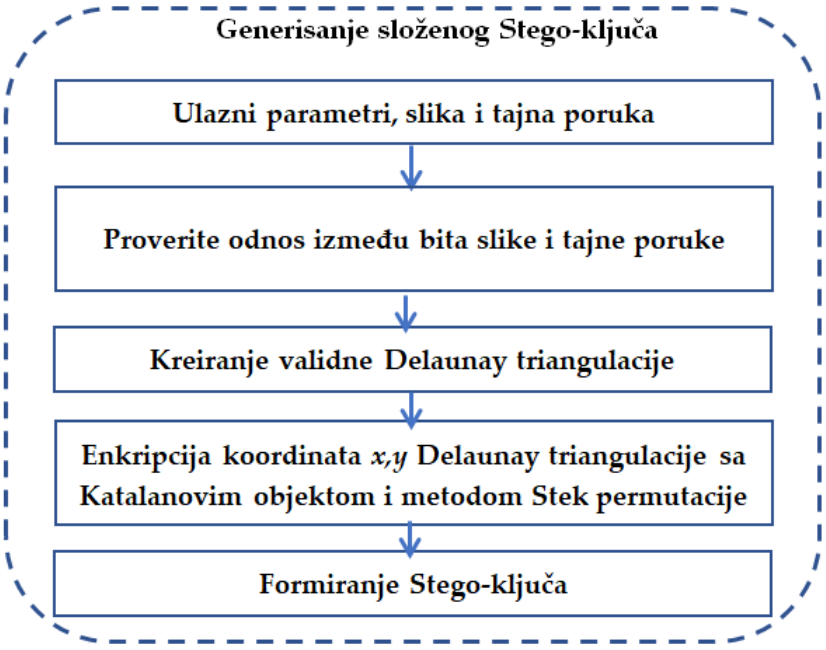
U prvoj fazi odabere se slika date rezolucije. Zatim se ta slika konvertuje u Base64 kod, a zatim u binarni zapis. Nakon izbora slike, u sistem se unosi tajna poruka koja se takođe prevodi u binarni zapis.

U drugoj fazi proverava se odnos 0 i 1 bita na slici i u tajnoj poruci. Važno je napomenuti da broj 0 bitova koji se pojavljuju na slici uvek mora biti veći ili jednak broju 0 bitova koji se pojavljuju u poruci. Isto vredi i za pojavu 1 bita. Na taj smo način ispitali maksimalnu dužinu tajne poruke koja se može ugraditi u odnosu na zadatu rezoluciju slike.

U trećoj fazi potrebna je validna Delone triangulacija (tj. nasumično se biraju temena sa (x, y) koordinatama koji moraju odgovarati zadatom bitu iz poruke).

U četvrtoj fazi, (x, y) koordinate temena Delone triangulacije šifruju se metodom stek permutacije i Katalanovim objektom (objašnjeno u poglavlju 2). Na taj način dobijamo šifrovani niz R_k koji je preveden u $Base64$ kodu.

U petoj fazi, izvorna slika, šifrovanu Delone triangulaciju i niz R_k (detaljno objašnjeni u sljedećem pasusu) u $Base64$ kodu kombinuju se u stego ključ. Na Slici 4.16 predstavljen je metod implementacije tajne poruke u slici.

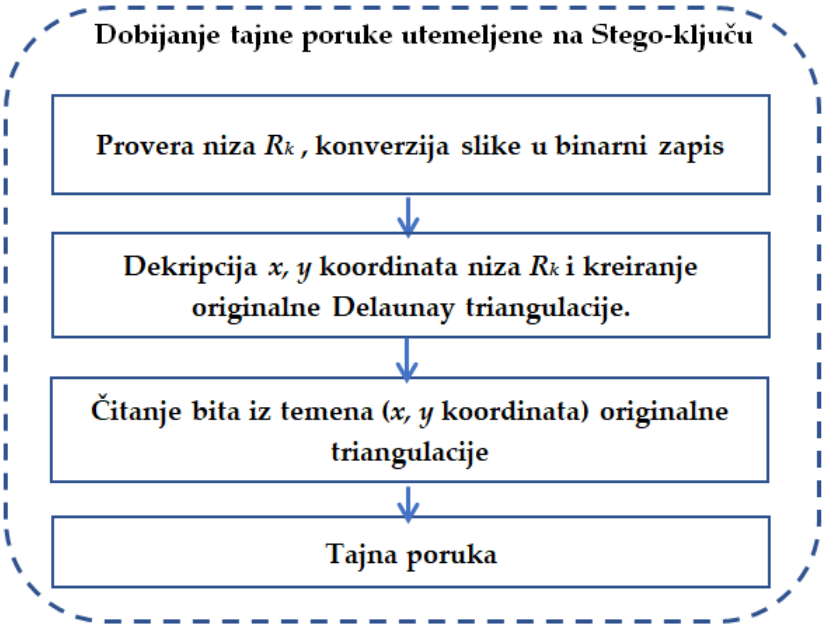


Slika 4.16: Proces implementacije tajne poruke u slici

U nastavku sledi postupak dobijanja tajne poruke iz stego ključa. U prvom koraku konvertuje se niz R_k sa koordinatama iz $Base64$ u celi broj. Zatim od njega se kreira Delone triangulacija i tako se proverava jesu li kreirana i dobijena Delone triangulacija jednake (tj. je li niz R_k korektan). Nakon toga se rezultirajuća izvorna slika pretvara u $Base64$, a zatim u binarni zapis.

U drugom koraku, koristeći Katalanov objekat i stek permutaciju, dešifruje se niz R_k i dobija se izvorna triangulacija.

U trećem koraku se čitaju bitovi slike koji su dobijeni iz temena Delone triangulacije. U četvrtom koraku ti dobijeni bitovi se dele u grupe (bajtove) od 8 bita. Tako dobijeni bajtovi od bitova pretvaraju se u tekst, čime se dobija tajna poruka. Na Slici 4.17 predstavljen je metod implementacije tajne poruke.

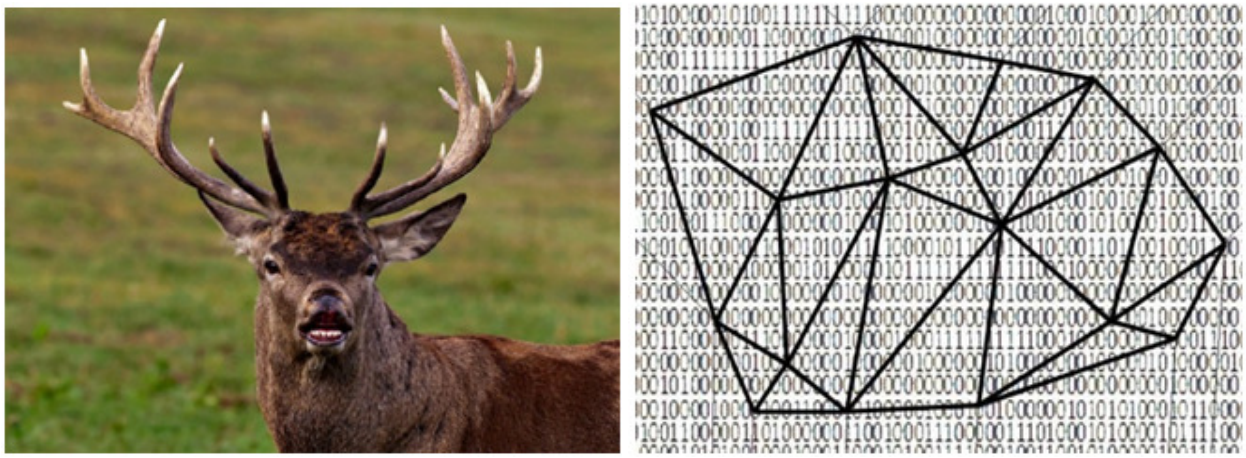


Slika 4.17: Proces dobijanja tajne poruke iz slike

4.3.3 Jedan primer metode steganizacije sa detaljima

Na samom početku potrebno je slučajno odabranu sliku pretvoriti u binarni zapis. Da bi se pretvorila odabrana slika u binarni zapis, potrebno ju je prvo pretvoriti u $Base64$ kod, a tek onda u binarni zapis. Polazna tačka ima koordinate (x, y) svakog temena Delone triangulacije, koji je nosilac bita "0" ili "1" tajne poruke. Ova metoda šifrovanja binarnog zapisa slike temelji se na primeni inkrementalnog algoritma Delone triangulacije (tj. šifrovanju koordinata (x, y) svakog nasumično dobijenog temena koje nosi jedan bit tajne poruke "0" ili "1" . Kao što je već spomenuto, postupak šifrovanja koordinata temena, temelji se na primjeni metode permutacije steka i binarnog zapisa Katalanovog objekata. Na Slici 4.18 je dat primer konverzije slike i njenog triangulisanog binarnog zapisa.

Sada se može predstaviti metoda šifrovanja skrivenih podataka na slici. Prvi je korak konvertovati željene informacije koje se žele sakriti u binarni zapis. U ovo primeru, treba sakrito reč "ENCRYPTION" = 01000101 01001110 01000011 01010010 01011001 01010000 01010100 01001001 01001111 01001110. Može se primetiti da se svako slovo konvertuje u 8



(a) Selektovana slika (b) Triangulacija na binirarni zapis slike

Slika 4.18: Primer Delone triangulacije binarnog zapisa slike

bitova. Takođe je potrebno uneti svaki bit ove reči u jedan niz S, jer bitovima se kasnije mora pristupiti po algoritmu kako bismo proverili njihove vrednosti. U drugom koraku bira se slika koju pretvaramo u binarni zapis. Svaki piksel ove slike predstavlja jedan bit "0" ili "1" koji će biti potencijalni nosilac tajne poruke. Važno je napomenuti da se može izbrati bilo koja slika definisane rezolucije. Rezolucija je važna jer je povezana sa Katalanovim objektom (npr. ako Katalanov objekat ima, na primer, $2n$ bitova, može pretvoriti koordinate (x, y) temena Delone triangulacije koji se nalaze u rasponu od $2n/2$ bita.

U trećem koraku se pristupa procesu kreiranja Delone triangulacije nad binarnim reprezentom slike. Obzirom da sada postoji broj bitova skrivene informacije (dužina niza S iz koraka 1) kao i sami bitovi u nizu, ova Delone triangulacija će imati onoliko temena koliko je u stvari dužina niza S. Drugim rečima, broj bitova skrivene informacije u ovom sličaju je 8 (karaktera)*10=80 bita, tako da je potrebno 80 temena sa (x, y) kordinatama da bi se formirala Delone triangulacija. Temena Delone triangulacije se biraju slučajno primenom inkrementalnog algoritma [2].

Prilikom nasumičnog izbora temena, uvodi se kontrola da svakom temenu odgovara po jedan bit iz niza bitova S koji nose skrivenu informaciju. Ako se to ne desi, proces izbora temena se ponavlja pozivom funkcije $random()$ u Algoritmu 4.3.1 i vraćanje na početak petlje pomoću $goto()$ funkcije, sve dok se ne nađe odgovarajući bit i tako do kraja dok se ne kreira Delone triangulacija. Koordinate (x, y) svakog temena nose po jedan bit skrivene informacije i čuvaju se u skupu R koji nam je potreban zbog procesa enkripcije.

Četvrti korak je rezervisan za metod steck permutacije kojim se uz izabrani Katalanov ključ (koji znaju i pošaljioc i primaoc) kriptuju koordinate (x, y) temena, iz niza R, novonastalih trouglova Delone triangulacije i čuvaju se u nizu R_k koji je potreban zbog procesa dekripcije. U ovom trenutku se dobijeni elementi skupa R_k konvertuje u $Base64$ kod. Da bi korisnik mogao da sazna skrivenu informaciju, potrebno mu je poslati originalnu sliku, enkriptovanu Delone triangulaciju i niz R_k kako bi on pristupio procesu dekripcije.

Tabela 4.4: Tabela steganizacije tajne poruke

Originalna slika	Binarni reprezet slike	Tajna poruka	Binarni niz S tajne poruke	Validna Delaunay triangulacija za svaki bit tajne poruke (niz R)	Katalan ključ - objekat	Enkripcija Delaunay triangulacije koristeći Stek permutaciju (niz R_k)	Niz R_k u Base64 kodu
image.jpg	011001000 110000101 110100011 000010011 101001101 001011011 010110000 101100111 011001010 010111101 101010011 100000110 010101100 111001110 110110001 001100001 011100110 110001010 000110100 001001101 011010110 100101100 001001110 011010100 100000101 000001010 000010100 001001100 010010000 010100000 101000001 010000010 100011000 111001010	E	0	teme 1 : X=68,Y=44	2816098$_{10}$= 1010101111 1000011000 10$_2$	teme 1: X = 20, Y= 164	WCA9IDIw LCBZPSAx NjQsICBYI D0gMjMyL CBZPSA1Ni wgIFggPSAy MjEsIFk9ID EyNCwgIFg gPSA0OCwg WT0gMTk5 LCAgWCA9 IDc5LCBZP SAxNTcsIC BYID0gMjA sIFk9IDIzM CwgIFggPS AyMjAsIFk9 IDYyLCAg WCA9IDEw NywgWT0g MTY5LC ....
			1	teme 2: X=58,Y=98		teme 2: X = 232, Y= 56	
			0	teme 3: X=95,Y=118		teme 3: X = 221, Y= 124	
			0	teme 4: X=96,Y=157		teme 4: X = 48, Y= 199	
			0	teme 5: X=151,Y= 79		teme 5: X = 79, Y= 157	
			1	teme 6: X=68,Y=188		teme 6: X = 20, Y= 230	
			0	teme 7: X=94,Y=230		teme 7: X = 220, Y= 62	
			1	teme 8: X=179,Y=43		teme 8: X = 107, Y= 169	
		N	0	teme 9 : X=197,Y=134		teme 9: X = 23, Y= 14	
			1	teme 10 : X=231,Y=118		teme 10: X = 63, Y= 124	
			0	teme 11 : X=195,Y=187		teme 11: X = 27, Y= 235	
			0	teme 12 : X=178,Y=230		teme 12: X = 106, Y= 62	
			1	teme 13 : X=159,Y=261		teme 13: X = 207, Y=261	
			1	teme 14 : X=123,Y=299		teme 14: X = 249, Y=425	
			1	teme 15 : X=87,Y=321		vertex 15: X = 93, Y= 273	
			0	teme 16 : X=150,Y=337		teme 16: X = 78, Y= 337	
		....			.....		

Neophodno je napomenuti da se enkriptovana Delone trinagulacija šalje iz razloga kontrole odnosno prevencije greške u procesu dekripcije. U nastavku je predstavljena tabela 4.4 sa primerom enkripcije tajne poruke "ENCRYPTION". Opisani scenario enkripcije tajne poruke realizovan je putem Algoritama 4.3.1 i 4.3.2.

Proces dekripcije se satoji u tome da i pošaljioc i primaoc poruke (slika, Delone triangulacija, niz R_k) znaju vrednost Delone ključa, jer je on nešto što nikako nebi smelo doći u posed potencijalnog napadača u procesu slanja poruke. Sada kada primaoc (kranji korisnik) ima sve što mu je potrebno, pristupa se procesu dekripcije.

Prvi korak jeste konverzija R_k niza iz **Base64** koda u string. Iz dobijenog stringa se izvlače (x, y) koordinate i formira se celobrojni niz R_k. Pokreće se proces kreiranja Delone triangulacije i dobija se enkriptovana Delone triangulacija. U ovom trenutku se

Algoritam 4.3.1 Delone triangulacija binarnog zapisa slike (p_r,$\overline{p_ip_j}$,$\mathcal{T}$)

Ulaz: Slučajno izabrana slika (konvertivana u binarni zapis) i niz S (biti tajne poruke).

1: Kreiranje inicijalne triangulacije $\mathcal{T}$ koja sadrži $\triangle p_{-1}$, p_{-2},p_{-3}.

2: **for** $r = 1$ to N (dužina niza S) **do** (Unesi p_r u $\mathcal{T}$)

if (p_r==0) or (p_r ==1) (Zavisi koji je bit (0 or 1) od tajne poruke u nizu S)

then

Pronađi $\triangle p_ip_jp_k \in \mathcal{T}$, koji sadrži p_r

Unesi (p_r koordinate (x, y) u niz R.

if p_r u unutrašnjosti $\triangle p_ip_jp_k$

then

LegalizeEdge(p_r, $\overline{p_ip_j}$, $\mathcal{T}$)

LegalizeEdge(p_r, $\overline{p_jp_k}$, $\mathcal{T}$)

LegalizeEdge(p_r, $\overline{p_kp_l}$, $\mathcal{T}$)

else (p_r je na ivici $\triangle p_ip_jp_k \in \mathcal{T}$, kaže se na ivici $\overline{p_lp_j}$)

LegalizeEdge(p_r, $\overline{p_ip_l}$, $\mathcal{T}$)

LegalizeEdge(p_r, $\overline{p_lp_j}$, $\mathcal{T}$)

LegalizeEdge(p_r, $\overline{p_jp_k}$, $\mathcal{T}$)

LegalizeEdge(p_r, $\overline{p_kp_l}$, $\mathcal{T}$)

else

Pozvati funkciju **random()** za pronalazak p_r piksela koji sadrži "0" ili "1"
(Zavisi koji je bit (0 ili 1) tajne poruke u nizu S)

Pozvati funkciju **goto()** i vratiti se na liniji 4.

3: Odbaciti $\triangle p_{-1}$, p_{-2} i p_{-3} kao i sve stranice koje sadrže njegova temena, iz $\mathcal{T}$.

4: **Izlaz**: $\mathcal{D}$ (Triangulisan binarni zapis slike ili podskup trouglova inicijalne triangulacije $\mathcal{T}$).

proverava dobijena triangulacija i ona koja je poslata sa originalnom slikom. Provera se vrši preklapanjem triangulacija jedne preko druge. Ukoliko su iste to nam ukazuje da je primalac poruke na dobrom putu dekripcije odnosno, da niz R_k, u poslatom obliku **Base64** koda, nije oštećen, tj. na njega nije delovao potencijalni napadač. U slučaju da je napadač došao u posed niza R_k i da je na osnovu poslate Delone triangulacije saznao koordinate njenih temena, postavlja se pitanje šta dalje, kako je moguće da sazna sadržaj tajne poruke?

Odgovor na ovo pitanje leži u Katalanovom ključu, i činjenici da je tajna poruka sadržana u binarnom reprezentu originalne slike što u ovom metodu napadač nema osnov da sazna. U nastavku se originalna slika ponovo korvertuje u binarni zapis i pomoću izabranog Katalanovog ključa kreira originalna Delone triangulacija. Drugim rečima se od R_k niza dobije prvobitni niz R sa originalnim koordinatama temena (x, y). Obzirom da se radi o originalnoj slici i originalnom binarnom zapisu, svako teme sadrži po jedan bit originalne poruke.

Naravno, vrednost svakog elementa niza R se smešta u neki novi niz S_k koji je potrebno u sekvencama od 8 bitova ponovo konvertovati u string i na taj način saznati pravu poruku. Nakon detaljnog objašnjenja procesa enkripcije-dekripcije tajne poruke, predstavljeno je kreiranje konkretne Delone triangulacije binarnog reprezenta slike za navedenu tajnu poruku

Algoritam 4.3.2 Enkripcija Delone triangulacije binarnog zapisa slike (p_r,$\overline{p_ip_j}$,$\mathcal{T}$)

Ulaz: Triangulacija $\mathcal{D}$ kao rezultat rada Algoritma 4.3.1 i niz R.

1: **for** $r = 1$ to N (dužina niza S) **do** (Pristup p_r u nizu $\mathcal{R}$)
2: Konverzija (x, y) koordinata p_r u binarni zapis
3: Pozivl **Stack permutation** metode i izbor **Catalan key** ili objekta.
4: Konverzija p_s u decimalni zapis (nakon permutacije, bit p_r postaje p_s)
5: Unesi (x, y) koordinate od p_s niz R_k.
6: Kreiraj inicijalnu triangulaciju $\mathcal{T}_s$ koja sadrži $\triangle p_{-1}$, p_{-2},p_{-3}.
7: **for** $s = 1$ to N (dužina niza S) **do** (Unesi p_s in $\mathcal{T}_s$)
Pronađi $\triangle p_ip_jp_k \in \mathcal{T}_s$, koji sadrži p_s
Pristupi (x, y) koordinatama od p_s u nizu R_k.
if p_s u unutrašnjosti $\triangle p_ip_jp_k$
then
LegalizeEdge(p_s, $\overline{p_ip_j}$, $\mathcal{T}_s$)
LegalizeEdge(p_s, $\overline{p_jp_k}$, $\mathcal{T}_s$)
LegalizeEdge(p_s, $\overline{p_kp_l}$, $\mathcal{T}_s$)
else (p_s je na ivici $\triangle p_ip_jp_k$, kaže se na ivici $\overline{p_ip_j}$)
LegalizeEdge(p_s, $\overline{p_ip_l}$, $\mathcal{T}_s$)
LegalizeEdge(p_s, $\overline{p_lp_j}$, $\mathcal{T}_s$)
LegalizeEdge(p_s, $\overline{p_jp_k}$, $\mathcal{T}_s$)
LegalizeEdge(p_s, $\overline{p_kp_l}$, $\mathcal{T}_s$)
8: Konverzija niza R_k iz decimalnog zapisa u **Base64** kod.
9: Odbaci $\triangle p_{-1}$, p_{-2},p_{-3} kao i sve stranice koje su sadržane u njegovim temenima, iz $\mathcal{T}_s$.
10: Izlaz: $\mathcal{D}_s$. (Enkriptovan, triangulisan binarni zapis slike ili podskup trouglova od skupa $\mathcal{T}_s$).

"ENCRYPTION".

Korak 1. U ovom koraku se tajna poruka konvertuje u binarni zapis tako što se svako slovo konveruje u 8 bita "ENCRYPTION"= 01000101 01001110 01000011 01010010 01011001 01010000 01010100 01001001 01001111 01001110. Svaki bit tajne poruke se unosi u niz S jer je potrebno da se tačno znaju vrednosti bita zbog procesa dekripcije u ovom slučaju je to 80 bita, i to će ujedno biti i dužina niza S.

Korak 2. U ovom koraku se bira slučajna slika i konvertuje u binarni zapis. Ovaj binarni zapis slike se posmatra kao njen binarni reprezent, tako što se polazi od predpostavke da je njegov svaki bit potencijalni nosioc bita tajne poruke.

Korak 3. Pristupa se procesu kreiranja Delone triangulacije tako što se za pozadinu uzme binarni reprezent slike i niz S iz koraka 1. Triangulacija se izvodi primenom inkrementalnog algoritma. Kao što je rečeno svakom bitu iz niza S se pridružuje po jedno teme p_r (Algoritam 4.3.1) Delone triangulacije sa koordinatama (x, y) na primer: $p_1 = ((68, 44) == 0)$, $p_2 = ((58, 98) == 1)$, $p_3 = ((95, 118) == 0)$, $p_4 = ((96, 157) == 0)$,

$p_5 = ((151, 79) == 0)$, $p_6 = ((68, 188) == 1)$, $p_7 = ((94, 230) == 0)$, $p_8 = ((179, 43) == 1)$, $p_9 = ((197, 134) == 0)$, $p_{10} = ((231, 118) == 1)$, $p_{11} = ((195, 187) == 0)$, $p_{12} = ((178, 230) == 0)$, $p_{13} = ((159, 261) == 1)$, $p_{14} = ((123, 299) == 1)$, $p_{15} = ((87, 321) == 1)$, $p_{16} = ((150, 337) == 0)$, $p_{17} = ((296, 63) == 0)$, $p_{18} = ((305, 100) == 1)$, $p_{19} = ((314, 137) == 0)$, $p_{20} = ((311, 173) == 0)$, $p_{21} = ((268, 188) == 0)$, $p_{22} = ((276, 246) == 0)$, $p_{23}((251, 283) == 1)$, $p_{24} = ((286, 323) == 1)$, $p_{25} = ((177, 337) == 0)$, $p_{26} = ((141, 378) == 1)$, $p_{27} = ((178, 394) == 0)$, $p_{28} = ((213, 412) == 1)$, $p_{29} = ((250, 392) == 0)$, $p_{30} = ((357, 376) == 0)$, $p_{31} = ((396, 375) == 1)$, $p_{32} = ((412, 301) == 0)$, $p_{33} = ((96, 390) == 0)$, $p_{34} = ((86, 430) == 1)$, $p_{35} = ((40, 298) == 0)$, $p_{36} = ((14, 428) == 1)$, $p_{37} = ((349, 174) == 1)$, $p_{38} = ((412, 120) == 0)$, $p_{39} = ((386, 190) == 0)$, $p_{40} = ((452, 117) == 1)$, $p_{41} = ((514, 247) == 0)$, $p_{42} = ((504, 192) == 1)$, $p_{43} = ((450, 265) == 0)$, $p_{40} = ((442, 374) == 1)$, $p_{45} = ((468, 375) == 0)$, $p_{46} = ((403, 415) == 0)$, $p_{47} = ((359, 450) == 0)$, $p_{48} = ((276, 468) == 0)$, $p_{49} = ((532, 41) == 0)$, $p_{50} = ((460, 78) == 1)$, $p_{51} = ((459, 153) == 0)$, $p_{52} = ((378, 302) == 1)$, $p_{53} = ((604, 134) == 0)$, $p_{54} = ((467, 41) == 1)$, $p_{55} = ((542, 341) == 0)$, $p_{56} = ((314, 245) == 0)$, $p_{57} = ((604, 192) == 0)$, $p_{58} = ((449, 176) == 1)$, $p_{59} = ((352, 285) == 0)$, $p_{60} = ((260, 337) == 0)$, $p_{61} = ((342, 432) == 1)$, $p_{62} = ((169, 413) == 0)$, $p_{63} = ((87, 465) == 0)$, $p_{64} = ((425, 470) == 1)$, $p_{65} = ((658, 43) == 0)$, $p_{66} = ((562, 63) == 1)$, $p_{67} = ((590, 97) == 0)$, $p_{68} = ((578, 211) == 0)$, $p_{69} = ((579, 395) == 1)$, $p_{70} = ((432, 249) == 1)$, $p_{71} = ((331, 283) == 1)$, $p_{72} = ((461, 448) == 1)$, $p_{73} = ((678, 207) == 0)$, $p_{74} = ((596, 359) == 1)$, $p_{75} = ((635, 431) == 0)$, $p_{76} = ((611, 466) == 0)$, $p_{77} = ((551, 483) == 1)$, $p_{78} = ((542, 284) == 1)$, $p_{79} = ((581, 134) == 1)$, $p_{80} = ((713, 393) == 0)$.

Ova temena se unose u niz R koji nam je potreban radi enkripcije binarnog reprezenta slike i ona su predstavljena u Tabeli 4.4.

Korak 4. U ovom, koraku definisana temena Delone triangulacija se konvertuju u binarni zapis i kritputju primenom **stek permutacje** i izabranog Kalatanovog ključa (u ovom slučaju je to $2816098_{10} = 10101011111000011000010_2$). Pravilo je da obe strane i pošaljioc i primaoc poruke znaju izabrani Katalanov ključ. Enkriptovana temena su redom: $p_{s1} = (20, 164)$, $p_{s2} = (232, 56)$, $p_{s3} = (221, 124)$, $p_{s4} = (48, 199)$, $p_{s5} = (79, 157)$, $p_{s6} = (20, 230)$, $p_{s7} = (220, 62)$, $p_{s8} = (107, 169)$, $p_{s9} = (23, 14)$, $p_{s10} = (63, 124)$, $p_{s11} = (27, 235)$, $p_{s12} = (106, 62)$, $p_{s13} = (207, 261)$, $p_{s14} = (249, 425)$, $p_{s15} = (93, 273)$, $p_{s16} = (78, 337)$, $p_{s17} = (416, 237)$, $p_{s18} = (353, 52)$, $p_{s19} = (488, 131)$, $p_{s20} = (365, 167)$, $p_{s21} = (388, 230)$, $p_{s22} = (324, 126)$, $p_{s23} = (251, 457)$, $p_{s24} = (460, 281)$, $p_{s25} = (99, 337)$, $p_{s26} = (135, 504)$, $p_{s27} = (106, 394)$, $p_{s28} = (87, 454)$, $p_{s29} = (250, 386)$, $p_{s30} = (309, 496)$, $p_{s31} = (390, 381)$, $p_{s32} = (454, 421)$, $p_{s33} = (48, 270)$, $p_{s34} = (92, 430)$, $p_{s35} = (160, 424)$, $p_{s36} = (140, 422)$, $p_{s37} = (469, 174)$, $p_{s38} = (454, 240)$, $p_{s39} = (266, 238)$, $p_{s40} = (278, 117)$, $p_{s41} = (520, 127)$, $p_{s42} = (498, 18)$, $p_{s43} = (282, 385)$, $p_{s44} = (490, 380)$, $p_{s45} = (342, 381)$, $p_{s46} = (331, 463)$, $p_{s47} = (317, 282)$, $p_{s48} = (324, 342)$, $p_{s49} = (580, 161)$, $p_{s50} = (406, 156)$, $p_{s51} = (411, 195)$, $p_{s52} = (504, 428)$, $p_{s53} = (724, 14)$, $p_{s54} = (347, 161)$, $p_{s55} = (716, 341)$, $p_{s56} = (488, 119)$, $p_{s57} = (724, 18)$, $p_{s58} = (275, 98)$, $p_{s59} = (304, 453)$, $p_{s60} = (260, 337)$, $p_{s61} = (348, 354)$,

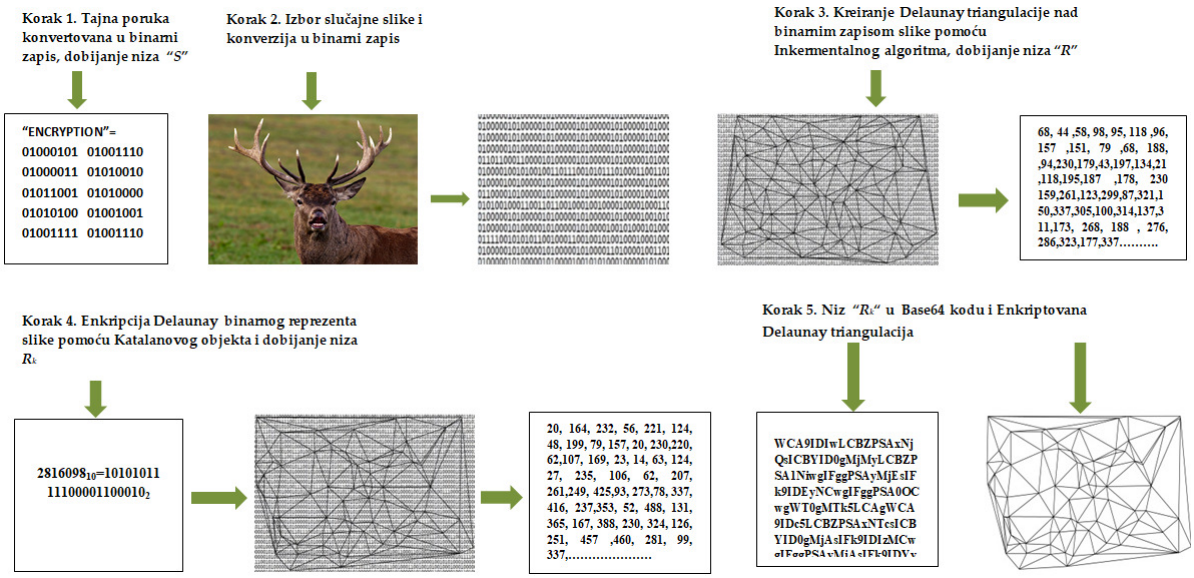
$p_{s62} = (163, 455)$, $p_{s63} = (93, 339)$, $p_{s64} = (419, 350)$, $p_{s65} = (586, 169)$, $p_{s66} = (616, 237)$, $p_{s67} = (668, 49)$, $p_{s68} = (536, 91)$, $p_{s69} = (537, 395)$, $p_{s70} = (354, 243)$, $p_{s71} = (409, 457)$, $p_{s72} = (407, 274)$, $p_{s73} = (558, 159)$, $p_{s74} = (596, 317)$, $p_{s75} = (761, 431)$, $p_{s76} = (569, 346)$, $p_{s77} = (557, 315)$, $p_{s78} = (716, 452)$, $p_{s79} = (533, 14)$, $p_{s80} = (659, 387)$. Sva navedena temana se smeštaju u niz R_k.

Korak 5 U ovom koraku se niz R_k konvertuje u **Base64** code, koji za navedeni primer izgleda ovako:
/9j/4AAQSkZJRgABAQEAlgCWAAD 4gHbSUNDX1BST0ZJTEUAAQEAA
AHLAAAAAAJAAABtbnRyUkdCIFhZWiAAAAAAAAAAAAAAAABhY3NwA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAQAA9tYAAQAAAAD
TLVF0BQ8AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAlyWFlaAAAA8AAAABRnWFlaAAABBAAAAB
RiWFlaAAABGAAAABR3dHB0AAABLAAAABRjcHJ0AAABQAAAAAxyVFJ
DAAABTAAAACBnVFJDAAABTAAAACBiVFJDAAABTAAAACBkZXNjAAA
BbAAAAF9YWVogAAAAAAAAAb58AADj0AAADkVhZWiAAAAAAAAAABilgAA
t4cAABjcWFlaIAAAAAAAAACShAAAPhQAAttNYWVogAAAAAAAAA808AAQ
AAAAEWwnRleHQAAAAATi9BAHBhcmEAAAAAAAAMAAAACZmYAAPKnA
AANWQAAE9AAAApbZGVzYwAAAAAAAAAAAFc1JHQgAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAD2wBDAAgGBg
cGBQgHBwcJCQgKDBQNDAsLDBkSEw8UHRofHh0aHBwgJC4nICIsIxw
cKDcpLDAxNDQ0Hyc5PTgyPC4zNDL2wBDAQkJCQwLDBgNDRgyIRwh
MjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIy
MjLwAARCADyAPIDASIAAhEBAxEB8QAHwAAAQUBAQEBAQEAAAAAAA
AAAAAECAwQFBgcICQoL8QAtRAAAgEDAwIEAwUFBAQAAAF9AQIDAA
QRBRIhMUEGE1FhByJxFDKBkaEII0KxwRVS0fAkM2JyggkKFhcYGRolJic
oKSo0NTY3ODk6Q0RFRkdISUpTVFVWV1hZWmNkZWZnaGlqc3R1dnd
4eXqDhIWGh4iJipKTlJWWl5iZmqKjpKWmp6ipqrKztLW2t7i5usLDxMX
Gx8jJytLT1NXW19jZ2uHi4+Tl5ufo6erx8vP09fb3+Pn6/8QAHwEAAwE
BAQEBAQEBAQAAAAAAAAECAwQFBgcICQoL8QAtREAAgECBAQDBAc
FBAQAAQJ3AAECAxEEBSExBhJBUQdhcRMiMoEIFEKRobHBCSMzUvAV
YnLRChYkNOEl8RcYGRomJygpKjU2Nzg5OkNERUZHSElKU1RVVldYWV
pjZGVmZ2hpanN0dXZ3eHl6goOEhYaHiImKkpOUlZaXmJmaoqOkpaa
nqKmqsrO0tba3uLm6wsPExcbHyMnK0tPU1dbX2Nna4uPk5ebn6Onq8
vP09fb3+Pn6/9oADAMBAAIRAxEAPwDOFwM5zTzc7sZrOZWB709Aa8
rlRukaIkGKUyCqRdgMYpGkO2o5bhZl4EGpAFKnPWsxZWC09bk9DQ4D
bsXOOlTWtnPeO628bSFEaRsfwqBkk1TSXPWut0cxWHg3WL5mAmuF
WBF77ScH8+fyoVND6EiiOae0B281
KTt6ihpRtxQ2wuktSuYzjg0+KFiKejqTyKspIg4FO7Gop6kAgYt0qwluoH
Ip5detMMvoalt9R8PbDtRvPrUD
zvu5PFDYOCRYW14phtmU0+Ofjmtm0giufD97P8AL50EsZB7lCP6hV

JcwcljnniYUnlsBVcp61GzoDU2JsZ53hqkEhAq0UVhnNMMINBOpD5xI5p
N49ac0eKZszRcrUVZecZpfMyetRGM5pwj5p6E3ZLvopu0UUXK1IHQelM
8sA1I3TNMZgKepftExWhz0qS10ye+l8m3j3yBWfaCBwoJP6Co1kB4zzX
ReCJVHiu33Y27JAc+mw1S1aQnJHNG2JWoZLdk5HWta52wXMsQ6RuV
I4qu0iHqKi8hNGTulVwBkn0rrvEFz9m0SysAoVlC7gDycDnP4tWfpFpHe
atCrLlEbe+P7o61Hqc4vNYvJCxZd+1QT7DNap2jctfA2ZkbGp4mO7NW
YrJpoJpY0ykIDSNnhQSAP1NQMoXOKjmMrtD2kLGgAmoQTk09GIOc0h
81wbcppPOINJJIc1GoLHrVIOboWvOyOtM85s9acseRTxApGRijQa1Qgk
NPDE81XPDYqeMjoaJRsHKTK47imPtY9KCwHAprMAOaixdhQcVteH71
YmuLaTdtnUAgHrjPH45rB38cU6CXZOrHse9OOjuOLVy1OpgmeNjkqx
Gary4NWtaJNwsykESgEkevv71QViRzTlHldgejsaelxxTSvHKu7MTY55D
Y4P51VBJ71FbXUlrOZI1ydrdswmlkcJO4H3Qxx9M0nH3SXa1yWtLiykh
E2B5sSzJgwALDIqBScV0HitRv0piQQdPiAIGOASKwFZe1Kas7Cshc+1Jy
TwKeNp709SoFSmKxHsPpRUu9aKQ7GZvL8CkPTFW4IVA5oliXOcVq5K
4ZFEKc8VteHWMWqBx18qQf8AjprNCDPSrti6xXUbMdqn5WPoCMH+d
F9QjTSkmRX+46hcllIJkZsHtk5qizYrYvog9yz4ILE5HPGCRSqptlbsKJbil
Bplnw6228kOSCyhMj3IrImQJNKEJwZGPJ55Jro9HijglkcrubZwAeeoJx7
8Vl/YDdeIUtcgJJNtJ7Bc8n8qu14jlf2aRp3sR0XwhaW7gi81Jxcyc9IVHy
A/UknwDVXObmb6Vo6rqcmu6xc3BYeUj+VEoPCovQCoha7F5GaU7Xt
2JcGUd5XtThOO4q39mUg8VUntsHikkmDgIzqTSbgDwaYsLVYFqSucU
WJsJ5+B1pVuSetRvbsOlAiI60WsOxMGB6GnB8GofLbHAoVW3c09xak
7Pg81G756UkikrUaMBwaLodyVJADzStKM+9Qsueav2+kfatFmv7eYtN
bviaEjohxhl9e+aLX2JWmxav8PZW7ggfLjAzkdv1qbUdNW00Tr6Ld5F1
DyT2kH3qbJCT4fiYhPm9F9B6X+ldBHEupCRCuDPZO0gyewdswDjrVo
48y+RvU0dzkYmCueuSpUEDpnj+tNhjeu4Yk4eeQKMnoWOKgJaHSrq
eQIkK4AxyxOcfqv61teDZLK2SDVrqYExRbkjKk4fGMtj9PelGm2hv4UaH
jllTWobZMAQWyx4BzjljACxXLFiKvapfNqWq3F0SSJHJBPp2qmy5Hpk7y
0MrDd7DvViIkjrUPlkDNKjFWxzUNFLQsbTRSeZ7UVJd0IVePAprs2BUs9
wjN0qGYqU+Umi2pOpNaJE1xEbjcIC48wp1C55x+Fa+uaFaw6RcXmn
34uISpCpt+cEjufbrmsCCUgEZps6JKGG4qSMHuD9R0P86q1nqVF2epd
02Y6pplvI2Q4QI3fngdOuev61G6SwvhwQMnB7H6Vm20j2N7yRHGxG
MY8tvb2P1DNdbp+rkzIsf7o+vZiSc5z1GqceY1cVLVCaMhV0L7syMCqk
4BUZz+Bz+lRXW2yu9SvZchobV3TacfM3y//FV0qvvFw8QRtpAHHRh1/
nXL+NZYrfRZXUbZpisbLnOQGP8AjWsIZJa0Oc0eIxWKls5Zi1XnkLDior
Ij7HGMYBXIHpnmrK7ccVhOfvszKwdh1p4IJ5p20FvSnPFgcGjmQWIpC
ueBUnmu8DrW14RtI7rVLqKfbtaylAz6kAcfnXObWzV20uKybsP8z1FN3
qTxSGM4zUTKVOaaSYmuUtxhScGpPLWqkTnHSrKlnXntUvQEiKbABq
s...

Korak 6. Nakon završetka procesa enkripcije niz R_k, originalna slika, i enkriptovana Delone trangulacija binarnog reprezenta slike se šalju preko nekog medijuma (interneta) primaocu. Da bi otkrio pravu poruku primalac treba da pristupi procesu dekripcije koji je već opisan.

U nastavku pogledajte Sliku 4.19 koja objašnjava prethodno opisane korake.



Slika 4.19: Realna Delone triangulacija binarnog reprezenta slike i kreiranje šifrata

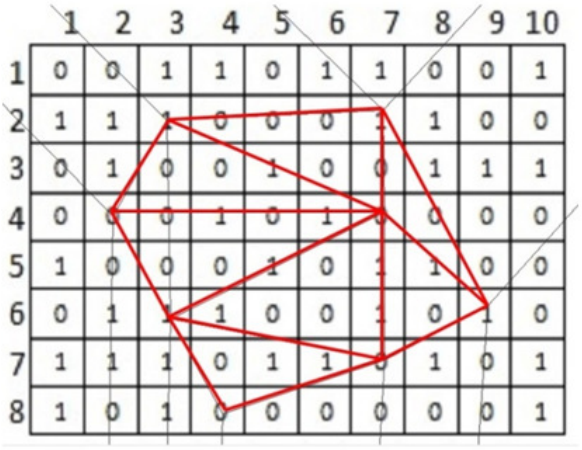
4.3.4 Složenost algoritma i maksimalna dužina tajne poruke u karakterima

Ako se posmatra složenost Algoritma 4.3.1, može se primetiti da broj trouglova kreiranih Delone algoritmom triangulacije iznosi najviše *9n+1*, a vremenska složenost ovog algoritma je *O(n log n)* koristeći *O(n)* memorijske lokacije, kao što je dokazano u [6] (Lema 9.11, Teorema 9.12). Ovde je Algoritam 4.3.1 takođe Delone algoritam triangulacije, sa dodanom kontrolom bitova koji nose tajnu poruku. Međutim, ne može povećati broj povrataka u petlji počevši od trećeg reda, zadržavajući istu proporcionalnost vremenskog trajanja s obzirom na broj bitova n. Kada se govori o maksimalnoj dužini tajne poruke koja se može ugraditi u sliku date rezolicije (npr. 800 × 600), pre svega mora se poštovati odnos koji će se proveriti pre postupka ugrađivanja poruke u sliku, a to se izražava kao:

$$B_{0s} >= B_{0p} \; and \; B_{1s} >= B_{1p} \tag{4.2}$$

gde je B_{0s} ukupan broj 0 bitova na slici, B_{1s} je broj 1 bitova na slici, B_{0p} broji 0 bitova u tajnoj poruci, a B_{1p} je broj 1 bita u tajnoj poruci. Treba napomenuti da se ova metoda ne temelji na uklanjanju bitova sa slike. Osnovna ideja je sakriti tajnu poruku bez mijenjanja izgleda slike, čak i u jednom pikselu. Na osnovu prethodno izloženog razmotriće se slučaj smanjenja BPP-a (bit po pikselu) sa 1/20 na manji odnos, poput 1/10. Ako se analiziraju podaci koji proizilaze iz slike određene rezolucije (800 × 600), dužina tajne poruke bila bi (480000/10) / 8 = 6000, što je dvostruka dužina tajne poruke koja se može postaviti na

slici, nego BPP koji se koristi u LSB tehnici. To je BPP od 1/20. Nije narušen izvorni izgled slike niti u jednom pikselu. Kao potvrda da je ovo tvrđenje istinito, kreirana je jedna Delone triangulacija (Slika 4.20.) u mreži od *10×8* piksela gde se poštuje odnos 1/10. To znači da je jedan bit skrivene poruke smešten u svaku podmrežu od *5×2* piksela.



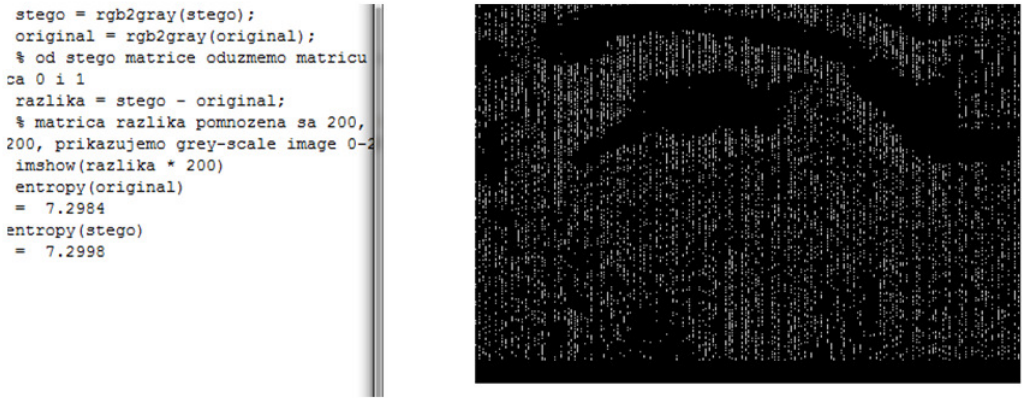
Slika 4.20: Naš BPP raspored = 0,1 ili 1/10

4.3.5 Steganaliza i sigurnosna ispitivanja

Jedna od najčešćih tehnika koja se danas koristi u Steganografiji jeste LSB tehnika. Ona je karakteristična po tome što se najmanje značajni bitovi slike menjaju, tako da čine ugrađenu informaciju. Sledeći primer pokazuje kako se slovo A može sakriti u prvih osam bajtova od tri piksela na 24-bitnoj slici. Biti slike su: 00100111 11101001 11001000 00100111 11001000 11101001 11001000 00100111 11101001. Slovo A je 01000001. Rezultat umetanja je 00100110 11101001 11001000 00100110 11001000 11101000 11001000 00100111 11101001.

Kao što se može primetiti sa ovog primera, umetanje LSB-a u proseku zahteva promenu samo polovine bitova na slici. Budući da 8-bitnom slovu A treba samo osam bajtova da ga sakrije, deveti bajt od tri piksela može se upotrebiti za početak skrivanja sledećeg znaka skrivene poruke [66]. Ovaj algoritam zamenjuje bit najmanje težine svakog piksela sa bitovima poruke koja se želi sakriti. Obzirom da se menja samo LSB piksel, on je ljudima vizualno nevidljiv [48]. Kako bi se procenila sigurnost predstavljenog rešenja, urađene su sledeće analize: entropija približnih vrednosti (izvorna u odnosu na stego sliku), kao i raspodela bitova u stego slici (crna ili RGB (Read Green Blue)). Realizacija testa steganizacije provedena je u nekoliko koraka. U prvom koraku odabrano je 150 slika (JPG format) kreiranih na istom uređaju sa istom rezolucijom. Zatim su za analizu uzeta 24 bita slike, i konačno je određena količina informacija po pikselu koja bi se sakrila. Kao osnova za realizaciju ovog testa korišćene su preporuke prethodnog istraživanja stego analize [35]. U tim studijama je dokazano da niti jedan postojeći alat ne može otkriti LSB algoritam gde

su informacije 0,005 BPP. Ispitivanjem su utvrđeni dopušteni parametri LSB algoritma za siguran steganografski kanal. Predstavljena je uporedna analiza koja je pokazala da klasifikator ima slične ili bolje performanse od postojećeg algoritma. Ako se pogledaju rezultati, uočiće se da je entropija približna odnosno da je za obe slike ostala ista, pa je to poželjno svojstvo metode. Slika 4.21 prikazuje rezultate ispitivanja učestalosti pojavljivanja svih mogućih preklapanja n-bitnih obrazaca unutar serije. Uz to, može se videti da je entropija (neodređenost) stego slike bila u stabilnoj ravni ili veća od izvorne slike.

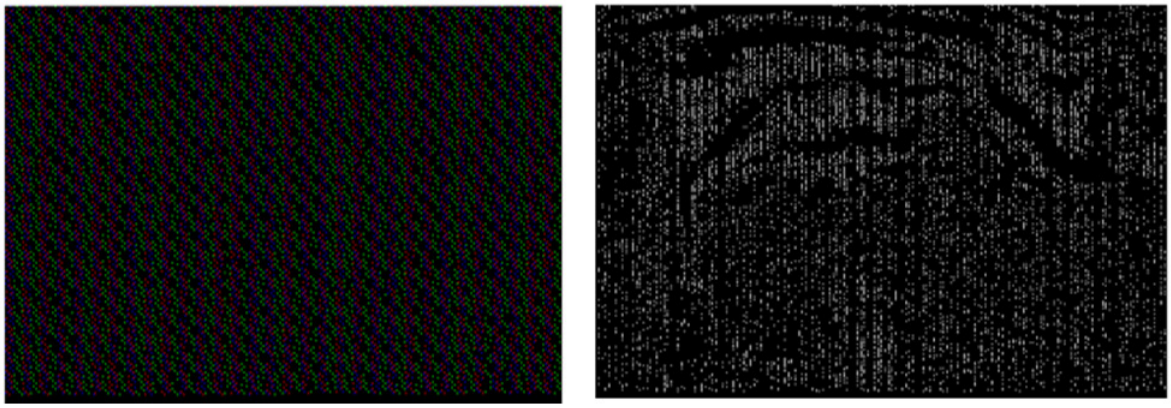


Slika 4.21: Uporedna analiza entropije izvorne i stego slike

Da bi se dobila jednolična distribucija bitova, primenjena je permutacija bita poruke pre ugrađivanja. Slika 4.21 prikazuje distribuciju bitova za jednu odabranu sliku, u kojoj su tajni podaci ugrađeni Katalanovom stego metodom. Prikazuju se samo ugrađeni bitovi sa vrednošću "1", dok su biti sa vrednošću "0" izostavljeni. U pogledu ovog kriterijuma, koji se odnosi na jednoličnu raspodelu skrivenih bitova, važno je naglasiti da je u ovoj metodi definisana Delone triangulacija, koja je uvek slučajna. Međutim, kod LSB tehnike uvek postoji unapred određeni niz bitova koji se postavljaju u najmanje značajne (važne) položaje ili bitove. Na taj način se postiže puno bolja entropija u testiranju. Na Slici 4.22 je data predstavljena distribucija bitova.

U procesu određivanja količine podataka koje treba sakriti, korišćene su smernice iz prethodnih istraživanja LSB steganalize, gdje je utvrđeno da niti jedan alat ne može otkriti LSB algoritam u kojem su podaci 0,005 BPP (bita po pikselu). Ugradnjom tajnih podataka u sliku, testirane su neke dodatne mogućnosti. Jedan od njih prikazivao je dodatne parametre koji su ukazivali da li je postupak uspešno proveden s obzirom na količinu informacija instaliranih po pikselu. Ova tehnika ne menja sliku (nosača), ali u kombinaciji s određenim ključem omogućuje efikasno izdvajanje poruka. Zato ova tehnika spada u kategoriju steganografskih metoda jer omogućuje skrivanje i automatsko generisanje tajnih podataka, ali uz postojanje složenog stego ključa. Tabela 4.5. prikazuje uporednu analizu brzine (u sekundama) izdvajanja tajne poruke za sledeće tri steganografske metode:

1. **DTM**-Delone metoda triangulacije (naša predložena metoda)



Slika 4.22: Distribucija bitova na slici (crna ili RGB)

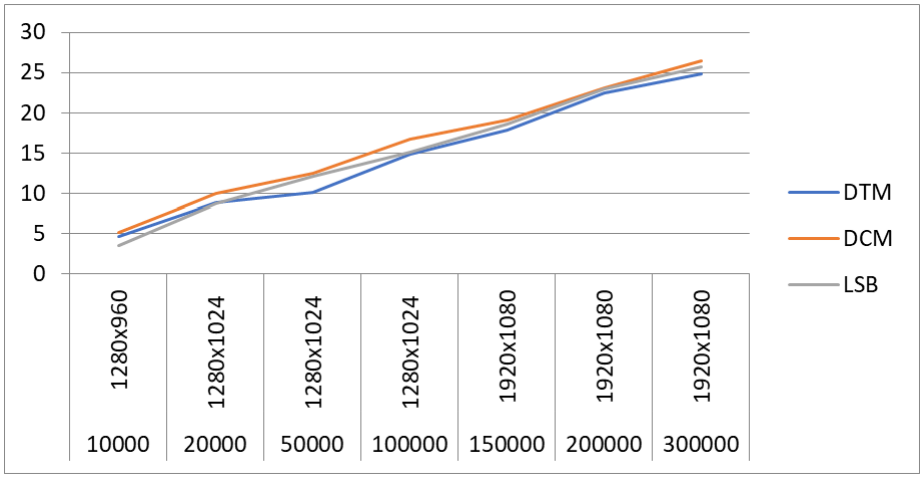
2. **DCM**-Katalanova metoda razlaganja (predstavljena u [35])

3. **LSB** - metoda najmanje bitnih bitova (tehnika predstavljena u [48]).

Tabela 4.5: Uporedna analiza brzine (u sekundama) izdvajanja tajne poruke.

T	**Dužina tajne poruke (u bitima)**	**Veličina slike (Rezolucija)**	**DTM**	**DCM [35]**	**LSB[48]**
1	500	800 x 600	0.36	0.33	0.27
2	1000	800 x 600	0.47	0.41	0.38
3	1500	800 x 600	0.89	0.79	0.62
4	3000	1280 x 960	1.12	1.15	0.94
5	5000	1280 x 960	2.51	2.69	1.74
6	10.000	1280 x 960	4.68	5.17	3.51
7	20.000	1280 x 1024	8.87	9.97	8.74
8	50.000	1280 x 1024	10.17	12.47	12.17
9	100.000	1280 x 1024	14.89	16.74	15.14
10	150.000	1920 x 1080	17.85	19.11	18.55
11	200.000	1920 x 1080	22.51	23.08	22.99
12	300.000	1920 x 1080	24.78	26.44	25.73

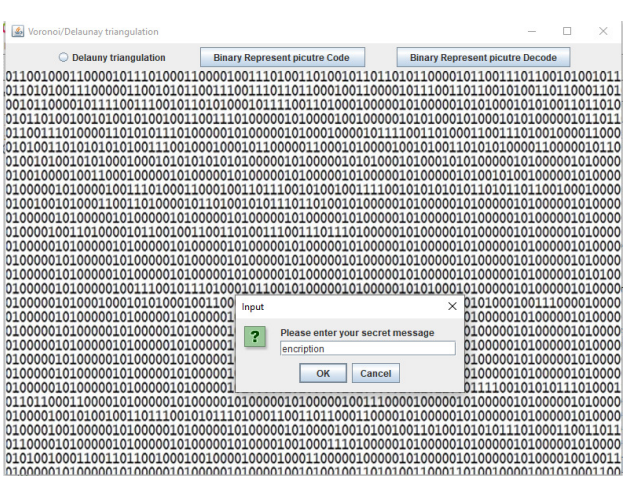
Kao što se može videti, ova predložena tehnika ima prednosti u povećanju rezolucije slike i povećanja dužine tajne poruke. Primjećuje se da je ova tehnika provela najmanje vremena postižući najviše vrednosti (dužine tajne poruke) u testiranju. To se može videti na sledećem grafiku (vidi Sliku 4.23). Objavljeni rad autora vezano za ovu predloženu metodu je [65].



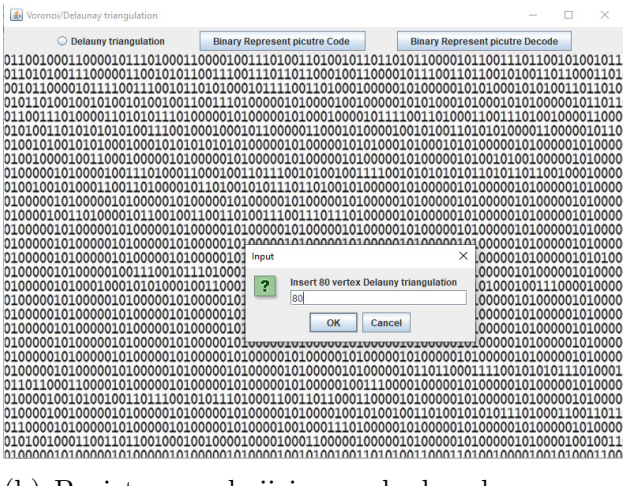
Slika 4.23: Uporedna analiza tri tehnike pri najvećim vrednostima u ispitivanju

4.3.6 Implementacija predloženog metoda u Java-Net Beans okruženju.

Implementacija predložene metode u Java-NetBeans okruženu zasniva se shodno koracima u opisu metoda. Klase koje su korišćene za realizciju ove metode su navedene u sekciji 3.2.3. U okviru navedenih klasa, dodate su još neke konkretne programske metode za realizaciju ove steganografske metode. U nastavku su predstavljeni neki segmenti aplikacije, dok pojedini segmenti koda od klasa i metoda mogu se pogledati u prilozima. U prvom koraku bira se slučajna slika koja se konvertuje u Base64 kod a zatim u Binarni zapis. Nakon toga se formira 2D niz veličine rezolucije slike (npr: 800 x 600) u koji se smešta binarni zapis slike. Nakon toga se unosi tajna poruka kao na Slici 4.24.



(a) Unos tajne poruke

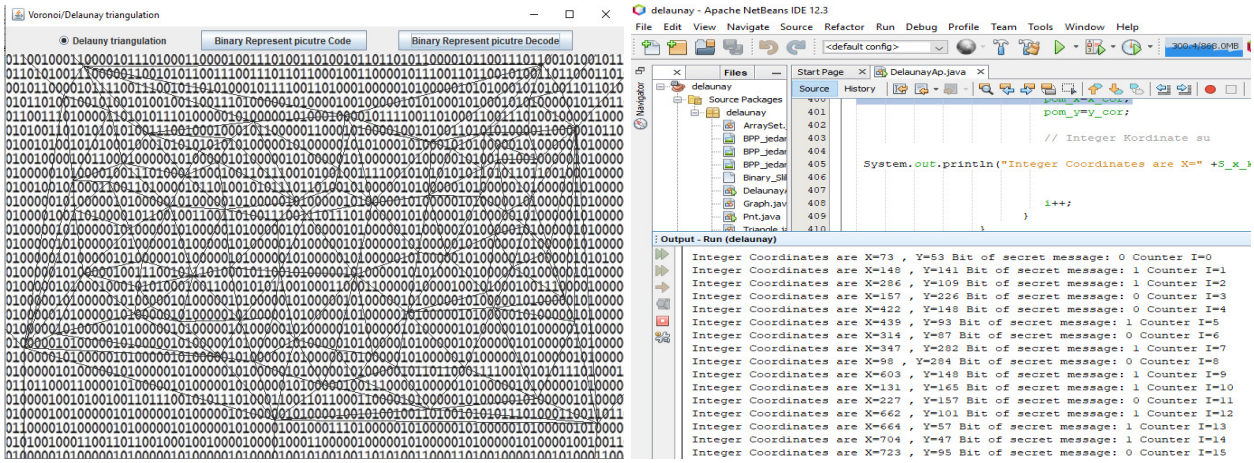


(b) Broj temena koji je neophodno da se unese shodno dužini poruke

Slika 4.24: Primer unosa tajne poruke sa brojem temena

Nakon unosa tajne poruke proverava se njena dužina i konvertuje se u binarni zapis. Drugim rečima, definiše se broj temena Delone triangulacije koji je neophodan da bi se tajna poruka implementirala. Tako na primer za unetu reč "encription" sa Slike 4.24 potrebno je da se unese 80 temena jer zadata reč ima 10 karaktera. Na ovoj slici se jasno može videti navedeni broj temena koji je neophodno uneti.

U ovom trenutku se definisanom 2D nizu prodružuju biti slike u vidu koordinata (x, y). Proverava se odnos bita "0" i "1" u odnosu na bitove tajne poruke. Ako je uslov zadovoljen onda se ide dalje. Drugim rečima, traži se "validna Delone triangulacija" kao na Slici 4.25. Takođe je predstavljeno i pridruživanje temena bitova tajne poruke temenima Delone tringulacije odnosno njihovim koordinatama (x, y).



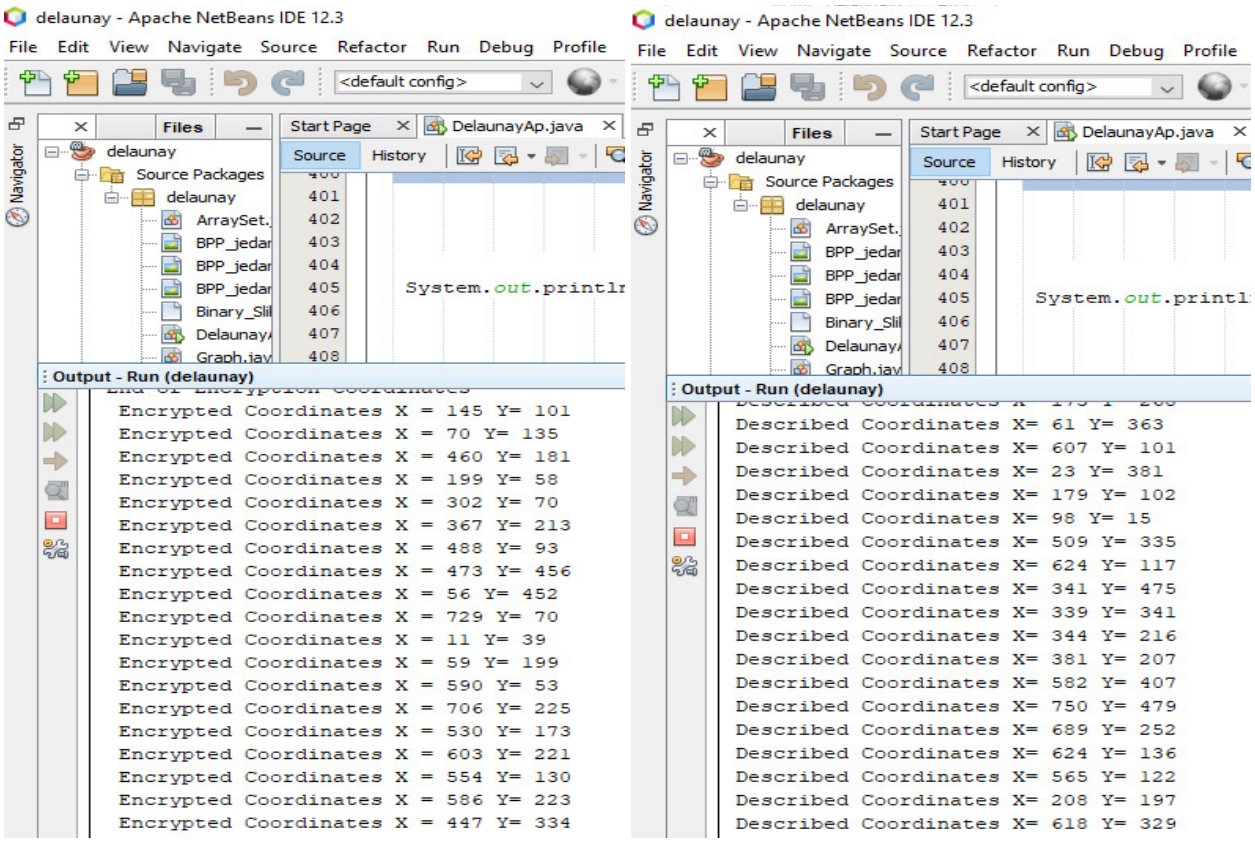
(a) Validna Delone triangulacija (b) Temena sa pridruženim bitima tajne poruke

Slika 4.25: Primer validne Delone triangulacije binarnog zapisa slike

Ovako definisana Delone triangulacija se kriptuje sa Katalanovim objektom (sa nizom temena tj. njihovih koordinata (x, y)) i ona je u stvari jedan od glavnih elemenata u stego ključu. Zatim se, pored ostalnih elemenata stego ključa niz sa koordinatama šalje primaocu poruke koji treba da od njega ponovo kreira Delone triangulaciju i konvertuje je u originalnu. Na Slici 4.26 su predstavljene originalne i kriptovane koordinate (x, y) temena Delanuay, odnosno niz u koji su one smeštene.

4.4 Metoda generisanja kriptografskih ključeva sa algoritmom triangulacije poligona i primenom Katalanovih objekata

Vizuelna kriptografija je specijalna tehnika šifrovanja koja omogućava skrivanje informacija (tajnih poruka) u slici na takav način da je čovek može dešifrovati jedino ako poseduje i koristi ispravan ključ. Skrivanje informacija je savremeni način komunikacije u kome se



(a) Originalne koordinate Delone triangulacija (b) Kriptovane koordinate Delone triangulacija

Slika 4.26: Primer kriptovanih temena Delone triangulacije binarnog zapisa slike

uspešno izbegavaju napadi i dešifrovanje poverljivih informacija. Ova metoda prikazuje primenu jednog algoritma računarske geometrije u postupku generisanja kriptografskih ključeva iz jednog segmenta 3D slike.

4.4.1 Neka prethodna istraživanja na ovu temu

Od samog početka računarska geometrija je povezivala različite oblasti nauke i tehnike kao što su teorija algoritama, kombinatorna i Euklidova geometrija, ali je uključivala i strukture podataka, optimizaciju itd. Danas, računarska geometrija ima veliku primenu u računarskoj grafici, vizuelizaciji, GIS-u, CAD programima itd. U pozadini nekog prikaza (slike, animacije itd.) kriju se složeni proračuni i teorije iz oblasti geometrije koje su već odavno potvrđene i razvijene, ali koje sa aspekta primene u savremenim informacionim tehnologijama su još na početku. Jedan od važnijih problema računarske geometrije, koji se primenjuje u računarskoj nauci, je problem triangulacije poligona. Ovaj problem se primenjuje u postupku dobijanja trodimenzionalnih prikaza objekata iz skupa tačaka.

Predloženom metodom generisanja kriptografskih ključeva se pokušava utvrditi koliki značaj imaju triangulacije poligona i Katalanovi objekti u kriptografiji, prvenstveno u

razvoju algoritama za generisanje pseudoslučajnih brojeva koji su neophodni za generisanje ključeva. U radovima [51, 52] mogu se videti konkretne primene u rešavanju nekih kombinatornih problema u računarskoj geometriji. U doktorskoj disertaciji [53] predstavljene su metode i tehnike u rašavanju nekih problema u oblasti računarske geometrije, na osnovu Katalanovih brojeva i kombinatornih problema koji se u drugom radu [54] primenjuju i u oblasti kriptografije. Znači, u ovoj metodi je napravljen spoj računarske geometrije, kriptografije i teorije brojeva. Generalno, teorija brojeva kod asimetričnih sistema ima važno mesto ne samo u generisanju ključeva već i u dizajnu samog kriptografskog algoritma, ali i u kriptoanalizi [54, 55, 56].

4.4.2 Povezanost metoda triangulacije konveksnih poligona i Katalanovih objekata

Triangulacija poligona je istorijski veoma star problem koji je doveo do otkrića Katalanovih brojeva [2]. Kod triangulacije se zapravo razmatra broj načina, koji čemo označiti sa T_n, gde je moguća maksimalna dekompozicija konveksnog n-tougla na $n-2$ trougla, pa otuda potiče ime triangulacija. Triangulacija n-tougaonog poligona zahteva podelu na trouglove sa $(n-3)$ unutrašnje dijagonale koje se ne ukrštaju. Za konveksne poligone sve dijagonale su unutrašnje dijagonale. U ovom slučaju broj triangulacija konveksnog n-tougaonog poligona je nezavistan od oblika i može biti jedinstveno okarakterisan brojem temena n.

U radovima [58, 53], predstavljeni su neki načini rešavanja ovog tipa problema. U nastavku je data veza između pojma triangulacije poligona i Katalanovih brojeva odnosno njihovih objekata. Ako sa T_n označimo broj triangulacija n-ugla, onda važi sledeća relacija:

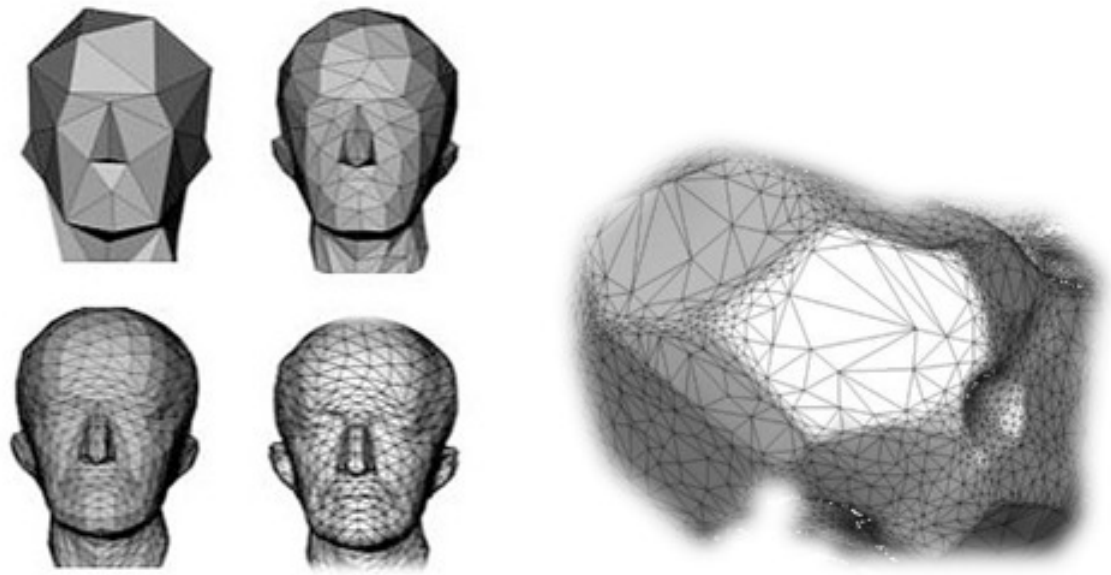
$$T_n = C_{n-2}, n \geq 0 \tag{4.3}$$

gde je n broj temena poligona. Na osnovu 4.3 možemo T_n izraziti kao:

$$T_n = \frac{1}{n-1}\begin{pmatrix} 2n-4 \\ n-2 \end{pmatrix} = \frac{(2n-4)!}{(n-1)!(n-2)!} \tag{4.4}$$

Triangulacija omogućava da se iz skupa tačaka dobije prikaz trodimenzionalnih objekata i omogućava mehanizam za tzv. glačanje trodimenzionalnih figura Slika 4.27. Triangulacija konveksnih poligona je aktuelan problem koji se javlja u dvodimenzionalnoj računarskoj geometriji.

Tehnika triangulacije u računarskoj geometriji je najčešća metoda panelizacije. Najlakši način segmentacije i ravnanja dvostruko zakrivljene površi je putem mreže trouglova. Prednost trougla kao geometrijskog tela je što je površina između tri tačke uvek ravna. U našem postupku primenićemo postupak triangulacija poligona koja ima široku primenu pri modelovanju 3D objekata. Prednosti ove metode su: mala odstupanja od izvornog oblika, dobra strukturalna svojstva i mogućnost oblaganja složenih slobodnih formi. Na osnovu



Slika 4.27: Postupak glačanja trodimenzionalnih figura

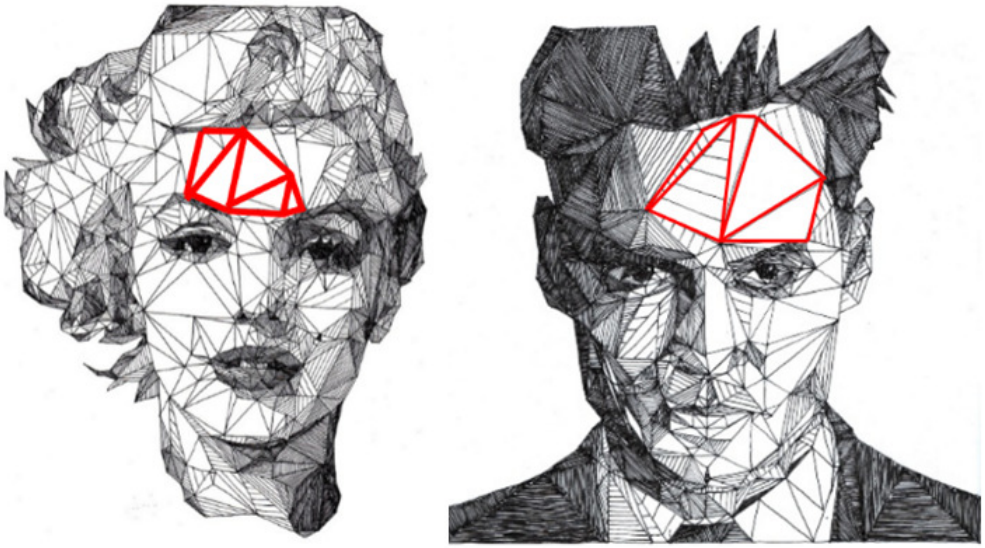
ove metode, u jednoj 3D slici izdvojen je jedan segment, koji će poslužiti kao materijal za generisanje Katalanovog ključa.

4.4.3 Ekstrakcija ključa iz jednog segmenta slike

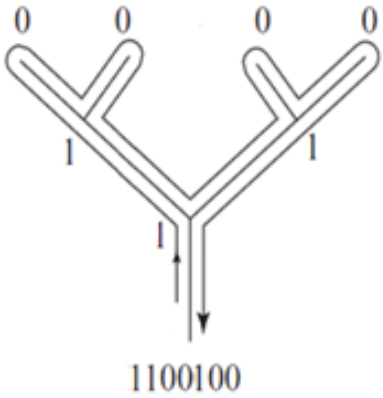
U ovoj sekciji je objašnjen način rada metode generisanja kriptografskog ključa na osnovu triangulacije. Ovaj postupak se sastoji od tri faze:

1. Izdvajanje jednog segmenta iz 3D slike i definisanje triangulacije izdvojenog poligona.
2. Pretvaranje triangulacije poligona u binarni ili neki drugi zapis koji odgovara svojstvu Katalanovog objekta. Iz ove faze dobijamo zapis Katalanovog ključa.
3. Primena Katalanovog ključa u šifrovanju na osnovu nekog kombinatornog problema koji se bazira na Katalanovim brojevima.

U drugoj fazi, potrebno je povezati dobijenu triangulaciju sa binarnim stablima. Binarno stablo (engl. binary tree) je poznat pojam u oblasti računarskih nauka i predstavlja strukturu namenjenu čuvanju podataka. Da bi se za svaku generisanu triangulaciju konveksnog poligona dobio odgovarajuči binarni zapis primeniće se Lukasievićev algoritam [2]. Postupak obilaska binarnih stabala po ovom algoritmu se realizuje tako što se čvorovi obeležavaju sa 0 a grane sa 1. Na taj način se dobija odgovarajući zapis (notacija), koji je jedinstven i odgovara tačno jednoj triangulaciji za koje važi to binarno stablo [59].



Slika 4.28: Prva faza: Izdvajanje jednog segmenta iz 3D slike i određivanje triangulacije

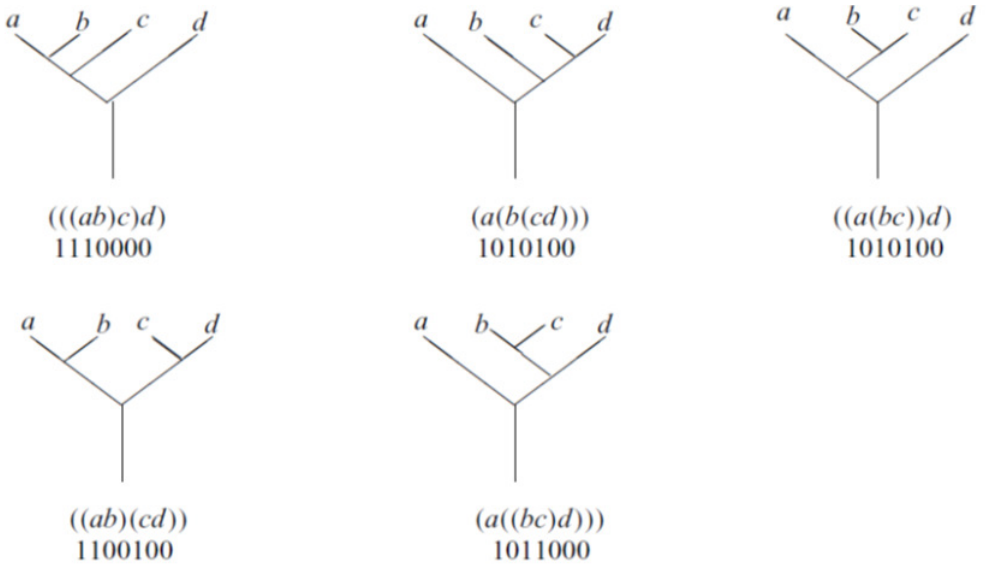


Slika 4.29: Način beleženja binarnih stabala primenom Lukasievićev-og algoritma

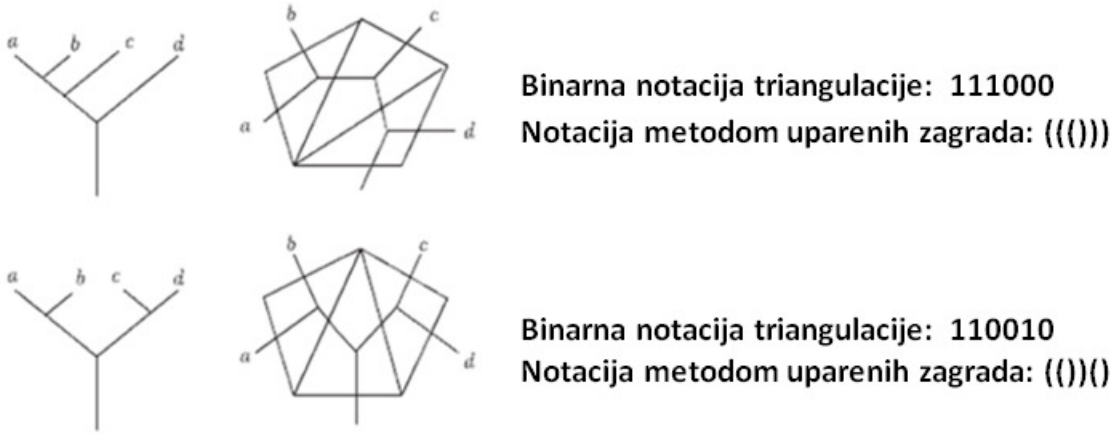
Ono što je bitno u ovom slučaju je da binarna stabla mogu reprezentovati Katalanove objekte. Svakom binarnom stablu odgovara po jedan binarni zapis. Na Slici 4.29, je prikazan postupak za generisanje binarnih zapisa za Katalanov broj C_3 a to je ukupno 5 kombinacija binarnih stabala odnosno 5 binarnih zapisa Katalanovog ključa.

Znači, sada je vrlo lako napraviti vezu između binarnog stabla i triangulacije poligona, a samim tim, jednostavno je svaku triangulaciju predstaviti u vidu binarnog ili zapisa u formi uparenih zagrada 4.31.

Primer: Metod za generisanje alfanumeričkog zapisa (u daljem tekstu AN zapis ili AN notacija) je nastao sa ciljem da uštedi memorijski prostor u postupku generisanja, distribucije i skladištenja velikih zapisa ključeva. Tačnije, AN notacija predstavlja skraćeni



Slika 4.30: Binarni zapis stabala



Slika 4.31: Druga faza: Triangulacije i odgovarajuće binarno stablo za prva dva slučaja iz prethodne slike

zapis Katalanovih ključeva koji su prvobitno predstavljeni u vidu uparenih zagrada (eng. Balanced Parentheses). Metod se zasniva na problemu balansiranih zagrada, ali može se primeniti i na druge zapise (npr. binarne zapise stabala). Sastoji se od 4 faze:

1. **Eliminacija** - prva faza gde se vrši uklanjanje prve otvorene i poslednje zatvorene zagrade. Na ovaj način nismo uticali na jedinstvenost svakog zapisa, zato što svaki zapis isto počinje i isto se završava.

2. **Zamena** - druga faza u kojoj vršimo konvertovanje karaktera u binarni zapis, pa su nam otvorene zagrade posle ove faze zamenjene u 1 a zatvorene u 0 (pravilo bit-stringa).

3. **Selekcija** - treća faza u kojoj se uzima prva i poslednja grupa od po dva ili tri binarna broja. Selektovane grupe binarnih brojeva se zamenjuju u alfa karakter (detaljno opisano u radu [58]) i ovaj deo čini alfa deo zapisa.

4. **Konverzija** - četvrta faza u kojoj se preostali centralni deo binarnog zapisa konvertuje u decimalni broj. Dobijeni rezultat iz faze konverzije čini numerički deo zapisa koji se dodaje alfa zapisu.

4.4.4 Kriptoanaliza Katalanovih ključeva u šifrovanju teksta

U izvorima [60, 61, 62] navedene su konkretne primene kombinatornih problema u kriptografiji. U ovom radu, predstavljeni su Katalanovi objekti kao ključevi za šifrovanje teksta preko određenog kombinatornog problema. Svi navedeni kombinatorni problemi u [59] mogu se rešiti na osnovu vrednosti koje poseduju svojstva Katalanovih objekata. Kratko rečeno, broj kombinacija i način generisanja Katalanovih brojeva predstavlja rešenje za određene kombinatorne probleme. Iz Tabele 1.1, možemo videti da je n osnova za generisanje ključeva a C_n određuje broj validnih ključeva u toj osnovi, tj. onih vrednosti koje zadovoljavaju svojstvo Katalanovog objekta (prostor ključeva). Na primer, za osnovu $n = 30$ imamo prostor ključeva $C_{30} = 3814986502092304$, odnosno vrednosti koje zadovoljavaju svojstvo Katalanovog objekta. Sa povećavanjem osnove n, drastično se povećava prostor ključeva. Navedena tabela prikazuje vrednosti za prvih 30 Katalanovih brojeva. Da bi obezbedili što jači tj. otporniji mehanizam šifrovanja na kriptoanalizu, za ključeve je potrebno birati uglavnom vrednosti gde je $n > 30$. Na osnovu prethodnog, u nastavku su date brojne vrednosti za n od 140 do 150.

Catalan(140)= 6563763990246161693492536077533454353889420384665868119527796560671706463922728 40
Catalan(141)= 2597771382055171036438595264488592497806939617029730903644099765561619037129981240
Catalan(142)= 1028208812757501263373597845944435911719390086180998385638154172942570891619279288 0
Catalan(143)= 4069993217165109167520491473530058817222585757799785276484360267897676445992980515 0
Catalan(144)= 1611155935622601835970180762625002593852251189639363274966912271567769848275841941 80
Catalan(145)= 6378411854725094939662770416419530816757542380901040910485447212097061454133127687 40
Catalan(146)= 2525330407789119221009341756704875466226455554887350891090156651320061065513932186 440
Catalan(147)= 9998943371381242321023474793439574481139884832189105555262377011307809353994353116 580
Catalan(148)= 3959313147057001992888490018878757680451363792611793474902551970920541958964206938 7800
Catalan(149)= 1567888006234572789183842047475988041458740061874270216061410580484534615749825947 75688
Catalan(150)= 6209251839260096211469785062189674495313420907290156219898831305495044372307257726878 24

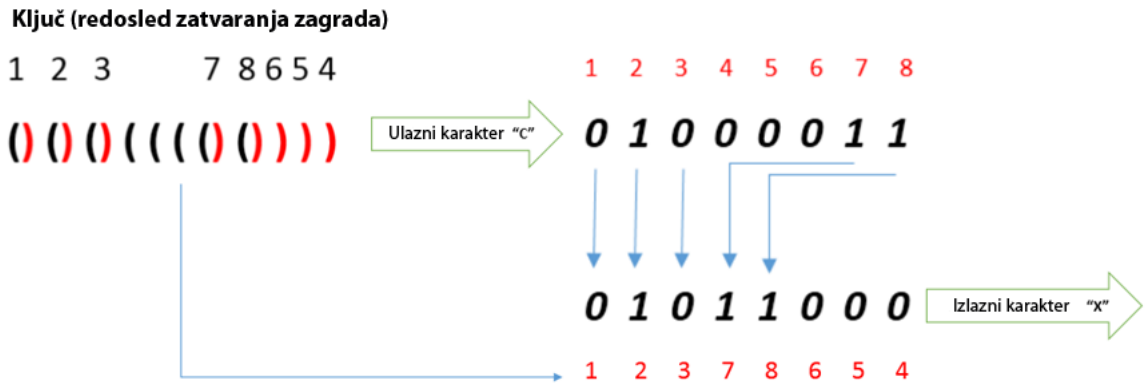
4.4.5 Primena Katalanovih ključeva za šifrovanje na primeru notacije uparenih zagrada

Kao što je već pomenuto u sekciji 2.1 binarna notacija Katalanovog ključa može se predstaviti u obliku uparenih (balansiranih) zagrada. Problem se odnosi na broj izračunavanja kombinacija mogućih parova zagrada. Taj broj mogućih validnih kombinacija je direktno

određen formulom za izračunavanje Katalanovih brojeva C_n. Ukoliko se želi binarni zapis Katalanovog ključa predstaviti u obliku notacije uparenih zagrada, onda bit 1 predstavlja otvorenu zagradu "(", a bit 0 predstavlja zatvorenu zagradu ")". Postupak šifrovanja se može primeniti na tekst poruke (String varijanta), ali se može primeniti i na binarni zapis poruke (ASCII Text to Binary). U nastavku je prikazan postupak šifrovanja teksta gde se vrednosti otvorenog teksta uzimaju u obliku binarnog zapisa (dobija se šifra supstitucije). Na osnovu rasporeda karaktera "(" i ")" u zapisu ključa, elementi mogu imati 2 stanja:

1. **Slobodan element** - To je karakter iz poruke koji nije šifrovan, tačnije nije prenet u šifrat. Slobodan element je uslovljen pojavom otvorene zagrade (u ključu), koji čeka svoj par tj. zatvorenu zagradu

2. **Zauzet element** - To je karakter iz poruke koji je šifrovan i prenet u šifrat. To je element koji je uslovljen pojavom zatvorene zagrade (u ključu). Na taj način, element je "zatvoren", tj. prenet u šifrat, jer se pojavio karakter ")", koji zatvara odgovarajući karakter "(".

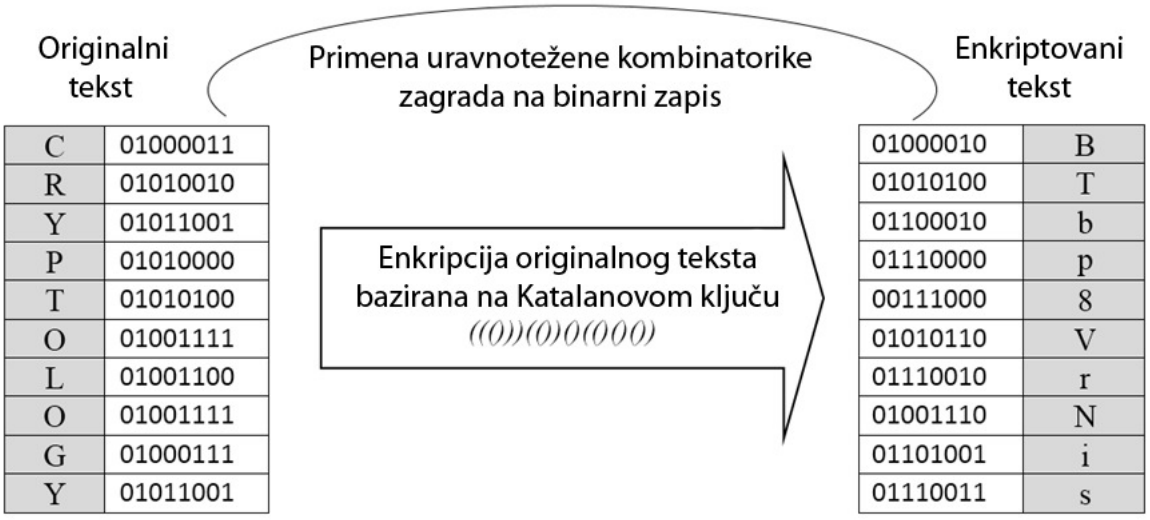


Slika 4.32: Šifrovanje karaktera C u karakter X, na osnovu Katalanovog ključa

Primer: Neka je dat otvoren tekst $P_{txt} =$ "$CRYPTOLOGY$". Ako se primeni *ASCII Text to Binary* na P, onda se dobija sekvenca bitova P_{bin}= 01000011 01010010 01011001 01010000 01010100 01001111 01001100 01001111 01000111 01011001. Primenom ključa $K = ((()))(())()(()()())$ na osnovu kojeg se vrši permutacija bitova iz poruke, dobijaju se sledeće sekvence bitova tj. binarni zapis šifrata: C_{bin}=01000010 01010100 01100010 01110000 00111000 01010110 01110010 01001110 01101001 01110011. Primenom Binary to $ASCIIText$ na šifrat C, dobija se zapis šifrata $C_{txt} =$ "$BTbp8VrNis$".

Na taj način obezbeđuje se jači mehanizam šifrovanja, odnosno jedan karakter je zamenjen nekim potpuno drugim znakom u zavisnosti od dobijene permutacije bitova. U ovom slučaju, nemamo klasičnu šifru transpozicije, kao u prethodnim primerima, već šifru supstitucije odnosno zamene. Ako se uporedi otvoren tekst $P_{txt} =$ "$CRYPTOLOGY$" i šifrat

C_{txt} = "$BTbp8VrNis$", može se uočiti da je prvi karakter "Y" zamenjen sa "b", a drugi karakter "Y" sa "s". Isti slučaj je i sa karakterima "O", gde je prvi zamenjen sa "V" a drugi sa "N".



Slika 4.33: Šifrovanje stringa CRYPTOLOGY u string BTbp8VrNis, na osnovu Katalanovog ključa

Bitno je još napomenuti da se ovde ne realizuje klasična supstitucija, jer se jedan karakter iz poruke ne zamenjuje uvek istim karakterom u šifratu. Način zamene zavisi od samog ključa i njegove dužine, kao i rasporeda bitova u ključu. Pored toga, postupak zavisi i od dužine poruke i veličine segmenata poruke koji se uzimaju u postupku šifrovanja.

4.4.6 Oblasti primene predložene metode

Glavne oblasti primene predložene metode su:

1. Biometrijska autentifikacija i generisanje kriptoloških ključeva iz 3D slike
2. Upravljanje kriptološkim ključevima i primena u kriptografiji sa javnim ključem
3. Steganografija i skrivanje tajnih informacija u slici.

Cilj višefaktorske autentikacije je da formira slojevitu odbranu, oslanjajući se na dva ili više različitih načina za utvrđivanje autentičnosti korisnika prilikom prijavljivanja u sistem (autentifikacije) i davanja privilegija pristupa (autorizacije). Ove tehnike kombinuju ono što korisnik zna, ono što korisnik poseduje i ono što fizički identifikuje korisnika (biometrijska verifikacija). Upravo, ova metoda bi mogla naći primenu u biometrijskoj autentifikaciji. Metoda se bazira na generisanju vlastitog ključa - na primer svako nosi svoj ključ u 3D prikazu karakterističnih crta lica. Znači, svako je nosilac svog jedinstvenog ključa koji

se generiše iz triangulacije skeniranog poligona. Poprilično zanimljiva mogućnost autentifikacije korisnika jer ne postoji mogućnost krađe ključa kao što je slučaj kod prethodna dva pristupa. Uvođenjem ovakvog postupka za utvrđivanje autentičnosti korisnika otežava se neovlašćeni pristup kompjuterima, mobilnim uređajima, fizičkim lokacijama, mrežama ili bazama podataka.

Sve više kompanija širom sveta odustaje od šifri i kartica za autentifikaciju i okreće se mnogo jačoj zaštiti - upotrebi biometrijskih podataka. Biometrijska autentifikacija koristi vlastite osobine korisnika u cilju što bržeg potvrđivanja nečijeg identiteta. Vodeći proizvođači pametnih telefona prepoznaju da korisnicima bezbednost igra sve važniju ulogu, pa su počeli da ugrađuju biometrijske senzore i autentifikaciju kako bi se pobrinuli da samo autorizovani korisnik može imati pristup uređaju. Primenjuju se različite tehnike, kao što su: prepoznavanje lica, autentifikacija preko otiska prsta, prepoznavanje glasa ili skeniranje mrežnjače.

Pored toga, primena ove metode bi mogla imati i drugu dimenziju, a to je u postupku upravljanja kriptološkim ključevima. Upravljanje kriptološkim ključevima je jedan od najbitnijih zadataka u savremenoj kriptografiji. Iako ovi algoritmi počivaju na obimnim akademskim istraživanjima sam proces kreiranja sigurnih i kvalitetnih kriptoloških algoritama nije jednostavan. U suštini, generisanje ključeva i održavanje tajnosti ključeva je mnogo teže. Prilikom napada na simetrične i asimetrične sisteme kriptoanalitičar će pre napasti sistem za upravljanje ključevima, nego što će pokušati da otkrije zakonitosti kriptografskog algoritma koji je primenjen. Da bi se sigurno zaštitio jedan šifarski sistem cilj je da se što kvalitetnije izvrše: generisanje, distribucija i upravljanje kriptološkim ključevima. Upravo u navedenim koracima se krije primena ove metode.

Ova metoda obezbeđuje efikasno generisanje ključa iz slike, a distribucija i upravljanje se vrše u vidu binarnog zapisa Katalanovog ključa. Znači, ova metoda obezbeđuje efikasno upravljanje ključevima sa akcentom na sigurno generisanje, distribuciju i čuvanje (skladištenje) ključeva. Bitno je naglasiti da apsolutnu slučajnost je teško definisati i njena mera se predstavlja entropijom. Sa druge strane, izvore slučajnosti moguće je postići softverski. Međutim, treba uzeti u obzir da je softver deterministički sistem, dok generisanje ključa iz pokretne slike obezbeđuje kvalitetniji slučajni niz, ali ipak mora se naglasiti da količina ovakvih nizova ostaje ograničena.

Sigurnost metoda upravljanja ključevima je od ekstremnog značaja. Kada je ključ jednom generisan on mora ostati tajan da bi se izbegle situacije kao što je impersonalizacija. Kada je reč o infrastrukturi sa javnim ključevima (PKI), u praksi se najviše napada dešava na nivou upravljanja ključevima, a vrlo retko napadi na same algoritme. Zato se predložena metoda može koristiti i u procesu skrivanja ključa u kriptografiji sa javnim ključem.

Treći aspekt primene ove metode je u postupku skrivanja informacija. Steganografija je tehnika skrivanja tajnih poruka na takav način da niko osim predajne i prijemne strane nije svestan postojanja komunikacije. Skrivanje poruka se temelji na prerušavanju poruke unutar

slike, filmova i teksta. Osnovna prednost steganografije u odnosu na kriptografiju je činjenica da poruke ne privlače pažnju na sebe. Ovom tehnikom se može izbeći napad "čovek u sredini" obzirom da napadač nije svestan postojanja komunikacije u nekom komunikacionom kanalu. U ovom slučaju, ključ se ne prenosi komunikacionim kanalom, jedinstveni ključ je sadržan (skriven) u slici. Pored ključa, na isti način je moguće skrivati i druge informacije. Predložena metoda je predstavljena u radu [10].

Poglavlje 5

Zaključna razmatranja

Sa aspekta, širokog spektra mogućnosti i područja svoje primene, najperspektivnija informaciona tehnologija današnjice je GIS. Naročito je izražena njegova primena u vojnoj analizi terena, tj. u kreiranju digitalnih modela terena (DMT). Ova digitalizacija omogućava 3D prikaz terena, odnosno njegovu vizuelizaciju u vidu rasterskih ili vektorskih slojeva. Jedna od najzastupljenijih metoda predstavljanja DMT-a je TIN model, tj. mreža nepravilih troglova čija su temena tačke sa terena sa poznatim visinama [9]. Pored primene GIS-a odnosno DMT-a u vojne svrhe i u uređajima za praćenje i navigaciju, primenjuje se i u drugim oblastima kao što su: građevinarstvo, hidroinženjerstvo, generisanju karata za rizike od poplava itd. U poslednje vreme sve značajniju ulogu ima u hidrauličnom modeliranju.

Razvojem softverskih modula za generisanje DMT-a, koji su integrisani u GPS prijemnike, omogućeno je veoma precizno i detaljno snimanje vodnog dna. Obzirom na ovako široku primenu DMT-a, ekripcija TIN modela je od velike važnosti. Kriptografija veoma dinamična oblast i u ovoj disertaciji su obuhvaćeni samo neki njeni osnovni matematički koncepti. Takođe je detaljno opisan algoritam enkripcije koordinata tačaka (temena) troglova u TIN modelu, korišćenjem svojstva *bitbalansiranosti* Katalanovih objekata i metoda stek permutacije bitova. Takođe je predstavljeno programsko rešenje u Java-Net-Beans okruženju. Realizovana GUI aplikacija poseduje sve elemente za jednostavno i efikasno šifrovanje i dešifrovanje koordinata tačaka i njihovo triangulisanje.

Predstavljen je metod Delone triangulacije slike u procesu autentifikacije na bankarski sistem. Predloženii metod je kombinacija računske geometrije i kriptografije. Ova metoda predstavlja nov iskorak za kodiranje koordinata trouglova pomoću Katalanovih objekta. Za 64-bitne ključeve postoji mnogo ukupnih važećih vrednosti koje zadovoljavaju uslov ravnoteže bitova (za $n = 32$ dimenzija prostora Katalanovih ključeva je $C_n = 5553406487704\ 8198$). Da bi se pronašli svi Katalanovi brojevi i registrovali na disku, potrebno je 44 427 251 901 MB ili oko 42 369 TB. Dakle, ova procedura je veoma zahtevna s obzirom na resurse memorije. S druge strane, ako želimo da pronađemo sve 64-bitne Katalanove brojeve, i ako nam je potreban 1 ms da pristupimo svakom elementu u skupu svih C_n, onda će vreme izvšenja trajati oko 176097 godina. Prosečno vreme će biti $176097/2 = 88048$ godina. Dakle,

ova procedura je takođe veoma zahtevna kada je vreme ograničeno.

Kao primer može se uzeti neka veća baza radi generisanja Katalanovih objekata, da bismo dokazali da se prostor Katalanovog objekata drastično povečava sa malim povećanjem baze. Na primer baza $n = 64$, daje izuzetno veliki broj vrednosti koje zadovoljavaju svojstva 128-bitnog ključa, a to je oko $368 * 10^{33}$ kombinacija. U predstavljenom kripto sistemu, za enkripciju, koriste se generisane baze koje su $n < 128$. Imajući u vidu da računarski resursi kao što su CPU vreme i memorija ograničavaju čitav proces generisanja Katalanovih objekata, postavlja se pitanje: da li je moguće generisati kompletan skup svih Katalanovih objekata u kriptoanalizi?

Ustvari, stvaranje velikog prostora Katalanovih objekata osigurava sigurnost prikazanog kriptosistema. Sama reč kriptografija asocira na bezbednost podataka u računarskim mrežama. Kao što je već pomenuto uporedo sa razvojem informacionih tehnologija, raste i mogućnost zloupotreba podataka koji se tim putem prenose. Razvoj hardvera i softvera, koji se koriste u procesu prenosa podataka, zahteva česte promene i usavršavanja istih. U svemu navedenom proizvođači nemaju vremena za detaljno testiranje opreme. Nedovoljno testiranje i predviđanje eventualnih propusta predstavlja osnovu za rad potencijalnih napadača.

Pored prethodno predstavljenih tehnika u radu je predstavljena još jedna metoda a to je skrivanje tajne poruke u slici pomoću Delone trinagulacije i Katalanovih objekata primenom stek permutacije. Glavne prednosti ove metode su upotreba slike za skrivanje tajne poruke, a zadržava je u izvornom obliku. Budući da se stego ključ sastoji od tri nezavisna dela, svaki njegov nezavisni deo možemo poslati putem medija (npr. interneta) u tri različita vremenska intervala i tako dodatno sprečiti napadača da otkrije tajnu poruku. Uz to, jedna od prednosti ove metode je što napadač ne može saznati tajnu poruku bez izabranog Katalanovog objekta i upotrebe metode šifrovanja, ili čak otkriti pravu vrednost R_k niza ili, u nekim slučajevima, dobiti šifrovanu Delone triangulaciju.

U budućnosti, ova predložena metoda može se razviti u smeru kombinacije sa nekim drugim dodatnim tehnikama. Pre svega, može se proširiti modelom predstavljenim u [49], gde su autori predstavili adaptivnu raspodelu korisnog tereta u višestrukoj steganografiji slike na temelju karaktreristike teksture slike. Uz to, može se razmotriti novi pristup vezan uz novu strategiju particije korisnog tereta u steganografiji slika u boji iz [46]. Takođe je moguća veza sa steganografskom tehnikom sa kubnom referentnom tabličnom steganografije predstavljenom u [50]. U ovoj disertaciji predložena je i metoda primene jednog algoritma triangulacije poligona u postupku generisanja kriptoloških ključeva iz jednog segmenta 3D slike.

Tako dobijen ključ ima svojstvo Katalanovih objekata, što daje prednost kada je kriptoanaliza u pitanju. U ovom slučaju ovi brojevi služe kao generatori pseudoslučajnih brojeva, koji u kombinaciji sa kombinatornim problemom uparenih zagrada, može obezbediti efikasan mehanizam šifrovanja i dešifrovanja teksta.

U nekim od budućih istraživanja može se uzeti u obzir veza predložene metode steganizacije sa metodom ugradnje vodenog žiga u slici prilikom privatnog pristupa podacima. Metodom ugradnje vodenog žiga geometrijske transformacije uništavaju sinhronizaciju između skrivenih informacija i prekrivene slike. Na ovaj način je otkrivanje ugrađenih podataka otežano ili čak onemogućeno [71]. Pored ove veze predložene metode u disertaciji se mogu povezati i sa metodom nepovratnog okvira otiska prsta koji se može poništiti, a zasnovan je na informacijama izvučenim iz Delone triangulacije pojedinih tačaka na otisku [72].

Prilozi

I) Izvorni kod u *Java*-i

Ovo poglavlje rezervisano je za predstavljanje glavnih klasa i njihovih segmenata u procesu kreiranja aplikacija za navedene metode. To su u stvari klase: `DelaunayAp, Triangulation, Triangle, Pnt, DelaunayPanel, Logic`.

A) Klasa: *DelaunayAp*

Ova klasa je zadužena za pokretanje aplikacije Voronoi-Delaunay triangulacija poligona, njene glavne metode `main()`, init(), `run()`, `actionPerformed()`, `mousePressed()`, `mouse Clicked()`

-Metoda `main()` je glavna izvršna metoda. Pored inicijalizacije `applet`-a karakteristična je i po tome što se unutar nje inicijalizuje broj temena za Delaunay triangulaciju.

-Metoda `init()` služi za inicijalizaciju `applet`-a. U ovom trenutku se zapravo stavlja u upotrebu `Swing` komponenta unutar niti za slaganje događaja.

-Metoda `run()` instalira applet kao `GUI` i na taj nčin dodaje na applet sve one komponente koje su potreben za rad aplikacije (Button-e, Radio-button, itd)

- Ostale metode `actionPerformed()`, `mousePressed()`, `mouseClicked()` vezene su za događaje sa mišem. U njima se odvijaju događaji za dodavanje temena na panelu, zatim uzimanje vrednosti koordinata `(x,y)` datih temena, kreiranje nizova za šifrovanje i dešifrovanje navedenih koordinata, poziv metoda za šifrovanje, dešifrovanje, itd.

```
1       import java.awt.*;
2       import java.awt.event.*;
3       import java.io.File;
4       import java.util.HashMap;
5       import java.util.HashSet;
6       import java.util.List;
7       import java.util.Map;
8       import java.util.Random;
9       import javax.swing.*;
10      import javax.swing.JOptionPane;
11      import java.math.BigInteger;
12      import java.io.IOException;
13      import javafx.scene.layout.Background;
```

```
14
15      public class DelaunayAp extends javax.swing.JApplet
16          implements Runnable, ActionListener, MouseListener {
17
18      private boolean debug = false;
19      private Component currentSwitch = null;
20      private static String windowTitle = "Voronoi/Delaunay triangulation";
21      private JRadioButton voronoiButton = new JRadioButton("Voronoi Diagram");
22      private JRadioButton delaunayButton= new JRadioButton("Delaunay
            triangulation");
23      private JButton clearButton = new JButton("Clear");
24      private JCheckBox colorfulBox = new JCheckBox("More colors");
25      private JButton vrednost_sifrovanih_koordinata= new JButton("The TIN
            model code");
26      private JButton vrednost_desifrovanih_koordinata= new JButton("The TIN
            model decode");
27      private JLabel circleSwitch = new JLabel("Introduce empty circles");
28      private JLabel delaunaySwitch = new JLabel("Introduce the Delaunay Edge")
            ;
29      private JLabel voronoiSwitch = new JLabel("Introduces Voronoi Edge");
30
31      int x_cor, y_cor;
32      int pom_x, pom_y;
33      int i,j,s,d,t;
34      int broj_temena;
35      int S_x_kordinata[];
36      int S_y_kordinata[];
37      int D_x_kordinata[];
38      int D_y_kordinata[];
39
40  public static void main(String[] args)
41      {
42          DelaunayAp applet = new DelaunayAp();
43          applet.init();
44          JFrame dWindow = new JFrame();
45          dWindow.setSize(800, 600);
46          dWindow.setTitle(windowTitle);
47          dWindow.setLayout(new BorderLayout());
48          dWindow.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
49          dWindow.add(applet, "Center");
50          dWindow.setVisible(true);
51          JOptionPane Pomocni_prozor = new JOptionPane();
52          String Edit_Broj_Tacaka = JOptionPane.showInputDialog("Unesite broj N
                tacaka ravni");
53          applet.broj_temena = Integer.parseInt(Edit_Broj_Tacaka);
54          if(applet.broj_temena>=3)
55             {
56           applet.S_x_kordinata=new int[applet.broj_temena];
57           applet.S_y_kordinata=new int[applet.broj_temena];
```

```
58            applet.D_x_kordinata=new int[applet.broj_temena];
59            applet.D_y_kordinata=new int[applet.broj_temena];
60              }
61           else
62           {
63             JOptionPane.showMessageDialog(null, "Uneli ste 3 tacke ravni a
                   potrebno je vise od toga pokusajte ponovo" , "Greska" ,
                   JOptionPane.PLAIN_MESSAGE);
64            }
65       }
66   public void init() {
67           try {
68               SwingUtilities.invokeAndWait(this);
69           } catch (Exception e) {
70               System.err.println("Initialization failure");
71           }
72       }
73   public void run() {
74           setLayout(new BorderLayout());
75           ButtonGroup group = new ButtonGroup();
76           group.add(voronoiButton);
77           group.add(delaunayButton);
78           JPanel buttonPanel = new JPanel();
79           buttonPanel.add(voronoiButton);
80           buttonPanel.add(delaunayButton);
81           buttonPanel.add(clearButton);
82           buttonPanel.add(new JLabel("          "));
83           buttonPanel.add(colorfulBox);
84           buttonPanel.add(new JLabel("          "));
85
86           buttonPanel.add(vrednost_sifrovanih_koordinata);
87           buttonPanel.add(new JLabel("          "));
88           buttonPanel.add(vrednost_desifrovanih_koordinata);
89           this.add(buttonPanel, "North");
90           JPanel switchPanel = new JPanel();
91           switchPanel.add(circleSwitch);
92           switchPanel.add(new Label("     "));
93           switchPanel.add(delaunaySwitch);
94           switchPanel.add(new Label("     "));
95           switchPanel.add(voronoiSwitch);
96           this.add(switchPanel, "South");
97
98           voronoiButton.addActionListener(this);
99           delaunayButton.addActionListener(this);
100          clearButton.addActionListener(this);
101          colorfulBox.addActionListener(this);
102          delaunayPanel.addMouseListener(this);
103          circleSwitch.addMouseListener(this);
104          delaunaySwitch.addMouseListener(this);
```

```
105         voronoiSwitch.addMouseListener(this);
106         voronoiButton.doClick();
107 
108         delaunayPanel.setBackground(Color.white);
109         this.add(delaunayPanel, "Center");
110         voronoiButton.addActionListener(this);
111         delaunayButton.addActionListener(this);
112         clearButton.addActionListener(this);
113         vrednost_sifrovanih_koordinata.addActionListener(this);
114         vrednost_desifrovanih_koordinata.addActionListener(this);
115         colorfulBox.addActionListener(this);
116         delaunayPanel.addMouseListener(this);
117 
118         circleSwitch.addMouseListener(this);
119         delaunaySwitch.addMouseListener(this);
120         voronoiSwitch.addMouseListener(this);
121         voronoiButton.doClick();
122     }
123 public void actionPerformed(ActionEvent e) {
124         if (debug)
125        {
126             System.out.println(((AbstractButton) e.getSource()).getText());
127         }
128         if (e.getSource() == clearButton) {
129             delaunayPanel.clear();
130         }
131 
132         if (e.getSource()== vrednost_sifrovanih_koordinata)
133         {
134             long pocetak_izvrsavanja = System.currentTimeMillis();
135             Sifruj_X_Y_Koordinate();
136             delaunayPanel.clear();
137           for (s=0; s<broj_temena; s++)
138            {
139             System.out.println(" Encrypted Coordinates X = "+D_x_kordinata[s
                    ]+" Y= "+D_y_kordinata[s]);
140                 Pnt Tacka = new Pnt(D_x_kordinata[s], D_y_kordinata[s]);
141                   if (debug) {
142                       System.out.println("Click " + Tacka);
143                               }
144                    delaunayPanel.addSite(Tacka);
145                    delaunayPanel.repaint();
146            }
147           long kraj_izvrsavanja = System.currentTimeMillis();
148           long vreme_izvrsavanja= kraj_izvrsavanja - pocetak_izvrsavanja;
149           System.out.println("Execution time: "+vreme_izvrsavanja+"
                  millisecond");
150         }
151 if (e.getSource()==vrednost_desifrovanih_koordinata)
```

```
152         {
153             Desifruj_X_Y_Koordinate();
154             delaunayPanel.clear();
155           for (s=0; s<broj_temena; s++)
156             {
157                     System.out.println("Described Coordinates X= "+
                            S_x_kordinata[s]+" Y= "+S_y_kordinata[s]);
158                     Pnt Tacka = new Pnt(S_x_kordinata[s], S_y_kordinata[s]);
159                       if (debug) {
160                           System.out.println("Click " + Tacka);
161                                   }
162                        delaunayPanel.addSite(Tacka);
163                        delaunayPanel.repaint();
164             }
165         }
166     delaunayPanel.repaint();
167     }
168  public void mousePressed(MouseEvent e)
169     {
170          if (e.getSource() != delaunayPanel)
171           {
172             return;
173           }
174         Pnt point = new Pnt(e.getX(), e.getY());
175         if (debug)
176         {
177             System.out.println("Click " + point);
178         }
179         delaunayPanel.addSite(point);
180         delaunayPanel.repaint();
181     }
182 public void mouseClicked(MouseEvent e)
183     {
184         x_cor = e.getX();
185         y_cor = e.getY();
186        if ((pom_x!=x_cor) && (pom_y!=y_cor))
187           {
188             S_x_kordinata[i]=x_cor;
189             S_y_kordinata[i]=y_cor;
190             pom_x=x_cor;
191             pom_y=y_cor;
192           System.out.println("Integer Coordinates are " +S_x_kordinata[i]+ "
                  , " +S_y_kordinata[i] +" Counter I="+i);
193            if (i==broj_temena-1)
194            {
195            JOptionPane.showMessageDialog(null," Uneli ste broj "+broj_temena+
                  " predvidjenih tacaka ","Obavestenje", JOptionPane.
                  PLAIN_MESSAGE);
196            }
```

```
197                 i++;
198             }
199     }
200 }
```

B) Klasa: *Triangulation*

Ova klasa je koncipirana tako da kreira triangulacije odnosno proverava da li svako teme trougla pripada Delone prostoru. Njene glavne metode su `delaunayPlace ()`, `update ()`, `locate ()`, `contains ()`.

-Metoda `delaunayPlace ()` se odnosi u suštini na načine pojave temena p_r iz prethodno opisanim aligoritmima. U zavisnosti od mesta gde se nalazi, pronalaze se odgovarajući trouglovi koji pripadaju trenutnoj Delone triangulaciji ili se stranice temena priključuju nekom od već postojećih trouglova.

-Metoda `update ()` se koristi za proveru validnosti Delone trogula. Drugim rečima ona je ta koja radi na pricipu legalizacije ivica, opisanom u algoritmu 1.3.1.

-Metode `locate ()`, `contains ()` su pomoćne i one služe za proveru lokacije pojave temana p_r unutar trougla ili na njegovoj ivici.

```
1
2  public class Triangulation extends AbstractSet<Triangle> {
3
4      private Triangle mostRecent = null;
5      private Graph<Triangle> triGraph;
6      public Triangulation (Triangle triangle) {
7          triGraph = new Graph<Triangle>();
8          triGraph.add(triangle);
9          mostRecent = triangle;
10     }
11     @Override
12     public Iterator<Triangle> iterator () {
13         return triGraph.nodeSet().iterator();
14     }
15
16     @Override
17     public int size () {
18         return triGraph.nodeSet().size();
19     }
20
21     @Override
22     public String toString () {
23         return "Triangulation with " + size() + " triangles";
24     }
25
26     public boolean contains (Object triangle) {
27         return triGraph.nodeSet().contains(triangle);
```

```
28      }
29
30      public Triangle neighborOpposite (Pnt site, Triangle triangle) {
31          if (!triangle.contains(site))
32              throw new IllegalArgumentException("Bad vertex; not in triangle")
                    ;
33          for (Triangle neighbor: triGraph.neighbors(triangle)) {
34              if (!neighbor.contains(site)) return neighbor;
35          }
36          return null;
37      }
38
39      public Set<Triangle> neighbors(Triangle triangle) {
40          return triGraph.neighbors(triangle);
41      }
42
43      public List<Triangle> surroundingTriangles (Pnt site, Triangle triangle)
            {
44          if (!triangle.contains(site))
45              throw new IllegalArgumentException("Site not in triangle");
46          List<Triangle> list = new ArrayList<Triangle>();
47          Triangle start = triangle;
48          Pnt guide = triangle.getVertexButNot(site);
49          while (true) {
50              list.add(triangle);
51              Triangle previous = triangle;
52              triangle = this.neighborOpposite(guide, triangle);
53              guide = previous.getVertexButNot(site, guide);
54              if (triangle == start) break;
55          }
56          return list;
57      }
58
59      public void delaunayPlace (Pnt site)
60        {
61          Triangle triangle = locate(site);
62            if (triangle == null)
63                throw new IllegalArgumentException("No containing triangle");
64           if (triangle.contains(site)) return;
65            Set<Triangle> cavity = getCavity(site, triangle);
66            mostRecent = update(site, cavity);
67      }
68
69      private Triangle update (Pnt site, Set<Triangle> cavity)
70         {
71          Set<Set<Pnt>> boundary = new HashSet<Set<Pnt>>();
72          Set<Triangle> theTriangles = new HashSet<Triangle>();
73          for (Triangle triangle: cavity) {
74              theTriangles.addAll(neighbors(triangle));
```

```
75              for (Pnt vertex: triangle) {
76                  Set<Pnt> facet = triangle.facetOpposite(vertex);
77                  if (boundary.contains(facet)) boundary.remove(facet);
78                  else boundary.add(facet);
79              }
80          }
81
82          theTriangles.addAll(newTriangles);
83          for (Triangle triangle: newTriangles)
84              for (Triangle other: theTriangles)
85                  if (triangle.isNeighbor(other))
86                      triGraph.add(triangle, other);
87          return newTriangles.iterator().next();
88      }
89  }
```

D) Klasa: *Triangle*

Ova klasa je programirana za konsturkciju trougla pomoću zadatih koordinata temena. Korišćene metode koje su neophodne za realizaciju ove klase su `isNeighbor ()`, `facetOpposite ()`, `getCircumcenter ()` `add ()`.

```
1
2     import java.util.Arrays;
3     import java.util.Collection;
4     import java.util.Iterator;
5     import java.util.NoSuchElementException;
6
7   class Triangle extends ArraySet<Pnt>
8     {
9      private int idNumber;
10     private Pnt circumcenter = null;
11
12     public boolean isNeighbor (Triangle triangle)
13       {
14         int count = 0;
15         for (Pnt vertex: this)
16             if (!triangle.contains(vertex)) count++;
17         return count == 1;
18        }
19
20
21     public Pnt getCircumcenter ()
22       {
23         if (circumcenter == null)
24             circumcenter = Pnt.circumcenter(this.toArray(new Pnt[0]));
25         return circumcenter;
26        }
```

```
27
28 }
```

E) Klasa: *Pnt*

Klasa `Pnt` je potrebna iz razloga prikupljanja koordinata `(x,y)` slučajno izabranih temena. Temena se nalaze u Euklidovom prostoru, predstavljena nizom realnih brojeva. Ova klasa uključuje jednostavne geometrijske operacije i rad sa matricama. Neke metode koje su neophodne za rad sa objektima ove klase su: `coord ()`, `dimension ()`, `dimCheck ()`, `extend()`, `add ()`, `angle ()`.

```
1
2 public class Pnt {
3
4     private double[] coordinates;
5     public Pnt (double... coords)
6    {
7         coordinates = new double[coords.length];
8         System.arraycopy(coords, 0, coordinates, 0, coords.length);
9     }
10
11    public double coord (int i) {
12        return this.coordinates[i];
13    }
14    public int dimension () {
15        return coordinates.length;
16    }
17    public int dimCheck (Pnt p) {
18        int len = this.coordinates.length;
19        if (len != p.coordinates.length)
20            throw new IllegalArgumentException("Dimension mismatch");
21        return len;
22    }
23    public Pnt extend (double... coords) {
24        double[] result = new double[coordinates.length + coords.length];
25        System.arraycopy(coordinates, 0, result, 0, coordinates.length);
26        System.arraycopy(coords, 0, result, coordinates.length, coords.length
              );
27        return new Pnt(result);
28    }
29
30    public Pnt add (Pnt p) {
31        int len = dimCheck(p);
32        double[] coords = new double[len];
33        for (int i = 0; i < len; i++)
34            coords[i] = this.coordinates[i] + p.coordinates[i];
35        return new Pnt(coords);
```

```
36      }
37      public double angle (Pnt p) {
38          return Math.acos(this.dot(p) / (this.magnitude() * p.magnitude()));
39      }
40
41  }
```

F) Klasa: *DelaunayPanel*

Ova klasa se koristi za iscrtavanje komponenti definisanih u klasi `DelaunayApp`. Njen konstruktor koji u sebi ima parametra `DelaunayApp` klase koristi se za kreiranje inicijalne triangulacije opisane u algoritmima kao $\mathcal{T}$, a sastoji se od trougla p_{-1}, p_{-2} i p_{-3}. Neke metode koje su karakteristične za ovu klasu su: `addSite()`, `clear()`, `getColor()`, `paintComponent()`, `drawAllDelaunay`, `drawAllVoronoi()`. U nastavku je predstavlj en segment koda ove klase.

```
1
2   class DelaunayPanel extends JPanel {
3
4       Image img=background_image.getImage();
5       public static Color voronoiColor = Color.white;
6       public static Color delaunayColor = Color.white;
7       public static int pointRadius = 3;
8
9       private DelaunayAp controller;
10      private Triangulation dt;
11      private Map<Object, Color> colorTable;
12
13      public DelaunayPanel(DelaunayAp controller)
14       {
15          this.controller = controller;
16          initialTriangle = new Triangle(
17                  new Pnt(-initialSize, -initialSize),
18                  new Pnt(initialSize, -initialSize),
19                  new Pnt(0, initialSize));
20          dt = new Triangulation(initialTriangle);
21          colorTable = new HashMap<Object, Color>();
22       }
23
24      public void addSite(Pnt point)
25       {
26          dt.delaunayPlace(point);
27       }
28      public void clear()
29       {
30          dt = new Triangulation(initialTriangle);
31       }
32      private Color getColor(Object item) {
```

```
33          if (colorTable.containsKey(item)) {
34              return colorTable.get(item);
35          }
36
37       Color color = new Color(Color.HSBtoRGB(random.nextFloat(), 1.0f, 1.0f));
38          colorTable.put(item, color);
39          return color;
40      }
41
42      public void draw(Pnt point) {
43          int r = pointRadius;
44          int x = (int) point.coord(0);
45          int y = (int) point.coord(1);
46          g.fillOval(x - r, y - r, r + r, r + r);
47      }
48
49         public void draw(Pnt[] polygon, Color fillColor) {
50          int[] x = new int[polygon.length];
51          int[] y = new int[polygon.length];
52          for (int i = 0; i < polygon.length; i++) {
53              x[i] = (int) polygon[i].coord(0);
54              y[i] = (int) polygon[i].coord(1);
55
56          }
57          if (fillColor != null) {
58              Color temp = g.getColor();
59              g.setColor(fillColor);
60              g.fillPolygon(x, y, polygon.length);
61              g.setColor(temp);
62          }
63          g.drawPolygon(x, y, polygon.length);
64      }
65
66  }
```

G) Klasa: *Logic*

Klasa `Logic` prvenstveno se odnosi na kreiranje aplikacije za autentifikaciju na bankarski sistem tj. na Delone triangulaciji i bojenju trouglova sa bojom centralnog piksela trougla koji zadovoljava uslov Delone triangulacije. Metode koje su neophodne za njenu realizaciju su: `createRandomPoints()`, `findFirstPoint()`, `Logic findSecondPoint()`, `findThirdPoint()`, `createTriangulation()`, `flipAdjacentTriangles()`, `colorTriangulation()`. Neke od ovih metoda kao i data klasa predstavljeni su u nastavku.

```
1
2  public class Logic extends Main  {
3
```

```
4       Random random=new Random();
5       public int numberOfPoints;
6
7       public int i,j,s,d,t;
8       public int S_x_kordinata[];
9       public int S_y_kordinata[];
10      public int D_x_kordinata[];
11      public int D_y_kordinata[];
12
13      String Katalan_Kljuc;
14      String Binarna_Vrednost_X_Koordinate;
15      String Binarna_Vrednost_Y_Koordinate;
16      String Sifrovana_Vrednost_Binarne_X_Kordinate;
17      String Sifrovana_Vrednost_Binarne_Y_Kordinate;
18      Integer Sifrovana_Vrednost_Integer_X_Kordinate;
19      Integer Sifrovana_Vrednost_Integer_Y_Kordinate;
20
21      String Binarna_Vrednost_D_X_Koordinate;
22      String Binarna_Vrednost_D_Y_Koordinate;
23      String Sifrovana_Vrednost_Binarne_D_X_Kordinate;
24      String Sifrovana_Vrednost_Binarne_D_Y_Kordinate;
25      Integer Sifrovana_Vrednost_Integer_D_X_Kordinate;
26      Integer Sifrovana_Vrednost_Integer_D_Y_Kordinate;
27
28      Point P_C;
29      Circle initialTringleCircle;
30      int p1, p2, p3;
31      int pointsCounter=0;
32      public File Selected_file_Del;
33      public BufferedImage help_background;
34      public boolean sifrovano=false;
35      public BufferedImage img=null;
36
37  public void createRandomPoints()
38      {
39
40          System.out.println("Kreiranje nasumicnih tacaka i sifrovano="+
                sifrovano);
41          if (sifrovano==true)
42          {
43              for (int i=0; i<points.length; i++)
44             {
45                 points[i]=new Point(D_x_kordinata[i], D_y_kordinata[i]);
46
47                   System.out.println("Sifrovane koordinate su "+i+" koordinata
                        X="+D_x_kordinata[i] + " Y="+D_y_kordinata[i]);
48
49                    while (unique(i)==false)
50                      {
```

```
51                          System.out.print("Ulazi i jedinstvenu petlju Unique
                                Sifrovane");
52                      points[i]=new Point(random.nextInt(width), random.
                            nextInt(height));
53                    }
54              }
55          } else
56              {
57                for (int i=0; i<points.length; i++)
58                {
59                  points[i]=new Point(random.nextInt(width), random.nextInt(
                        height));
60
61                   S_x_kordinata[i]=(int) Math.round(points[i].x);
62                   S_y_kordinata[i]=(int) Math.round(points[i].y);
63
64                   System.out.println("Izabrana slucajna "+i+" koordinata X="+
                         S_x_kordinata[i] + " Y="+S_y_kordinata[i]);
65
66                   while (unique(i)==false)
67                 {
68                      System.out.print("Ulazi i jedinstvenu petlju Unique
                            Izabrane");
69                   points[i]=new Point(random.nextInt(width), random.nextInt(
                         height));
70                 }
71                }
72              }
73      }
74
75  }
```

Literatura

[1] Jovanović V., Djurdjev B., Srdić Z., Stankov U., *Geografski Informacioni sistemi*, Univerzitet u Novom Sadu, 2012.

[2] Koshy T., *Catalan Numbers with Applications*, Oxford University Press, New York 2009.

[3] Saračević, M., Stanimirović, P., Krtolica, P., Mašović, S., Construction and Notation of Convex Polygon Triangulation based on ballot problem, *ROMJIST - Journal of Information Science and Technology*, 17(3), pp. 237-251, 2014.

[4] Geary, F.R., Rahman, N., Raman, R., A Simple Optimal Representation for Balanced Parentheses, *Theoretical Computer Science*, 368 (3), pp. 231-246, 2006.

[5] O'Rourke J., *Computational Geometry in C*, Cambridge University Press; 2 edition, 1998.

[6] Berg M., Kreveld M., Overmars M., Schwarzkopf O., *Computational Geometry Algorithms and Applications*, Springer-Verlag Berlin Hidelberg, 1997.

[7] Djurović E., *Algoritmi za konstrukciju Voronoj dijagrama i Deloneove triangulacije*, Master teza, Unviverzitet u Beogradu, Matematički fakultet.

[8] Janičić P., *Računarska Geometrija*, Unviverzitet u Beogradu, Matematički fakultet, Beograd 2016.

[9] Gigović J. Lj., *Digitanli modeli visina i njihova primena u vojnoj analizi terena*, Vojna akademija, Katedra prirodno-matematičkih i tehničkih nauka, Beograd 2009.

[10] Saračević, M., Aybeyan Selimi, Faruk Selimović. *Generation of Cryptographic Keys with Algorithm of Polygon Triangulation and Catalan Numbers*, Computer Science, 19(3). https://doi.org/10.7494/csci.2018.19.3.2749.

[11] Saračević M., Koricanin E., Bisevac E., Encryption based on Ballot, Stack permutations and Balanced Parentheses using Catalan-keys, *Journal of Information Technology and Applications*, 7(2), pp. 69–77, 2017.

[12] Saračević M., Hadzic M., Koricanin E., Generating Catalan-keys based on dynamic programming and their application in steganography, *International Journal of Industrial Engineering and Management*, 8(4), pp. 219–227, 2017.

[13] Carlitz, L. (1972) *Sequences, paths and ballot numbers, Fibonnaci Quarterly*, Vol. 10, pp. 531-549.

[14] H. C. Kazi, B. Stanley, D. Lilja, *Techniques for Obtaining High Performance in Java Programs*, ACM Computing Surveys 32 (3), (2000), 213–240

[15] S. Mašović, M. Saračević , H. Kamberović , *Java technology in the design and imple mentation of web applications*, Technics Technologies Education Management 7 (2), (2012), 504–512.

[16] M. Berg, O.Cheong, M. Kreveld, M. H. Overmars, *Computational geometry: Algo rithms and applications: 3rd edition*, Springer Verlag, 2008.

[17] P. Schneider, D. Eberly, *Geometric Tools for Computer Graphics* , Morgan Kaufmann Publishers, San Francisco, 2003.

[18] P.S. Stanimirović, P.V. Krtolica, M. Saračević, S. Mašović, *Decomposition of Cata lan numbers and Convex Polygon Triangulations*, International Journal of Computer Mathematics, (2013), DOI:10.1080/00207160.2013.837894

[19] D. M. Geary, *Graphic Java 2: Swing, volume 2 / 3rd edition*, Prentice Hall Profes sional, 1999.

[20] M. Saračević, S. Mašović, H. Kamberović, *Implementacija nekih algoritama računarske grafike u JAVA NETBEANS okrženju*, XVI International Scientific and professional conference – Information Technology (2012), 136-140.

[21] G. Davis, *Learning Java Bindings for OpenGL*, Author-House publisher, 2004.

[22] F. Klawonn, *Introduction to Computer Graphics: Using Java 2D and 3D: Second Edition*, Springer, 2012.

[23] B. Chen, H. Harry, *Interpretive OpenGL for Computer Graphics, Computers and Graphics 29 (2)*, (2005), pp. 331–339.

[24] J. X. Chen, C. Chen, *Foundations of 3D Graphics Programming: Using JOGL and Java3D*, Springer, 2008.

[25] Li, C.; Zhang, Y.; Xie, E.Y. *When an attacker meets a cipher-image in 2018:* A year in review. J. Inf. Secur. Appl. 2019, 48, 102361, doi:10.1016/j.jisa.2019.102361.

[26] Lee D.T., Preparata F.P., *Computational Geometry - A Survey, IEEE Transactions On Computers*, Vol c 33 (12), 1984.

[27] Luan, G., Li, A. , Zhang, D. , Wang, D. *Asymmetric image encryption and authen tication based on equal modulus decomposition in the fresnel transform domain* IEEE Photonics Journal Open Access Volume 11, Issue 1, February 2019, Article number 8572731

[28] Lin Yuan, Qiwen Ran, Tieyu Zhao *Image authentication based on double image en cryption and partial phase decryption in nonseparable fractional Fourier domain* Optics Laser Technology Volume 88, February 2017, Pages 111–120

[29] Huaqian Yang, Kwok-WoWong, Xiaofeng Liaoc, Wei Zhang, Pengcheng Wei *A fast image encryption and authentication scheme based on chaotic maps* Communications in Nonlinear Science and Numerical Simulation Volume 15, Issue 11, November 2010, Pages 3507–3517

[30] Zanobya N. Khan, Rashid Jalal Qureshi, Jamil Ahmad *On Feature based Delaunay Triangulation for Palmprint Recognition* JOURNAL OF PLATFORM TECHNOL OGY, VOL.3, NO.4, DECEMBER 2015

[31] S. Vijaya Ranjini, S. Rajarajan *Enhanced Fingerprint Recognition with OTP using Delaunay Triangulation to Improve ATM Security* Indian Journal of Science and Technology, Vol 9(S1), DOI: 10.17485/ijst/2016/v9iS1/108020, December 2016

[32] Selimovic Faruk, Stanimirovic Predrag, Saracevic Muzafer, Selimi Aybeyan, Krtolica Predrag *Authentication Based on the Image Encryption using Delaunay Triangulation and Catalan Objects* ACTA POLYTECHNICA HUNGARICA, (2020), vol. 17 br. 6, str. 207–224

[33] Danijel C. Clarke, Emil E. Larsen, Frederik Z. Thulstrup *Using Delaunay triangulation to create triangulated images* Roskilde University 5th semester project IMT December 18, 2017

[34] S. Huynh, Y. Cai, W. Shen, *Automatic Transformation of UML Models into Analytical Decision Models*, Technical Report DU-CS-08-01, Drexel University, 2008.

[35] Saračević, M., Jukić, S., Hasanovič, A. *A Steganography Method Based on Decomposition of the Catalan Numbers. In Digital Media Steganography—Principles, Algorithms, Advances*; Elsevier: Amsterdam, The Netherlands, 2020; ISBN 9780128194386.

[36] Saračević M., Adamović S., Miškovic V., Maček N., Šarac, M. *A novel approach to steganography based on the properties of Catalan numbers and Dyck words.* Future Gener. Comput. Syst. 2019, 100, 186–197.

[37] Pund-Dange S., Desai C.G., *Data Hiding Technique using Catalan-Lucas Number Sequence.* Indian J. Sci. Technol. 2017, 10, 12–17.

[38] Aroukatos N.,Manes K., Zimeras S., Georgiakodis, F. *Techniques in Image Steganography using Famous Number Sequences*. Int. J. Comput. Technol. 2013, 11, 2321–2329.

[39] Bhaskari D.L., Avadhani P.S., Damodaram A., *Combinatorial approach for information hiding using steganography and godelization techniques*. Int. J. Syst. Cybern. Inform. 2007,10, 21–24.

[40] Gutierrez-Cardenas, J.M. *Secret key steganography with message obfuscation by pseudo-random number generators*. In Proceedings of the 38th IEEE International Computer Software and Applications Conference Workshops, Vasteras, Sweden, 21–25 July 2014.

[41] Sahu A.K., Swain G., *An Optimal Information Hiding Approach Based on Pixel Value Differencing and Modulus Function*. Wirel. Pers. Commun. 2019, 108, 159–174.

[42] Sahu A.K., Swain G., *A novel n-rightmost bit replacement image steganography technique*. 3D Res. 2019, 10, 2.

[43] Sahu, A.K., Swain G., *Pixel Overlapping Image Steganography using PVD and Modulus Function*. 3D Res. 2018, 9, 40.

[44] Swain, G.; Sahu, A.K. *A Novel Multi Stego-image based Data Hiding Method for Gray Scale Image*. Pertanika J. Sci. Technol. 2019, 27, 753–768.

[45] Sahu A.K., Swain G., Babu E., *Digital image steganography using Bit Flipping*. Cybern. Inf. Technol. 2018, 18, 69–80, doi:10.2478/cait-2018-0006.

[46] Liao X., Yu, Y., Li, B., Li, Z., Qin, Z. *A New Payload Partition Strategy in Color Image Steganography*. IEEE Trans. Circuits Syst. Video Technol. 2020, 30, 685–696, doi:10.1109/TCSVT.2019.2896270.

[47] Faruk Selimović, Predrag Stanimirović, Muzafer Saračević, Selver Pepić *ENCRYPTION OF 3D PLANE IN GIS USING VORONOI-DELAUNAY TRIANGULATIONS AND CATALAN NUMBER* Facta Universitatis ISSN:0352-9665 Vol.35, No 4 (2020) https://doi.org/10.22190/FUMI2004205S Page 1205–1217.

[48] Fridrich, J.; Goljan, M.; Du, R. *Reliable detection of LSB steganography in color and grayscale images, In Proceedings of the 2001 Workshop on Multimedia and Security*: New Challenges, Ottawa, ON, USA, 30 September–5 October 2001; pp. 27–30.

[49] Liao X., Yin J., Chen M., Qin Z., *Adaptive Payload Distribution in Multiple Images Steganography Based on Image Texture Features*. IEEE Trans. Dependable Secur. Comput. 2020, doi:10.1109/TDSC.2020.3004708.

[50] Liao X., Guo S., Yin J., Wang H., Li X., Sangaiah A.K. *New cubic reference table based image steganography.* Multimed. Tools Appl. 2018, 77, 10033–10050, doi:10.1007/s11042–017–4946–9.

[51] Saračević M., Stanimirović P., Mašović S., Biševac E, *Implementation of the convex polygon trangulation algorithm*, Facta Universitatis, series: Mathematics and Informatics Vol.27, No.2, pp. 213–228, 2012.

[52] Stanimirović P., Krtolica, P., Saračević M., Mašović, S. *Decomposition of Catalan numbers and Convex Polygon Triangulations*, International Journal of Computer Mathematics, 2014, Vol. 91, No. 6, pp. 1315–1328.

[53] Saračević M.,*Methods for solving the polygon triangulation problem and their implementation (PhD thesis)*, Faculty of Science and Mathematics, University of Niš, 2013

[54] Saračević M., *Application of Catalan numbers and some combinatorial problems in cryptography (Bachelor's thesis)*, Faculty of Informatics and Computing, Singidunum University in Belgrade, 2017

[55] Amounas, F., El-Kinani, E.H., Hajar, M. Novel *Encryption Schemes Based on Catalan Numbers*, International Journal of Information and Network Security, 2013, Vol.2, No.4, pp. 339–347.

[56] Cohen, E., Hansen, T., Itzhaki, N. *From entanglement witness to generalized Catalan numbers*, Scientific Reports, 2016, Vol.6, No.3.

[57] Higgins, P.M. Number Story: *From Counting to Cryptography, Springer Science and Business Media*, Berlin, Germany, 2008.

[58] Mašović, S., Saračević, M., Stanimirović, P. *Alpha-Numeric notation for one Data Structure in Software Engineering*, Acta Polytechnica Hungarica: Journal of Applied Sciences, 2014, Vol.11, No.1, pp.193-204.

[59] Stanley, R. P. *Catalan addendum to Enumerative Combinatorics*, [on-line], 2012, http://www-math.mit.edu/ rstan/ec/catadd.pdf. [Available 24.05.2017.]

[60] Horak P., Semaev, I., Tuza, I. Z. *An application of Combinatorics in Cryptography*, Electronic Notes in Discrete Mathematics, 2015, Vol. 49, pp. 31-35.

[61] Koscielny, C., Kurkowski, M., Srebrny, M. *Modern Cryptography Primer: Theoretical Foundations and Practical Applications*, Springer Science and Business Media, Berlin, Germany, 2013.

[62] Lachaud G., Ritzenthaler C., Tsfasman M.A. *Arithmetic, Geometry, Cryptography, and Coding Theory*, American Mathematical Society, United States, 2009.

[63] Anderas Pomarolli *JavaFX Programming Cookbook* Exelixis Media P.C., 2016

[64] Monica Pawlan, Cindy Castillo *JavaFX Overview, Release 2.2.21* Oracle and/or its affiliates 2011, 2013

[65] Faruk Selimović, Predrag Stanimirović, Muzafer Saračević, Predrag Krtolica *Application of Delaunay Triangulation and Catalan Objects in Steganography* Mathematics 2021, 9(11), 1172; https://doi.org/10.3390/math9111172

[66] All Answers Ltd. (November 2018). *Image Based Steganographyusing LSB Insertion Technique*. Preuzeto sa: https://ukdiss.com/examples/image-based-steganographyusing.php?vref=1

[67] *Java Binding for OpenGL (JOGL)*,
Preuzeto sa *https* : *//www.tutorialspoint.com/jogl/jogl_tutorial.pdf*

[68] *Java 2D Graphics and Imaging*, Oracle Documention,
Preuzeto sa https://docs.oracle.com/javase/8/docs/technotes/guides/2d/

[69] *Java 3D API*, Oracle Documention,
Preuzeto sa: https://www.oracle.com/java/technologies/javase/java-3d.html

[70] *Difference between AWT and Swing in Java*,
Preuzeto sa: https://www.geeksforgeeks.org/difference-between-awt-and-swing-in-java/

[71] D. Essaidani, H. Seddik and E. Ben Braiek, *"Robust and blind watermarking approach based on modified Delaunay triangulation,"* 2016 2nd International Conference on Advanced Technologies for Signal and Image Processing (ATSIP), 2016, pp. 16-20, doi: 10.1109/ATSIP.2016.7523049.

[72] Amit KumarTrivedia, Dalton Meitei, Thounaojama, Shyamosree Pala, *Non-Invertible cancellable fingerprint template for fingerprint biometric* Computer Vision Laboratory Department of Computer Science and Engineering National Institute of Technology Silchar, India, https://doi.org/10.1016/j.cose.2019.101690

Biografija autora

Faruk Selimović je rođen 30.09.1985. godine u Novom Pazaru. Osnovnu školu i srednju Elektro-Tehničku školu završio je u rodnom gradu sa odličnim uspehom. Osnovne studije upisuje na Fakultetu za informatiku i informacione tehnologije Univerziteta u Novom Pazaru 2004. godine i završava ih 2008. godine sa prosekom 9,00. Nakon završetka osnovnih studija biva angažovan kao saradnik u nastavi. Iste godine 2008. upisuje Master studije na Departmanu za Računarske nauke, Univerziteta u Novom Pazaru i završava ih 2010. sa prosečnom ocenom 9,60. Doktorske studije je upisao 2012. godine na Prirodno-matematičkom fakultetu Univerziteta u Nišu, na Departmanu za računarske nauke. Sve ispite i studijsko-istraživačke radove je položio sa prosečnom ocenom 9,20.

Od 2010. godine, angažovan je kao asistent na Departmanu za tehničke nauke, na predmetima: *Uvod u programiranje I (Object Pascal, C, C++), Arhitektura računara, Računarska tehnika, Web programiranje (PHP, HTML/CSS, JavaScript), Informacioni sistemi (Java programski jezik), Baze podataka (Oracle) Opšta Informatika (Word, Excel, PowerPoint).*

Od 2016 godine radio je kao Softwer Developer – Implementator Poslovnog Softvera u IT kompaniji "Art Computer" Novi Pazar - po modulima Maloprodaja, Veleprodaja, Magacinsko Poslovanje, Proizvodnja, Obračun zarada. (Object PASCAL-u i Firebird baza, Delphi, Report Builder, IBExpert alati). Robno i finansijsko poslovanje navedenih modula. Ternutno zapošljen u Gimnaziji u Novom Pazaru kao profesor na IT smeru za učenike sa posebnim sposobnostima za računarstvo i informatiku.

Autor je oko 18 naučnih i stručnih radova, od toga preko 10 naučnih radova u međunarodnim i domaćim časopisima, od kojih je 3 u časopisima sa SCI/SCIE liste.
Oblasti interesovanja su mu: *Objektno-orijentisano programiranje; Računarska geometrija i grafika; Projektovanje informacionih sistema, Kriptografija, Steganografija ...*

ПРИРОДНО - МАТЕМАТИЧКИ ФАКУЛТЕТ
НИШ

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, **РБР**:	
Идентификациони број, **ИБР**:	
Тип документације, **ТД**:	монографска
Тип записа, **ТЗ**:	текстуални / графички
Врста рада, **ВР**:	докторска дисертација
Аутор, **АУ**:	Фарук Б. Селимовић
Ментор, **МН**:	Предраг С. Станимировић
Наслов рада, **НР**:	Примена Воронои - Делоне триангулација и Каталанових објеката у заштити података
Језик публикације, **ЈП**:	српски
Језик извода, **ЈИ**:	енглески
Земља публиковања, **ЗП**:	Србија
Уже географско подручје, **УГП**:	Србија
Година, **ГО**:	2021
Издавач, **ИЗ**:	ауторски репринт
Место и адреса, **МА**:	Ниш, Вишеградска 33.
Физички опис рада, **ФО**: (поглавља/страна/ цитата/табела/слика/графика/прилога)	5 поглавља/ 100 страна/ 72 референци/ 11 табела/ 48 слика/ 1 прилог
Научна област, **НО**:	Рачунарске науке
Научна дисциплина, **НД**:	Рачунарска геометрија, Објектно - оријентисано програмирање
Предметна одредница/Кључне речи, **ПО**:	Воронои - Делоне триангулација полигона, Јава програмирање, Криптографија, Стеганографија
УДК	004.4, 004.415.24/.26, 004.415.3
Чува се, **ЧУ**:	
Важна напомена, **ВН**:	
Извод, **ИЗ**:	Ова докторска дисертација је из области рачунарске геометрије у комбинацији са методама за заштиту података, са акцентом на имплементацију апликација техникама објектно оријентисаног програмирања. Представљене су нове методе за заштиту података помоћу Воронои - Делоне триангулација. Прва метода заснована је на енкрипцији 3Д равни у ГИС-у. Друга метода је заснована на аутентификацији корисника (клијената) банака, енкрипцијом слике помоћу инкременталног алгоритма Воронои - Делоне триангулација и Каталанових објеката. Трећа метода, је метода стеганографије која се темељи на комбинацији Каталанових објеката и Воронои - Делоне триангулације. Четврта метода се односи на предлог примене једног алгоритма рачунарске геометрије у поступку генерисања Криптографских кључева из једног сегмента 3Д слике.
Датум прихватања теме, **ДП**:	**23.12.2020.**
Датум одбране, **ДО**:	
Чланови комисије, **КО**: Председник:	
Члан:	
Члан:	
Члан, ментор:	

PRIRODNO-MATEMATIČKI FAKULTET
NIŠ

KEY WORDS DOCUMENTATION

Accession number, **ANO**:		
Identification number, **INO**:		
Document type, **DT**:		monograph
Type of record, **TR**:		textual / graphic
Contents code, **CC**:		doctoral dissertation
Author, **AU**:		Faruk B. Selimović
Mentor, **MN**:		Predrag S. Stanimirović
Title, **TI**:		Application of Voronoi-Delaunay triangulation and Catalan objects in data protection
Language of text, **LT**:		Serbian
Language of abstract, **LA**:		English
Country of publication, **CP**:		Serbia
Locality of publication, **LP**:		Serbia
Publication year, **PY**:		2021
Publisher, **PB**:		author's reprint
Publication place, **PP**:		Niš, Višegradska 33.
Physical description, **PD**: (chapters/pages/ref./tables/pictures/graphs/appendixes)		5 chapters/ 100 pages/ 72 references/ 11 tables/ 48 images/ 1 appendix
Scientific field, **SF**:		Computer Science
Scientific discipline, **SD**:		Computational Geometry, Object-oriented programming
Subject/Key words, **S/KW**:		Voronoi - Delaunay polygon triangulation, Java programming, Cryptography, Steganography
UC		004.4, 004.415.24/.26, 004.415.3
Holding data, **HD**:		
Abstract, **AB**:		This doctoral dissertation is in the field of computer geometry in combination with data protection methods, with an emphasis on the implementation of applications using object-oriented programming techniques. New methods for data protection using Voronoi - Delaunay triangulation were presented. The first method is based on 3D plane encryption in GIS. The second method is based on the authentication of users (clients) of banks, by encrypting the image using the incremental algorithm Voronoi - Delone triangulation and Catalan objects. The third method is the method of steganography, which is based on a combination of Catalan objects and Voronoi - Delone triangulation. The fourth method refers to the proposal of application of one algorithm of computer geometry in the procedure of generating Cryptographic keys from one segment of 3D image.
Accepted by the Scientific Board on, **ASB**:		**23.12.2020.**
Defended on, **DE**:		
Defended Board, **DB**:	President:	
	Member:	
	Member:	
	Member, Mentor:	

ИЗЈАВА О АУТОРСТВУ

Изјављујем да је докторска дисертација, под насловом

ПРИМЕНА ВОРОНОИ - ДЕЛОНЕ ТРИАНГУЛАЦИЈА И КАТАЛАНОВИХ ОБЈЕКАТА У ЗАШТИТИ ПОДАТАКА

која је одбрањена на Природно – математичком факултету Универзитета у Нишу:

- резултат сопственог истраживачког рада;
- да ову дисертацију, ни у целини, нити у деловима, нисам пријављивао/ла на другим факултетима, нити универзитетима;
- да нисам повредио/ла ауторска права, нити злоупотребио/ла интелектуалну својину других лица.

Дозвољавам да се објаве моји лични подаци, који су у вези са ауторством и добијањем академског звања доктора наука, као што су име и презиме, година и место рођења и датум одбране рада, и то у каталогу Библиотеке, Дигиталном репозиторијуму Универзитета у Нишу, као и у публикацијама Универзитета у Нишу.

У Нишу, 05.07.2021

Потпис аутора дисертације:

Фарук Б. Селимовић

Фарук Б. Селимовић

ИЗЈАВА О ИСТОВЕТНОСТИ ЕЛЕКТРОНСКОГ И ШТАМПАНОГ ОБЛИКА ДОКТОРСКЕ ДИСЕРТАЦИЈЕ

Наслов дисертације:

ПРИМЕНА ВОРОНОИ - ДЕЛОНЕ ТРИАНГУЛАЦИЈА И КАТАЛАНОВИХ ОБЈЕКАТА У ЗАШТИТИ ПОДАТАКА

Изјављујем да је електронски облик моје докторске дисертације, коју сам предао/ла за уношење у **Дигитални репозиторијум Универзитета у Нишу,** истоветан штампаном облику.

У Нишу, 05.07.2021

Потпис аутора дисертације:

Фарук Б. Селимовић

Фарук Б. Селимовић

ИЗЈАВА О КОРИШЋЕЊУ

Овлашћујем Универзитетску библиотеку „Никола Тесла“ да у Дигитални репозиторијум Универзитета у Нишу унесе моју докторску дисертацију, под насловом:

ПРИМЕНА ВОРОНОИ - ДЕЛОНЕ ТРИАНГУЛАЦИЈА И КАТАЛАНОВИХ ОБЈЕКАТА У ЗАШТИТИ ПОДАТАКА

Дисертацију са свим прилозима предао/ла сам у електронском облику, погодном за трајно архивирање.

Моју докторску дисертацију, унету у Дигитални репозиторијум Универзитета у Нишу, могу користити сви који поштују одредбе садржане у одабраном типу лиценце Креативне заједнице (Creative Commons), за коју сам се одлучио/ла.

1. Ауторство **(CC BY)**
2. Ауторство – некомерцијално **(CC BY-NC)**
3. **Ауторство – некомерцијално – без прераде (CC BY-NC-ND)**
4. Ауторство – некомерцијално – делити под истим условима **(CC BY-NC-SA)**
5. Ауторство – без прераде **(CC BY-ND)**
6. Ауторство – делити под истим условима **(CC BY-SA)**

У Нишу, 05.07.2021

Потпис аутора дисертације:

Фарук Б. Селимовић

Фарук Б. Селимовић