UNIVERZA NA PRIMORSKEM
FAKULTETA ZA MATEMATIKO, NARAVOSLOVJE IN
INFORMACIJSKE TEHNOLOGIJE

ZAKLJUČNA NALOGA

ZAKLJUČNA NALOGA

EVKLIDSKA TRANSFORMACIJA RAZDALJE

DAVID PIŠČANEC

UNIVERZA NA PRIMORSKEM
FAKULTETA ZA MATEMATIKO, NARAVOSLOVJE IN
INFORMACIJSKE TEHNOLOGIJE

Zaključna naloga

Evklidska transformacija razdalje

(Euclidean distance transform)

Ime in priimek: David Piščanec
Študijski program: Računalništvo in informatika
Mentor: doc. dr. Peter Rogelj
Somentor: izr. prof. dr. Janez Žibert

Koper, september 2013

Ključna dokumentacijska informacija

Ime in PRIIMEK: David PIŠČANEC

Naslov zaključne naloge: Evklidska transformacija razdalje

Kraj: Koper

Leto: 2013

Število listov: 40 Število slik: 31 Število tabel: 5

Število prilog: Število strani prilog: Število referenc: 9

Mentor: doc. dr. Peter Rogelj

Somentor: izr. prof. dr. Janez Žibert

UDK:

Ključne besede: Transformacija razdalje, Evklidska metrika, segmentacija, Voronojski diagram

Math. Subj. Class. (2010):

Izvleček:

Transformacija razdalje je postopek, ki pripomore k reševanju številnih problemov s področja digitalne obdelave slik. Z njim izračunavamo razdalje od točk zanimanja v okolici objekta do njim najbližjih točk objektov na sliki. Posredno s tem merimo razdalje med objekti, ki so potrebne predvsem pri najrazličnejšem primerjanju objektov, pa tudi pri nekaterih postopkih interpolacije, filtriranja, kompresije in podobno. Transformacije razdalje lahko temeljijo na različnih metrikah, med katerimi izpostavljamo Evklidsko metriko, ki kot najbolj naravna predstavlja absolutno najkrajšo razdaljo med točkama. Obstaja več različnih algoritmov za izračun transformacije razdalj, težava Evklidske transformacije razdalje pa je, da večina klasičnih algoritmov vrne le približen rezultat. Če želimo točne rezultate, moramo uporabiti algoritme, ki so računsko zahtevnejši. V tej zaključni nalogi se posvečamo dvema aktualnima in zelo različnima algoritmoma, kjer je prvi enostavnejši in poda približen rezultat, drugi pa je zahtevnejši, točen in z linearno časovno zahtevnostjo. Oba postopka smo realizirali v programskem jeziku C++ in ju med seboj primerjali. Evklidsko transformacijo razdalje smo nato ilustrirali

na praktičnem problemu s področja medicine - primerjavi rezultatov ročne segmentacije slik. Gre za segmentacijo področja tumorja pred radioterapevtskim posegom, kjer je za uspešnost zdravljenja ključna pravilna določitev robov tumorja. Vpliv razlik v segmentaciji na okoliška zdrava tkiva aproksimiramo z razliko razdalj od točke zanimanja do segmentiranih področij tumorja, ki jih izračunamo prav z Evklidsko transformacijo razdalje.

Key words documentation

Name and SURNAME: David PIŠČANEC

Title of final project paper: Euclidean distance transform

Place: Koper

Year: 2013

Number of pages: 40 Number of figures: 31 Number of tables: 5

Number of appendices: Number of appendix pages: Number of references: 9

Mentor: doc. dr. Peter Rogelj

Co-Mentor: izr. prof. dr. Janez Žibert

UDC:

Keywords: Distance transform, Euclidean metric, segmentation, Voronoi diagram

Math. Subj. Class. (2010):

Abstract:
The Distance Transform is a family of processes ancillary in solving Digital Image Processing problems. A Distance Transform process computes distances spanning Feature points with their nearest point on an observable object. The information obtained therein measures inter-object distance, required principally by object comparison problems. Additionally, the Distance Transform finds use in several processes of: Filtering, interpolation, compression, etc. Several metrics may serve as the basis of a transform, of which the Euclidean metric receives the most focus, due to provision of a shortest distance measure in the common space equipped with Euclidean properties. Several algorithms are available for Distance Transform computation. Unfortunately many classical algoritms, applied to the Euclidean metric, provide only approximate results. If an exact result is required, algorithms bearing a higher computational expense must be used. This final project treats a pair of Distance Transform algorithms. One is simpler and provides only an approximate result. The other trades simplicity for exactness - An asymptotical run time in grade of O(n) is achieved. The two algorithms are here implemented in the C++ programming language, and comparatively examined

afterwards. Results of an Euclidean Distance Transform are then illustrated on a practical medical problem - Comparison of expert image segmentation. The segmentation chart is sourced from the medical image of tumour in pre-op. Correct delineation of the tumour tissue boundary is essential for a successful clinical treatment. The degree of segmentation disagreement, as reflecting on the surrounding tissue, may be aproximated at a point of interest, by the calculation of an Euclidean distance transform against the segmented tumour surface.

Kazalo

1 Uvod **1**

2 Uporabni del **4**

2.1 Lastnosti Razdalje 12

3 Dvoje Algoritmov **16**

4 Zaključek **39**

Literatura **40**

Seznam tabel

3.1 Izvleček Borgefors et al. Algoritma Številka 2 19
3.2 Maurer et al. Procedura **ComputeFT** 26
3.3 Naivna Implementacija 1-Indeksiranih Polj v C 28
3.4 Primer Prehoda Polja 29
3.5 Rezultati 31

Seznam slik

1.1 Primeri Uporabe 1
1.2 Primeri Uporabe - Nadaljevanje 2

2.1 Digitalizacija in mreže 5
2.2 Rob mreže 6
2.3 Podatki algoritma 7
2.4 Potek skeniranja 8
2.5 Anizotropija po dekalibraciji 8
2.6 Bisekcija 10
2.7 Poprava mreženja 11
2.8 Druga lastnost 14

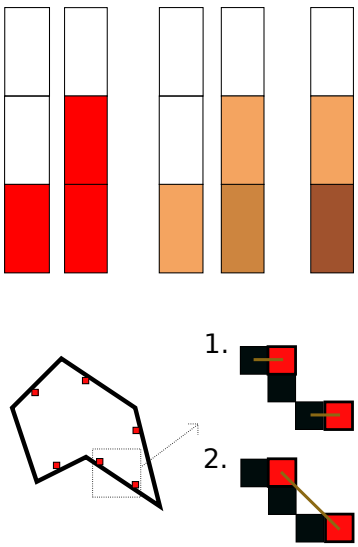
3.1 Propagacija Razdalj v Eni Dimenziji 16
3.2 Propagacija Razdalj v Treh Dimenzijah 18
3.3 Propagacija Razdalj v Eni Dimenziji: Prenovljeno 19
3.4 Propagacija Med Rezinami 20
3.5 Voronojski Diagram 22
3.6 Vrstica Delne Konstrukcije 23
3.7 Slapovni Model 23
3.8 Vrstica - Dodatno 24
3.9 Vrstica v Eni Dimenziji 24
3.10 Dimenzijski Vrtež v Ravnino 25
3.11 Števec 27
3.12 Pogled od Spodaj, s Strani 30
3.13 Pogled od Spredaj 30
3.14 Izvzetek 30
3.15 Rezultati: Vizualizacija 34
3.16 Rezultati: Prirastki 34
3.17 Izbrana regija segmentacije 35
3.18 Absolutna diferenca regije 36
3.19 Krajišči na poligonskem objektu 36
3.20 Bližina dveh predmetov 37

3.21 Negirana diferenca regije 38

1 Uvod

Na področju digitalne obdelave slik obstaja množica problemov, k rešitvi katerih lahko bistveno pripomore razpoložljivost postopkov zmožnih izračuna razdalj med elementi različnih tipov. Algoritmi, na voljo za izvedbo takih postopkov, se v veči meri razlikujejo po tem, kako se zakljukajo v variaciji prijazne dele interpretacije zgoraj. Elementi so lahko denimo, za procesiranje na voljo v različnih prostorskih razporeditvah. Naključno raztreseni ali z določeno stopnjo strukture, katere najpomembnejša je taka mrežne oblike. Definicije razdalje rezultat določajo z omogočanjem izbire različnih metrik. Predmet obravnave je tukaj Evklidska transformacija razdalje, kar botruje fokusu specifično na Evklidsko metriko, vendar so nekateri algoritmi sposobni izračune drugih metrik pretvoriti v aproksimacijo Evklidske. Točke, razdalja med katerimi je predmet izračunov, je mogoče dojeti kot ločene diskretne, ali jih grupirati kot volumne, in tako doseči prostorsko interpretacijo rezultatov. Prostorski problemi zahtevajo algoritme, ki omogočajo razporeditve v treh in višjih dimenzijah.

Evklidska transformacija razdalje je uporabna pri različnih operacijah na slikah, denimo [2] [3] [8]: interpolaciji, filtriranju, segmentaciji, registraciji, kompresiji, skeletonizaciji, grozdenju (Clustering), izračunu premera objektov.

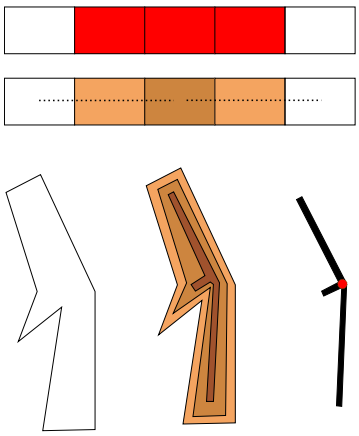


Slika 1.1: Primeri Uporabe

Slika 1.1 prikazuje dva od primerov uporabe Evklidske transformacije razdalje. Prvi

je interpolacija objektov. Prvi par trovrstičnih stolpcev prikazuje prostorsko pokritje, ki ustreza dvema objektoma (Rdeče), stičočima se z praznim prostorom (Belo). Od teh, je drugi objekt dobljen z raztegom prvega. Drugi par z barvami predstavi rezultat nad njima pognane Evklidske transformacije. Zadnji stolpec kot rezultat prikazuje interpolacijo teh razdalj. Vidno je, da taka interpolacija poda informacijo o spremembi. Če se drugi objekt razteza prek meja prvega, bo v njega presežnih točkah kot rezultat neničelna razdalja. Zmes vrednosti razdalj prvega in drugega merjenega objekta poda tudi informacijo o tem, kako "globoko" po raztegu ležijo točke (Na sliki temnejši odtenek rjave).

Preostanek slike podaja primer registracije. Problem, reševan z algoritmi registracije je transformacija množice podatkovnih točk, ki se karnajtesneje prilegajo obliki objekta - modela. Tipični kriterij po katerem je merjena stopnja ujemanja je višina seštevka kvadrata razdalj do modela. Slika prikazuje dva pogoja, katera se pri tem kriteriju lahko minimizira. Prvi pogoj je razdalja podatkovnih točk do krajišča modela, dosežen ko postane Evklidska transformacija (Merjeno od krajišča) na kraju podatkovne točke, vrednosti nič. Drugi pogoj je medsebojna razdalja podatkovnih točk. Ko so po izpolnitvi prvega pogoja vse točke razporejene po krajišču, preostane samo, da se jih razporedi optimalno, kar je ločen problem zunaj namena Evklidske transformacije.



Slika 1.2: Primeri Uporabe - Nadaljevanje

Slika 1.2 prikazuje dva dodatna primera: Izračun premera in skeletonizacijo. Izračun premera uporablja dejstvo, da je iskanje najdaljše razdalje dualen problen iskanju najkrajše. Določene algoritme se za ta namen da prilagoditi, na sliki je prikazan izračun premera v dveh dimenzijah. Od točke, katere razdalja do krajišč je največja, meritev poteka v obeh smereh. Zadnji preostali primer prikaže postopek skeletonizacije. Iregularno oblikovan objekt je sloj za slojem olupljen, z odstranitvijo točk po naraščajočih vrednostih Evklidske transformacije razdalje originalnega objekta. Od-

stranitev se konča ko za vsako še ohranjeno točko velja, da bi nje odstranitev povzročila razpad oblike na več delov. Iz tako dobljene oblike lahko, kot v shematiki prikazani na koncu, izvlečemo strukturo v obliki sklepov *oz.* tečajev, in večih izrastkov.

Temu uvodu sledeči razdelki poročila predstavijo preostanek vsebine. Najprej je Evklidska transformacija razdalje podrobneje definirana. Sledi za interpretacijo že zajetih vhodnih podatkov pomemben koncept obrobja. Potreba po uporabi učinkovitih podatkovnih struktur za opis obrobja vodi do predstavitve mrež, mreženja. Dlje časa je v uporabi primer naprave za razpoznavo nepravilnosti na objektih. Skozenj so predstavljeni koncepti anizotropije, kompenzacije mrežnih nepravilnosti za popravo. V kasnejšem poglavju so opisane dodatne lastnosti Evklidske razdalje. Te postanejo relevantne pri opisu enega od dvojice implementiranih algoritmov. Po teh prerekvizitnih lastnostih je algoritmoma predstavljeno delovanje, opisane so tudi nju implementacijske posebnosti.

2 Uporabni del

Evklidska transformacija razdalje izračuna razdaljo med ospredjem in ozadjem (ti. "Foreground", "Background") določene razporeditve prostorskih elementov. Sklicujoč se na [4], formula:

$$FT(u) = \arg\min_{v \in Bd_I} \|u - v\|$$

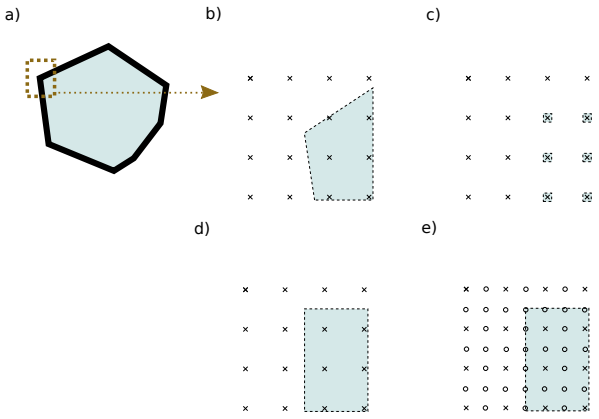
koncizno zapisano v popolnosti določa Evklidsko transformacijo. To je - v kolikor so definirani obrobje stika ospredja z ozadjem Bd_I, in operator metrike dolžine vektorja $\|x\|$. Od teh so različne možne definicije obrobja najbolj zaslužne za razlike v interpretaciji. Pokaže se, kot bo predstavljeno v nadaljevanju, da je moč različne funkcije razdalje v algoritmih uporabiti v poenotenem sistemu. Funkcije razdalje so v specifičnem algoritmu implementiranem za predstavitev podvržene dvem dodatnim lastnostim. Za razliko od generičnosti pri uporabi metrik, različne od definicij obrobja zahtevajo prilagoditev algoritmov. Predstavljene bodo definicije obrobij kot v [2], in njih lastnosti. Variabilnost je v sistem vnešena zaradi potreb po digitalizaciji problema. Denimo: Polja indeksirana z eno ali večimi dimenzijami je v klasičnih računalniških sistemih preprosteje manipulirati kot analitične enačbe prostornin (Naj prisotnost/neprisotnost vrednost polja v točki predstavlja pokritje objekta. Naj po drugi strani pokritje predstavlja notranjost zaprtega grafa parametrične funkcije v prostoru).

Geometrično obrobje karakterizira naslednji izraz:

$$Bd_I^G = \{x \in \mathbb{R}^k : \Delta(x, F_I) = \Delta(x, B_I)\}$$

V njem spremenljivka x teče preko celotnega n-dimenzionalnega prostora $\mathbb{R}^k$. Definicijsko območje F_I in B_I so točke diskretne mreže v prostoru, po kriteriju ospredja oziroma ozadja razdeljene na dve podmnožici. Kljub temu da je eden od členov v izrazih pogoja enakosti član te mreže, je rezultat izraza tej disjunkten (In sicer *vedno* - Nadaljevanje pri opisu slike 2.1). Digitalno obrobje bo uporabljeno v obliki ki bo v nadaljevanju opisana v razdelku o implementiranem približnem algoritmu Evklidske transformacije razdalje iz članka [3]. V tej obliki je obrobje diskretna anizotropna mreža. Ujema se torej z definicijskim območjem ospredja in ozadja. Ustreza zaradi tega, ker bosta z isto "mrežno" abstrakcijo implementirana oba koncepta.

Kot prikazano na sliki 2.1 ***a*** , naj obstaja objekt. Slika ***b*** nato predstavlja del objekta, ki bo opazovan. Preko sledi objekta je pozicionirana *ti.* digitalna mreža.



Slika 2.1: Digitalizacija in mreže

Pomemben je potek roba glede na točke v mreži, preostala površina se nato razteza v smeri navzdol in desno. Če na obliko ne postavljamo posebnih omejitev je razvidno, da je rob objekta v splošnem geometrijski: Kakorkoli že na gosto bi takšno mrežo sestavili in prilagodili nje postavitev, vedno obstaja taka prostorska konfiguracija, katera ostane v nepolnosti pokrita z nje točkami.

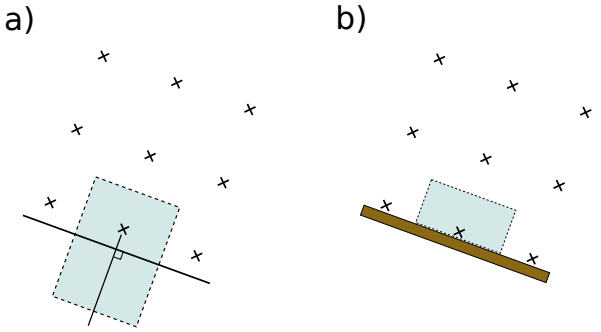
Ako se sprijaznimo z to limitacijo (Če drugega ne - zato ker so algoritmi Evklidske transformacije razdalje ki so predmet obravnave, zasnovani za delovanje v takem digitalnem načinu), so objekti digitalizirani z izgubo informacije o obliki. Na sliki ***c*** je prikazana digitalizacija istega območja kot na sliki ***b*** . Informacija ki ostane, je prisotnost ospredja v posameznih točkah. Naj bo polje v $\mathbb{R}^k$, končnih velikosti $\{n_d : 0 \leq d{<}k\}$. Mogoče je izračunati, da je število mogočih edinstvenih zajemkov v tem polju zgolj končno število, in sicer:

$$\text{štZajemkov} = \prod_{d=0}^{d=k} n_d$$

V primeru, da je uporabljeno trivialno mapiranje objektov v mrežo, v katerem je točka označena kot v ospredju če je objekt prisoten na njej istoležnih koordinatah in objekt ni prisoten v nobeni drugi točki, je s tem opravljen izogib omejitvi izgube podatkov. Pogoj takega tipa je zelo umetno postavljen: V splošnem je torej napravljena razdelitev na dva razreda objektov: Maloštevilčen razred ki ustreza trivialnemu mapiranju, in drugi večinski razred nepredstavljivih. Če obstaja možnost mapiranja tega zaviselega preostanka (V trenutnem primeru bi smiselna rešitev bila uporaba algoritma Evklidske transformacije razdalje, delujočega na z funkcijo $f : \mathbb{R}^k \to \{ospredje, ozadje\}$ analitično podano definicijo objekta), to trivialno mapiranje ni posebno pomanjkljivo. Nasprotno; Izkaže se, da bo izrabljena udobnost predstavitve mrežne strukture v podatkovnem modelu implementacije. Predhodno razmišljanje vodi v predstavitev naslednjega argumenta: V Fiziki in Kinematiki je uporabljan koncept masne točke *oz.* masnega

delca. Lastnost masnega delca je, da ne poseduje prostorskega razteza. Primer sistema v katerem so uporabljivi masni delci je določitev kinematičnih vrednosti rakete, leteče v medplanetarnem prostoru, oddaljene od drugih nebesnih teles. Po Newtonovih zakonih univerzalne gravitacije, masna točka privlači drugo masno točko z narastkom sile premo sorazmernim z njunimi masami, in upadom kvadratno sorazmernim z inverzom kvadrata njune razdalje. Inverzno kvadraten upad je dovolj strm, da je pri izračunih možno zavreči prostorske lastnosti sodelujočih objektov, in jih ohraniti zgolj kot točke z primerno prilagojenimi masami. Primer aplikacije Evklidske transformacije razdalje je inicializacija polja z vrednostjo ospredja na točkah ustrezajočih lokacijam planetov, in ozadja na vseh ostalih. Ko je algoritem pognan na tem polju, je dobljen rezultat, vzorčen na točki na kateri se nahaja raketa, razdalja do najbližjega planeta.

Na slikah ***d*** in ***e*** je pravtako prikazana digitalizacija območja s slike ***b*** . In sicer je, v nasprotje prejšnji masnotočkovni interpretaciji, uporabljena volumetrična. Prisotnost ospredja v posamezni točki mreže je smatrana kot prisotnost objekta v volumnu, raztezajočem se od vkjučno centra (Mesta ki se v prostoru objekta prilega točki mreže), do *roba* razteza. Rob je definiran kot množica vseh točk v n-dimenzionalneh prostoru v katerem je postavljena mreža, za katere velja da so po metriki razdalje bližje centru točke v obravnavi, kot centru katere koli druge točke mreže. Poseben primer se pojavi pri mrežah končnega razsega. Točke na samih mejah končne mreže (Torej krajišča, in njih vezne površine) v "zunanji" smeri namreč ne premorejo soseščine.



Slika 2.2: Rob mreže

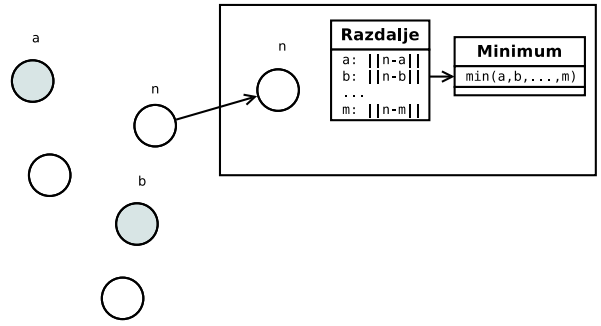
Prikazan na sliki 2.2 ***a*** , je volumen točke na meji mreže. V točki je konstruirana premica pravokotna na robno ploskev, katero v tem dvodimenzionalnem primeru predstavi premica. Po kriteriju bližine centra se volumen zagotovo razteza vsaj po celotni površini te premice. Vsaka točka te premice je namreč pravokotna na robno ploskev. Ker je točka na meji mreže del robne ploskve mreže, in je pravokotnica konstruirana skoznjo, je rezultat pravokotne projekcije točke na ploskev ravno ta mejna točka. Iz algebre je znano, da je pravokotna projekcija vektorja na podprostor, temu vektorju najbližja točka podprostora. Torej je del volumna izbrane točke, in hkrati ni del vo-

lumna nobene druge (Kar se lahko zgodi na meji dveh volumnov). Dejanski raztez je vizualno predstavljen na sliki ***a*** kot razlivajoč se navzven iz premice na sredini, do polovice razdalje do obeh ostalih robnih točk na vsaki strani. Več o razdelitvi prostora, ki jo povzroči kriterij enakomernosti oddaljenosti, v razdelku o dodatnih lastnostih Evklidske razdalje. Aplikabilne so ugotovitve o regijah Voronojskega diagrama dveh točk (Slika 2.8).

Da v namen dela z mrežami končne velikosti, volumni iz brezkončnih postanejo omejeni z mejami razsega mreže, je uporabljena konstrukcija kot na sliki ***b*** . Namen je vnesti omejitve, dodane so v obliki dodatnega sloja točk, po obliki ležečega preko celosti robne ploskve, na sliki prikazano z rjavo barvo. Definiciji je na konec dodan sledeč popravek: "... kot centru katere koli druge točke mreže, *ali katere koli točke omejitvenega sloja*". Volumen je torej omejen z mejami razsega mreže. Ugodni lastnosti definicije: Volumni posameznih mrežnih točk so vedno paroma disjunktni, in vse točke prostora so del vsaj enega volumna.

Slike so predstavljene z diskretno množico točk. Za potrebe indeksiranja po dimenziji in po vrsti je posamezno točko moč predstaviti kot člana kartezijskega produkta $C = \{x_0^1, \ldots, x_{n_1-1}^1\} \times \cdots \times \{x_0^k, \ldots, x_{n_1-1}^k\}$. Tako imenovan član je n-terica $(i_0, \ldots, i_k) \in \mathbb{R}^k$ realnih števil.

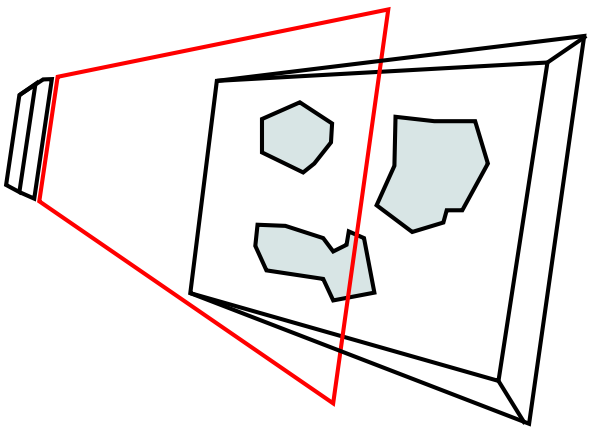
Dve pogosti metodi indeksiranja implementira naslednji postopek. Izbrano dimenzijo n indeksira n. element te n-terice, pri čemer ostali ohranjajo vrednost: $(\ldots, i_n, \ldots)$. Vrsto (S čimer je na tem mestu mišljena premica v k-dimenzionalnem prostoru, paralelna $0 < n < k$ vektorju standardne baze $\mathbb{R}^k$) prav tako indeksira n. element n-terice. Pri tem je za element vrste R_n^k, njegov naslednjik $R_{n+1}^k = (\ldots, i_{n+1}, \ldots)$.



Slika 2.3: Podatki algoritma

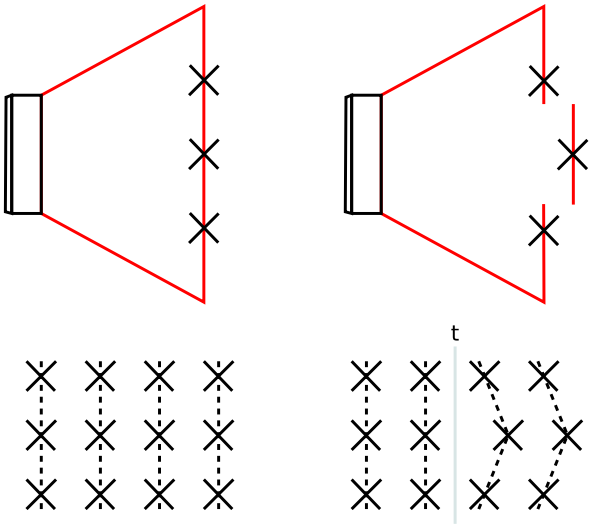
Pri rešitvah, kjer Evklidska transformacija razdalje pomaga modelirati fizičen sistem, je ugodno postaviti predpostavko da n-terica naslavlja prostorsko točko v enotah domene problema. (a, b, c) denimo pomeni točko dolžine a, širine b, višine c, v centimetrih, kakor odmerjeno po baznih vektorjih nepomičnega opazovališča nastavljenega v prostoru. Alternativna interpretacija je denimo a kot zaporedna številka rezine, b in

c kot koordinati na ploskvi.



Slika 2.4: Potek skeniranja

Implementacija ki je zmožna izrabiti zgoraj omenjeno interpretacijo, je naprava namenjena razpoznavi površinskih nepravilnosti na objektih, kot na sliki 2.4. Napravo sestavljata pomični trak in optični čitalec. V postopku je posamezna rezina tečena pod odčitnim območjem. Tako delovanje podatke zajame postopno, vrsto po vrsto. To je v kontrast, na primer, uporabi fotografskega svetlobnega senzorja, ki v enem koraku zajame celotno ploskev.



Slika 2.5: Anizotropija po dekalibraciji

Primer je izbran z namenom prikaza koncepta anizotropije slik, predstavljajočih vhodne podatke algoritma. Na sliki 2.5 sta skenirna glava in odtis prikazana kot na sliki 2.4. Z križci so označena mesta kjer, pri skeniranju posamezne vrste, senzor zajame podatke. Levi vzorec predstavlja zajem pri pravilnem delovanju sistema. Tako zajemki

posameznih, kot sekvence zajetih vrst, so razporejeni v regularno mrežo. Desni vzorec predstavlja zajem pri pojavi določenega defekta. Naj do časa t postopek zajem poteka rutinsko, brez težav. Nato naj pri času t skenirna glava utrpi mehansko okvaro: Kot prikazano na diagramu, se odtis enega od vzorcev vrste zamakne v smeri osi gibanja glave. Mreža zajemnih mest, ki jo predstavlja razporeditev zajemkov, zdaj na mestih okvare izraža iregularnost - anizotropijo.

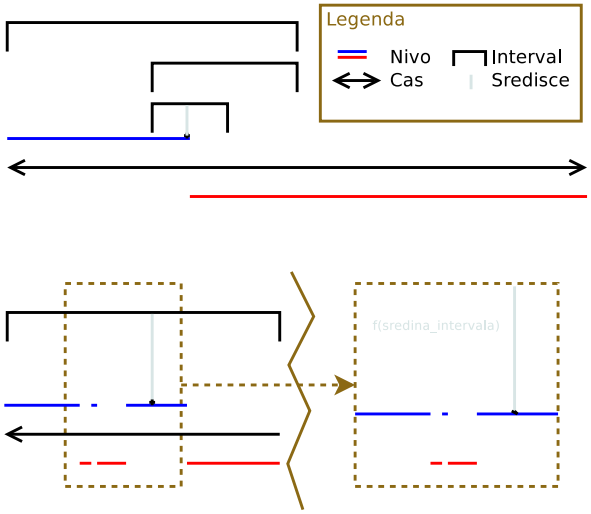
Ako je napaka dejansko opažena in detektirana, za razliko od denimo nadaljevanja procesiranja, z subtilno napačnimi podatki (V sistemu implementiranem kot na tukaj predstavljen način namreč vsak možen odčitek predstavlja določeno prostorsko konfiguracijo postopku izpostavljenih objektov. Redundance pri zajemu ni, tako tudi ne odvisnosti med posameznimi zajemki. Brez dodatnega domenskega znanja zajemkov tako ne moremo izločiti na osnovi validiranja vhodnih podatkov), tedaj obstaja možnost pretečenja sistema skozi diagnostičen postopek. V primeru na sliki 2.5 je problem mehanske narave (Nekak ekvivalent dekalibraciji senzorja). Probleme te narave je mogoče analizirati, pridobljeno znanje je uporabljivo v namen modeliranja modalitete defekta. V nakazanem primeru je mogoče, z nekaj predpostavljanja glede načina poškodbe skenirne glave, izpeljati dejstvo, da je poškodovan zgolj del podatkov: Od znane predhodne točke kjer je sistem preverljivo deloval pravilno, lahko sklepamo da je čas, t, nastanka napake v zamejenem intervalu. t lahko nato izoliramo z uporabo različnih postopkov, denimo: Z ponovnim predvajanjem zajema (Sampling replay) od znane predhodne točke, z bisekcijo skeniranja koordinat.

Od teh ima ponovno predvajanje prednost v tem, da z dodatno redundanco omogoča sprotno preverjenje zajema do točke napake.

Bisekcija, če si predstavljamo funkcijo $f : cas \rightarrow nivo$ z dvema nivojema vrednosti (Opravilna glava, okvarjena glava), lahko deluje, v kolikor velja da obstaja os, ki sekajoč številsko premico tej razdeli vrednosti po nivoju na čisti podmnožici. Taka razdelitev ustreza grafu zvezne funkcije, ki seka abscisno os v eni točki. Razdelitev je mogoče dojeti kot časovno koherenco defektivnostnega stanja glave. Če to ne drži, v primeru nekoherence, so invariante zahtevane za pravilno delovanje bisekcije kršene:

Na sliki 2.6 sta prikazani dve instanci scenarija okvare. Gornjega generira koherentni tip. Z intervali so prikazane razdelitve, izvršene pri pogonu bisekcije. Interval izbran v vsakem sledečem si koraku, je izbran po kriteriju nivoja. (Na sliki rdeče: Izberi levi interval; Modro: Izberi desni interval) Spodnji primer generira nekoherentni tip. Vzorčenje vrednosti $f(sredina\_intervala)$ bi definicijsko območje razdelilo na dva dela. Glede na nivo na mestu vzorčenja bi bil za naslednji korak izbran desni pod-interval, tako puščoč mesta obeh vrednosti na obeh straneh razmejitve.

Ne glede na to katerakoli od takih zaobidkov bi bil izbrana, se zmeraj ohranja originalni problem: Tehnične rešitve omilijo problem. Izkaže se, da študija Evklidske

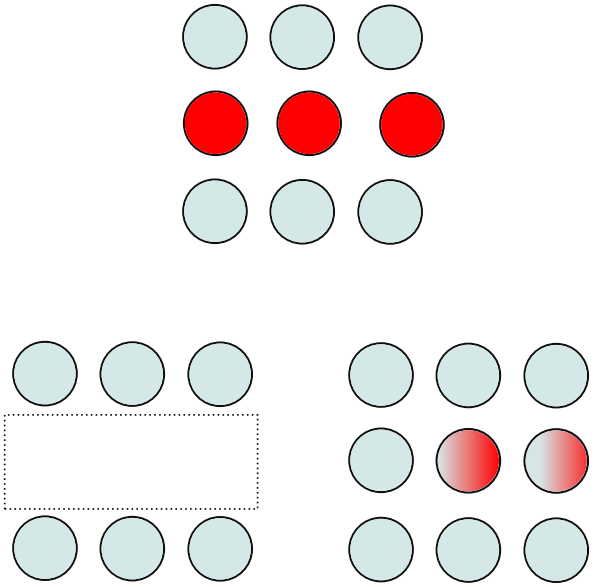


Slika 2.6: Bisekcija

Transformacije rezultira v ponudbi sistemske rešitve - algoritmi v uporabi postanejo predelani v namen rokovanja z anizotropijo. Če podatki ne morejo biti obnovljeni ali nadomeščeni, je taka algoritmična izboljšava način za izboljšavo obstoječih.

Opis na predhodnih straneh predstavi vhodne podatke algoritma kot n-terice v $\mathbb{R}^k$. Te je mogoče razumenti kot strukturo v obliki mreže. V primerih skladnih z zgoraj opisanim primerom čitalnika, se kot anizotropija izrazijo določene nepravilnosti. Bolj podrobno: Take nepravilnosti, pri katerih je mrežna struktura podvržena določeni transformaciji. Kompenzacije, ki nepravilnosti prilagodijo potrebam algoritmov Evklidske transformacije razdalje, delujejo na podatkih z učinkom ponovne vzpostavitve te mreže. Najprej bodi opisanih dvoje tehnik za obdelavo vhodnih podatkov, izpostavlujoč slabosti. Nato pa opis tehnike predstavljene v [2]. Mreža skrajšana za, oziroma obrezana okrog, določenega števila vrst (Odvisno od dimenzionalnosti problema, in dimenzije odrezka: Premic, ravnin, hiperravnin, ...), ohranja strukturo. Implementacija bi bila enakovredna preprostemu zavržku dela zajetih podatkov. Ker so ti podatki še zmeraj uporabljivi (V primeru uporabljenem v predhodnih razdelkih zgolj za določeno stopnjo zamaknjeni od idealne, kalibracijske konfiguracije), jih je zavreči in doseči suboptimalen rezultat škoda.

Slika 2.7 prikazuje primer zajema podatkov, ter dve instanci poprave mreženja. Vidi se, da je zamaknjena druga vrstica, na sliki obarvana rdeče. Prvi zajemek je poravnan, razlika z naslednjimi nato kumulativno narašča. V levem primeru poprave je vrstica zgolj izbrisana. V desnem so posamezni zajemki interpolirani glede na sosednje (Na sliki obarvano z gradientom). Pri interpolaciji so lahko upoštevani razni koncepti, kot je znanje o lokacijski koherenci objektov, ki jih zajemamo. Splošna oblika problema pa je taka, da so pridobljeni zajemki različnih lokacij neodvisni. Interpolacija



Slika 2.7: Poprava mreženja

je zanimiva, ker ohranja mrežno strukturo in tako omogoči uporabo podatkov zgolj z predhodnim predprocesiranjem, brez modifikacije uporabljenih algoritmov Evklidske transformacije, teoretično pa, kot je opisano predhodno, se ne razlikuje od zavržka podatkov. Koncepta mreže (Indeksirane z n-tericami) in koordinatne množice (Mapirane z indeksi n-teric položajne koordinate), sta strogo ločena. Mogoče pa je oba izraziti brez dupliciranja uporabljenih abstrakcij. Izrazek 2.3 (Remark 2.3) obravnavo konceptov poenoti s sledečim sistemom:

> Naj bo Δ metrike L_p. V izotropi sceni (Z razliko koordinatnimi dvojicami $N_j^i = x_(^i j+1) - x_j^i$ zmeraj enako fiksnemu številu σ), naj bo $\Delta(c,d) = \Delta^*(c,d)\sigma$. Za izotrope slike identifikacija C^* z C tako ne zahteva spremembe v definiciji funkcije razdalje, razen v končni multiplikaciji z σ. Za razliko od tega pa mora za anizotrope slike razdalja biti izvlečena iz števil N_i^d, podanih skupaj z sliko.
>
> $$\Delta^*(\langle c_i\rangle, \langle c_i\rangle) = \left(\sum_{i=1}^{k} |x_{c_i}^i - x_{d_i}^i|^p\right)^{1/p}$$
>
> Funkcija razdalje Δ mora biti dojeta kot subrutina, podana z gornjo formulo. V tej velja definicija: $|x_{c_i}^i - x_{d_i}^i| = \sum_{min\{c_i,d_i\}<j\leq max\{c_i,c_i\}} N_i^j$.

Pripomki glede števil N_i^d, pomožnih podatkov anizotropih slik: Števila se morejo razlikovati med vsakim zaporednim parom koordinat, pripomorejo torej $O(n)$ dodatnih podatkov. Kot razvidno iz gornje formule funkcije razdalje Δ^*, in nje sledeče definicije

$|x^i_{c_i} - x^i_{d_i}|$, priobčenje razdalje iz danih koordinat pri anizotropih slikah zahteva dodatni sumacijo preko vseh vmesnih indeksov (Po številu obstaja n teh). Določeni algoritmi razdaljo računajo $O(n)$ krat. Tak je v prejšnjem razdelku predstavljeni naivni vsak-z-vsakim (Na sliki 2.3). Temu algoritmu dodatne komputacije računanja razdalje, izključujoč rešitve kot npr. medpomnjenje rezultatov, časovno kompleksnost pokvarijo na $O(N^3)$. Razlika med direktnim indeksiranjem in indirektnim indeksiranjem se pokaže če so števila N^i_j odvisna of i in j - niso konstantna. Razlika med relativno in absolutno koordinatno diferenco: Če so vsote $|x^i_{c_i} - x^i_{d_i}|$, za vsako točko n-terice slike predizračunane v globalni koordinatni sistem. Predizračun ne razreši vsakega sistema učinkovito. V primeru skenirnega sistema objektov, okvarjenega po modelu anizotropije kot na sliki 2.5, je dovolj hraniti eno dodatno relativno diferenco, po odločitvenem pravilu naslednje enačbe.

$$\begin{cases} \text{čas} < \text{t} & \text{Normalna diferenca} \\ \text{čas} = \text{t} & \text{Diferenca upoštevajoč zamik senzorja} \\ \text{čas} > \text{t} & \text{Normalna diferenca} \end{cases}$$

Torej ena diferenca, ki stopi v efekt pri času t. Vse absolutne diference od časa t dalje pa se razlikujejo od izotropnih.

2.1 Lastnosti Razdalje

Algoritmi Evklidske transformacije razdalje, kot obravnavani tukaj, načeloma podpirajo, po zasnovi ali kot "bonus" po naključju, uporabo na problemih z različnimi interpretacijami razdalje oziroma metrike. Trivialni algoritem, kot na sliki 2.3, na izbiro metrike načeloma ni občutljiv. Ker najbližjo točko posameznemu zajemku določi tako, da izračuna metriko do vsake ostale točke posebej in izbere najnižjo, lahko to stori za vsako funkcijo $f : \cdots \to \mathbb{R}$. Seveda za večino veljavnih funkcij ne obstaja ugodna intuitivna interpretacija rezultatov. Na Voronojskih diagramih zasnovani algoritem linearne časovne kompleksnosti, ki bo predstavljen v nadaljevanju, zahteva da: Je funkcija razdalje metrika; Je metrika, ki podpira dve dodatni lastnosti. Ti dodatni lastnosti sta tu opisani.

Bodijo definirani naslednji objekti: Vsak $d = 1, \ldots, k$, vsaka vrsta R v $\mathbb{R}^k$ paralelna d-ti prostorski osi, in točki mreže v prostoru $u = (u_i)$ in $v = (v_i)$.

Prva dodatna lastnost: Če je $u_d = v_d$, $y \in R$ tak da $y_d = u_d$, in $\Delta(u, y) \leq \Delta(v, y)$, potem $\Delta(u, x) \leq \Delta(v, x)$ za vsak $x \in R$.

Druga dodatna lastnost: Če je $u_d < v_d$, nato obstaja $x_{\overline{uv}} \in \mathbb{R}$, tak da za vsak

element $x \in R$ velja: Če $x_d < x_{\overline{uv}}$, nato velja $\Delta(u,x) < \Delta(v,x)$; In če $x_d > x_{\overline{uv}}$, nato velja $\Delta(u,x) \geq \Delta(v,x)$.

Pred komentiranjem teh lastnosti razdalje bo predstavljen pomožen rezultat. Euler v objavku, v prevodu imenovanem "General formulas for the translation of arbitrary rigid bodies" [5], leta 1775 naslovi problem transformacije (Pri čemer ta vključuje rotacijo in translacijo) togih teles v prostoru. Povzeto je v prostor postavljen tridimenzionalni ortogonalni sistem baznih vektorjev. Koordinate telesa pred in po izvedeni transformaciji so izražene v tem sistemu. Intuitivno se pri togi spremembi telesa razdalje med vsakim posameznim parom izbranih točk ohranjajo. Zato je izbran pogoj invariantnost evklidske metrike za transformacijo (Oziroma *kvadrata* te metrike). Z dodatkom tega pogoja, je formule mogoče faktorirati v posebno obliko.

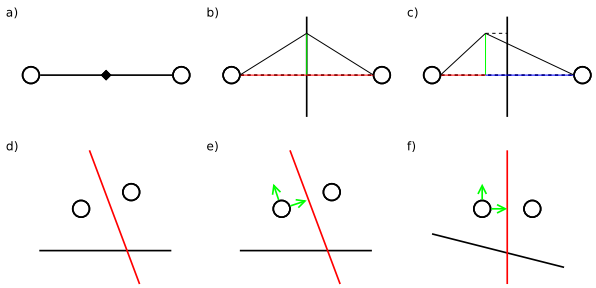
$$x = f + Fp + F'q + F''r$$
$$y = g + Gp + G'q + G''r$$
$$z = h + Hp + H'q + H''r$$

Če iz teh enakosti izvzamemo translacijo, torej $f = g = h = 0$, lahko dobljeni vzorec prepoznamo kot enačbo za množenje matrike dimenzij 3×3 z vektorjem: $A_{3\times 3}x$. V tej so števila $\{F, G, H\}$ konstante matrike, $\{p, q, r\}$ vhodni vektor, in $\{x, y, z\}$ rezultat. Kot razvidno iz Eulerjevega treatmaja, so posamezni stolpci enotske dolžine: $F^2 + G^2 + H^2 = 1$. Ker so tudi neodvisni, se da sklepati da je dobljena matrika ortogonalna. Če ortogonalno matriko interpretiramo kot ortonormirano bazo vektorskega prostora, potem posamezen stolpec te ustreza enemu od baznih vektorjev. Bistveno dejstvo je, da je bila matrika izpeljana iz predpostavke o ohranjanju razdalje. Rezultat: Rotacija (Izbira ortonormirane baze) prostora ohranja razdaljo.

Prva lastnost: V treh dimenzijah je ortogonalni komplement premice ravnina, katere normala je ta premica. V izbrani točki na premici, je konceptualni ustreznik ortogonalnemu komplementu množica vseh točk pravokotnih na premico v točki: Torej 2D rezina v prostoru. Naj bo R premica ki teče paralelno k x-osi, x koordinata točke na premici. Če $\{y, z\}$ predstavljata vektor v rezini na tej točki, tedaj (x, y, z) zapiše koordinate poljubne točke. (x, y, z) se da razstaviti na komponenti: $(x, 0, 0)$ v smeri premice, $(0, y, z)$ v smeri rezine. Skalarni produkt teh dveh komponent je vedno ničelen, če posamezne člene paroma pomnožimo, je v vsakem zmnožku vsaj en člen enak nič. Če bazna vektorja $\{y, z\}$ zavrtimo tako, da se denimo y poravna z smerjo (y, z), se da to poljubno točko (x, y, z) zapisati kot $(x, y', 0)$. Drži, da ima vsaka linearna preslikava na realnem prostoru lihe dimenzionalnosti eigenpodprostor, oziroma po Eulerjevem rotacijskem teoremu ima vsaka transformacija togega telesa rotacijsko os,

ki ob transformaciji ohranja položaj na njej ležečih točk. Pri vrtenju $\{y, z\}$ se v ravnini položaji ne ohranjajo. Os vrtenja mora tako ležati zunaj te ravnine; Vrtenje $\{y, z\}$ dosežemo z rotacijo okoli x-osi. Posledica je, da po transformaciji baznih vektorjev, točka zapisana $(x, y', 0)$ še vedno leži v isti rezini kot originalna. Zaradi ekvivalence rotacije in izbire ortonormirane baze, omenjene v razdelku o Eulerju, taka rotacija vedno obstaja. Če tak zavrten koordinatni sistem postavimo v točko x na premici R, je razvidno da daljici $\overline{(0,0,0)(x,0,0)}$ in $\overline{(x,0,0)(x,y',0)}$ tvorita pravokotni trikotnik. Stranica $\overline{(0,0,0)(x,0,0)}$ je skupna vsaki točki v rezini. Primerjava razdalj to dveh poljubnih točk v rezini tako zavisi le od dolžine druge stranice. Razdalje do različnih točk (Vsake z svojo različno rotacijo) so medsebojno primerljive, ker rotacija prostora ohranja razdaljo. Navedena odvisnost od le ene dolžine je jedro prve lastnosti. Če sta v isti rezini izbrani dve točki, je njih razdalja do poljubne točke na premici pravokotni na rezino, najnižja za točno določeno od teh.

Druga lastnost: Ta lastnost v grobem rečeno obravnava particioniranje prostora na dva dela. Kriterij za to uporabljen je enakomernost oddaljenosti. V prostor postavljeni dve točki ustvarita karakteristično razdelitev, ki bo analizirana tu. Razvidno bo, da ta kriterij povzroči particioniranje na dva dela. Ta dela sta drug od drugega ločena z površino ki je podprostor dimenzionalnosti ene manj kot celotni prostor.



Slika 2.8: Druga lastnost

Na sliki 2.8 a, b, c konstruirajo ločno površino, enakomerno oddaljeno od dveh točk.

Na a sta točki označeni z izvotljenima krogoma. Veže ju daljica, na kateri je označena središčna točka. Če poljubni točki A in B zvežemo z daljico $\overline{AB}$, je sredina $\frac{A+B}{2}$ te gotovo enako oddaljena od obeh točk, in tako del ločne površine.

Na b je izbrana točka na daljici, ki teče skozi središčno točko, in je pravokotna na vezno daljico. Vidno je da postaneta formirana dva pravokotna trikotnika. Kateta trikotnika obarvana rdeče je enaka ne glede na izbrano točko. Z obeh strani se namreč veže s središčno točko. Kateto obarvano zeleno si delita oba trikotnika, zato je pravtako enaka.

Na c je izbrana točka ležeča v preostanku izvornega prostora - Torej ležeča izven pravokotnice. Točka je z pomožno črto zvezana z njeno ortogonalno projekcijo na to

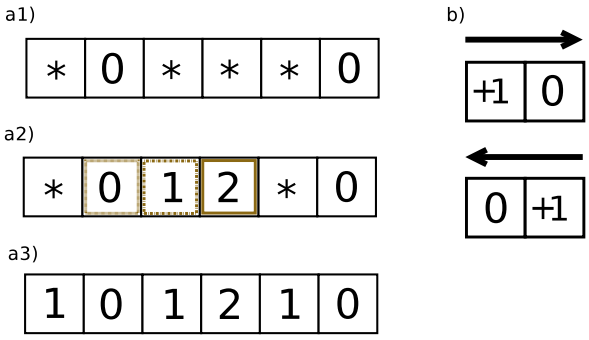
pravokotnico. Kateta obarvana zeleno je pri taki, poljubno v prostoru izbrani točki, enaka primeru projecirane točke na pravokotnici. Od ostalih katet (Na c obarvanih rdeče in modro), je ena daljša in druga krajša.

To velja v primeru dveh dimenzij. V treh dimenzijah ločna površina namesto premice postane ravnina. Utemeljitev teče na način analogen tistemu uporabljenemu pri prvi lastnosti, za zavrten koordinatni sistem postavljen "v točko x na premici R".

3 Dvoje Algoritmov

Za potrebe predstavitve sta bila implementirana dva algoritma Evklidske transformacije razdalje. Eden delujoč po principu propagacije razdalj, in drugi katerega v ozadju podpirajoči koncept je učinkovita konstrukcija (*oz.* vzorčenje, v nobeni točki procesa naslednji namreč ni manifestiran eksplicitno) Voronojskega diagrama na točkah premice paralelne enemu od baznih vektorjev prostora.

Algoritem propagacije razdalj je opisan v Borgefors et al. [3]. Sledena je njihova eskpozicija. Borgefors uporablja akronim *DT* - Transformacija razdalje. Kontrastirano z algoritmi Evklidske transformacije razdalje, *EDT*, so uporabljane druge metrike: *ti.* Chamfer, Manhattan in N-Neighbour. Implementirana varianta uporablja metriko, ki je *približek* Evklidski. D-Evklidske razdalje vsem točkam, razen v posebnih specifičnih prostorskih konfiguracijah merjenega objekta, priredijo točno Evklidsko razdaljo. Vendar tudi v najslabšem primeru velja, da razlika med pravo vrednostjo in dejansko izračunano ne presega nizke konstantne vrednosti.



Slika 3.1: Propagacija Razdalj v Eni Dimenziji

Slika 3.1 prikazuje koncept propagacije v eni dimenziji. Razširitev na različico delujočo v n. dimenzijah je enostavna, torej je predstavitev reprezentativna, kar se cilja aplikacije na objektnih scenah trodimenzionalnih tiče. ***a1*** prikazuje začetno stanje vrste (Enodimenzionalnega polja): Zvezdice v poljih označujejo prisotnost ospredja, ničle pa prisotnost ozadja. Ta ilustracija elementov algoritma se direktno prevede na podatkovno reprezentacijo uporabljivo pri izvedbi (Pri implementiranem Voronojskem algoritmu to ne zgodi, potrebnih je več podpornih podatkovnih struktur). Pripravljen je niz v dolžine enake številu polj. Elementi v_n so celoštevilskega tipa. Indeksi

niza, ki pripadajo poljem z zvezdico so inicializirani na vrednost neskončnosti, indeksi pripadajoči ničlam pa na ničle. ***a2*** prikazuje stanje, ki je iz začetnega dobljeno z pogonom treh korakov *ti.* navzprednjega prehoda. Navzprednji prehod je opravljen sledoč naslednji formuli:

$$\text{for } i = 2, 3, \ldots, M \qquad v_i := \text{minimum}(v_i, v_{i-1} + 1)$$

Na sliki ***a3*** je prikazano končno stanje, dobljeno po zaključku navzprednjega prehoda, in opravku sledečega vzvratnega prehoda. Vzvratni prehod je opravljen sledoč naslednji formuli:

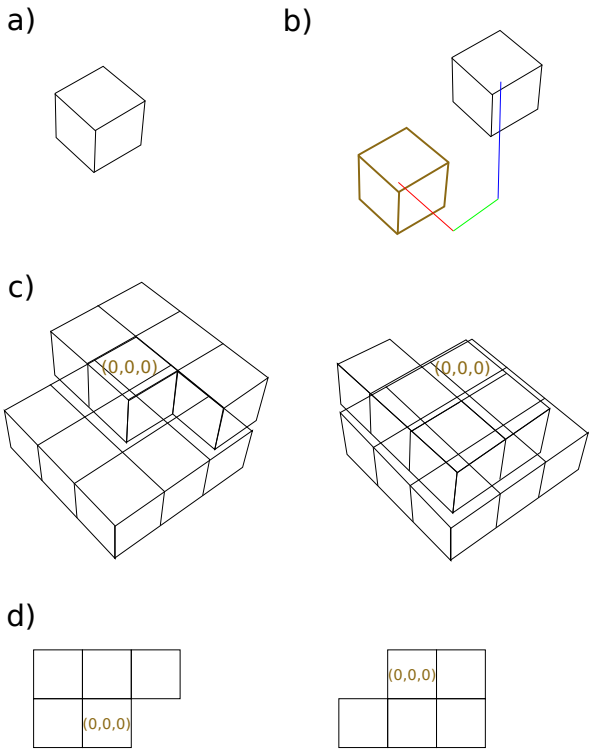
$$\text{for } i = M - 1, M - 2, \ldots, 1 \qquad v_i := \text{minimum}(v_i, v_{i-1} + 1)$$

Brez uporabe minimum, prehoda vsakemu elementu ospredja pripišeta razdaljo od najbližjega levega oziroma desnega elementa. Tak pripis se, po minim-iziranju, sklada z interpretacijo problema iskanja razdalje do najbližje točke ozadja, kot izborom najkrajše izmed vseh možnih poti propagacije. V eni dimenziji ima vsaka točka samo dva soseda. Točka ospredja je lahko najbližja ali levi ali desni točki ozadja, pri čemer je nabor točk levo in desno disjunkten. Obstaja tako le ena najbližja pot (Oziroma dve enakovredni). Ker navzprednji in vzvratni prehod preideta obe, bo najkrajša pot dobljena kot rezultat.

Na sliki ***b*** sta takoimenovani maski. Maske so način za opis - V celoti definirajo propagacijo razdalje. Za izbran algoritem, posebej sestavljene maske zahtevajo vsaka od različnih uporabljenih metrik, in vsaka izbrana dimenzionalnost pri kateri mora algoritem delovati. Posamezen element maske je maskvoksel. Obe maski Evklidske metrike dimenzionalnosti ena sestavljata dva maskvoksla. Uporaba mask teče po naslednjem postopku: Edini ničelni element maske je zamišljen kot lebdeč postavljen nad element niza trenutno v obravnavi. Vsakemu od ostalih elementov niza, ki je pokrit z maskvokslom, je odčitana vrednost. Te odčitane vrednosti, skupaj z trenutno vrednostjo ničelnemu ustrezajočega elementa, sestavljajo kandidatno množico. V kandidatni množici je najdena najnižja vrednost, ki postane nova vrednost ničelnemu ustrezajočega elementa niza. Maska je nato premaknjena na, in postopek ponovljen za naslednji element.

Propagacija razdalje z maskami je lokalna operacija. Na prireditev vrednosti elementa vplivajo zgolj elementi v soseščini. Mehanizem, ki omogoči izračun, prave, globalno veljavne razdalje je iteracija: Naslednji element je izbran tako, da v njega kandidatni množici sodeluje eden od predhodno obdelanih elementov.

Za obravnavo so bolj zanimivi primeri iz višjih dimenzij. Implementirani algoritem D-Evklidske razdalje deluje v treh, sledi opis delovanja kot primerljivo z referenčnim enodimenzionalnim zgoraj.



Slika 3.2: Propagacija Razdalj v Treh Dimenzijah

Na sliki 3.2 ***a*** maskvoksli šele upravičijo svoj naziv ("Volumetric Picture Element" - Volumenski Slikovni Element). Medtem ko v nižjih dimenzijah privzemajo ploskovno obliko, so njih volumni v treh kocke (V štirih in dalje pa hiperkocke). Številske vrednosti uporabljane v elementih maske zamenjajo vektorji, $m_i = (x, y, z)$ na položaju i, maske m. Ker maske postanejo trodimenzionalne, indeks $i \in \mathbb{R}^3$ teče preko vektorjev, namesto skalarjev. Ničelni element je ponovno en sam, vrednosti $m_{(0,0,0)} = (0, 0, 0)$. Element maske $i = (x, y, z)$ dobi vrednost $m_{(x,y,z)} = (|x|, |y|, |z|)$. Enačba Manhattanske norme ℓ_1 v treh dimenzijah je:

$$d_1(p, q) = \sum_{i=1}^{n} |p_i - q_i|$$

$$d_1\left((x, y, z), 0\right) = |x| + |y| + |z|$$

Razvidno je tako, da so vrednosti elementa maske pravzaprav členi Manhattanske razdalje do ničelnega elementa. Slika ***b*** nakaže kako je sešteta razdalja razstavljena po treh baznih vektorjih.

Slika ***c*** prikazuje maski, uporabljeni v navzprednjem in vzvratnem superprehodu. Kot v enodimenzionalnem primeru sta simetrični okoli ničelnega elementa. Prostorsko-vokselni sta samo ti dve, naslednji s slike ***d*** sta ploskovni. Poslednji sta predhodno predstavljeni enodimenzionalni maski slike 3.1 ***b*** . Uporabljen je bil izraz "superprehod". Algoritem zdaj zahteva več prehodov, v povzeti različici:

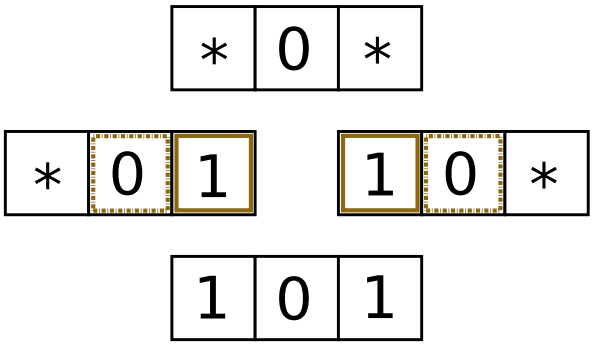
- Vrsta (Tip maske ***c*** , 3.1 ***b***): V posamezni vrsti se najprej maska pomika naprej, nato druga nazaj.
- Rezina (Tip maske ***d*** , 3.1 ***b***): Gornja sekvenca preide vrste v rezini, nato v obratni smeri.
- Prostor: Gornja sekvenca preide vsako rezino, nato v obratni smeri.

Nekaj birokracije glede poimenovanja prehodov in izbire mask, ki se ne pokaže za bistveno je izpuščene. Gornji oris postopka je dovolj za obravnavo sledeče potencialne šibkosti v ekspoziciji Borgefors:

Tabela 3.1: Izvleček Borgefors et al. Algoritma Številka 2

```
Forward pass:
for k1 := 2 step 1 until M1 do
for k2 := 2 step 1 until M2 do
...
for kN := 2 step 1 until MN do
```

Tabela 3.1 Indeksi kN so inicializirani na 2. Ta zapoznela inicializacije se zdi skladna z formulami enodimenzionalnega navzprednjega prehoda zgoraj. Posebne pozornosti takrat ni bila deležna.

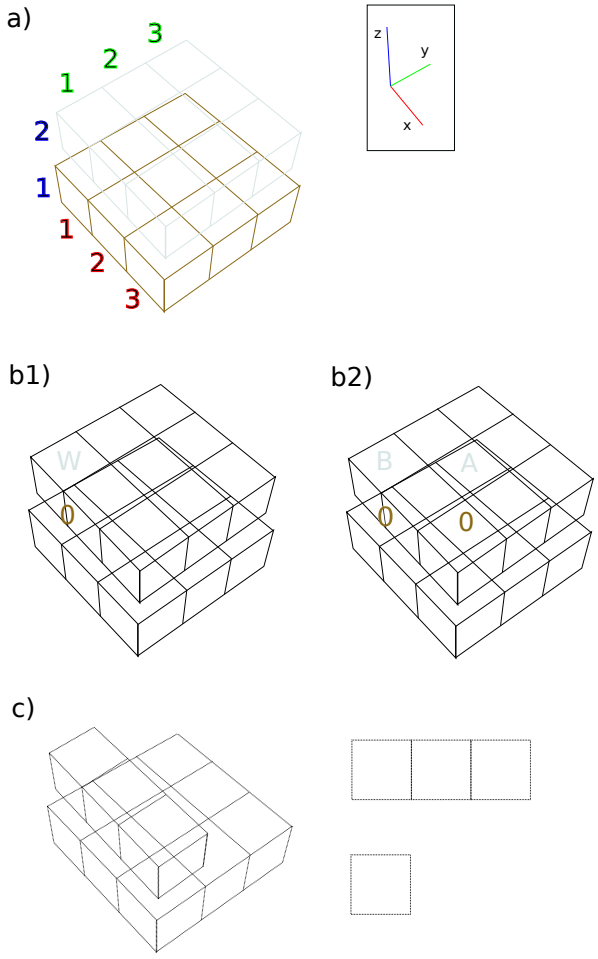


Slika 3.3: Propagacija Razdalj v Eni Dimenziji: Prenovljeno

Slika 3.3 je v pomoč pri vizualizaciji, čemu zapoznela inicializacija v eni dimenziji deluje pravilno. Predstavljen je degeneriran primer z tremi elementi. Trije elementi so ravno dovolj, da polje vsebuje dva roba z vrednostjo ospredja, hkrati z enim ničelnim elementom. Polje z večimi elementi pripelje do istega rezultata, dokler je ohranjen vzorec ene vrednosti ospredja na robu, ki ji sledi ničelen element. Vrednosti ostalih elementov na robne ne vplivajo. Vpliva ni zato, ker se razdalje propagirajo po določeni

poti propagacije. Robni elementi premorejo zgolj enega soseda, in tako zgolj eno pot. Med potekom posameznega prehoda se ob srečanju ničelnega elementa, vrednost v_i ($0 = \text{minimum}(0, v_{i-1} + 1)$) zagotovo postavi na nič. Akumulacija razdalje se tako resetira. Vrednost, ki jo po prehodu zavzame robni element, je razdalja od najbližjega ničelnega elementa. Po definiciji primera, je najbližji ničelni element v neposredni soseščini robnega. Ker do robov teče samo ena pot (Izmenično: Leva, desna), in prehoda propagirata samo po eni poti (Izmenično: Desni, levi), sledi, da na vsakega od robov vpliva samo eden od prehodov. Ker zapoznela inicializacija preskoči robni element v prehodu, ki nanj ne vpliva, se pravilnost rezultata ohranja.

Zapoznela inicializacija pravilno deluje tudi v dveh dimenzijah. V tem primeru so preskočene entitete robne vrstice elementov.



Slika 3.4: Propagacija Med Rezinami

Dve rezini sta obarvani na sliki 3.4 ***a*** , robna rjavo, njej sledeča modro. Po algoritmu tabele 3.1, se procesiranje prične v elementu $(2, 2, 2)$, preskočene so torej ena rezina, ena vrstica, in en element. Na sliki ***b1*** sta postavljena dva elementa. Ozadni element vrednosti nič na koordinatah $(1, 1, 1)$, in poljuben element W na koordinatah $(1, 1, 2)$.

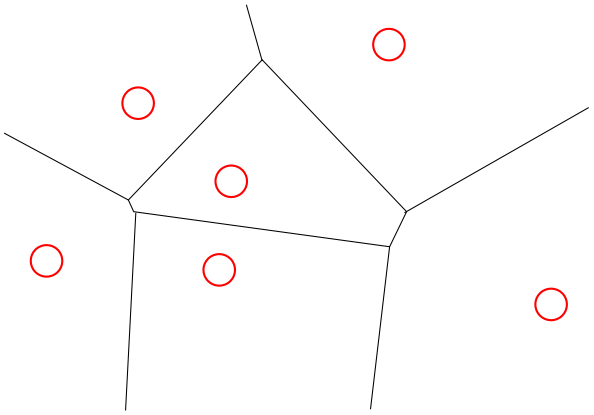
Naslednja propagacija, po poti od 0 do W, katera bi morala končati z vrednostjo razdalje rezultata $(0, 0, 1)$, se nikoli ne izvrši. Propagacija je v medrezinski smeri z pričetkom v prvi vrstici prve rezine. W bi z istim končnim efektom lahko zavzel vrednost kateregakoli elementa prve vrstice *druge rezine*. Edine maske, ki razdalje propagirajo v medrezinski smeri, so trodimenzionalne z slike 3.2 ***c*** . Element W je v neposrednem stiku z elementom 0. Premica, ki elementa veže, je najkrajša pot med elementoma, propagacija jo preide v enem koraku. Zaradi formule delovanja algoritma, propagirana razdalja z številom korakov strogo narašča (v_i je rezultat minimuma članov kandidatne množice. Članom, pridobljenim iz elementov maske, je razdalja povečana za vsaj 1). Trikotniška neenakost je tako ohranjena. Taka enokoračna propagacija pa se v podani konfiguraciji elementov nikoli ne izvrši. Slika ***c*** prikazuje entitete preskočene z zapoznelo inicializacijo v različnih dimenzijah. Enodimenzionalni primer je bil podrobneje razdelan zgoraj, preskočen robni element na rezultat ne vpliva. V dvodimenzionalnem primeru je preskočena vrstica. Če zamislimo:

- $1D$ preskakuje podprostor dimenzionalnosti 0 (Posamezne elemente), propagacijo nato vrši v dimenzionalnosti 1.
- $2D$ preskakuje podprostor dimenzionalnosti 1 (Vrstice), propagacijo nato vrši v dimenzionalnosti 2.

Kot nato razvidno s slike ***c*** , se med tema primeroma, in trodimenzionalnim primerom prikaže razlika. Tro- (In višje-) dimenzionalni primer je prvi, v katerem je po opravljenem preskoku, propagacija vršena preko dveh dimenzionalnisti (2-ploskve in 3-prostora). Upoštevajoč trodimenzionalno konstrukcijo s slike ***c*** , sta v osi z preskočeni dve vrstici zaporedoma. Ta dodaten korak preskoka ovira propagacijo. Pri obdelavi začetne druge rezine so (V oseh x, y) uporabljene ploskovne maske slike 3.2 ***c*** . Te razdalje ne propagirajo v smeri z osi. Primer pri katerem je bila na implementaciji algoritma odkrita zgoraj opisana šibkost prikazuje slika ***b1*** . Vidna sta dva ničelna elementa v prvi rezini. Točki, skozi katere je propagacija opazovana sta pravtako dve, v drugi rezini locirani takoj nad ničelnimi. Za ničlo na koordinatah $(2, 2, 1)$ najprej obstaja propagacijska pot v A, nato, pri prehodu druge rezine, pa iz A v B. Ta pot (V vrednostih razdalj: $(0, 0, 1) + (1, 1, 0) = (1, 1, 1)$)) ni najkrajša. V B namreč pripelje tudi pot razdalje $(0, 0, 1)$, ki teče med ničlo $(1, 1, 1)$ na eni strani, in B $(1, 1, 2)$ na drugi. Izključujoč pomote pri implementaciji testnih primerov, avtor menim da Borgeforsov algoritem v treh dimenzijah, za določene konfiguracije vhodnih podatkov ne najde najkrajše poti, ali poti sploh. Ker je algoritem približen to ni presenetljivo. Se pa modaliteta napak razlikuje od v članku predstavljene. Namreč, razlike v vrednosti med idealno Evklidsko razdaljo in dobljeno bi se morale poroditi, zaradi potrebe Evklidske razdalje po propagaciji po poti ustrezajoči premici, kjer pa aproksimacija propagira

po diskretni mreži. V namene implementacije je bil vpeljan naslednji zaobidek: Ker algoritem v eni in dveh dimenzijah deluje pravilno, in se težave pojavijo pri propagaciji v treh dimenzijah med rezinami zaradi začetnega preskoka, je bil prilagojen tako da prve rezine ne preskoči. Potreba po taki prilagoditvi je nesrečna, ker morajo biti pri testiranju elementov proti maski, v obdelavi prve rezine vpeljane dodatne preverbe (Preprečujoč indeksiranje zunaj meja polja). Za alternativen način bi bilo smiselno raziskati možnosti prilagoditve mask tipa slike 3.2 $\boldsymbol{d}$. Če so "nadgrajene" na tip $\boldsymbol{c}$, postanejo razdaljo sposobne propagirati v treh dimenzijah.

Opisan naslednji je na Voronojskih diagramih zasnovan algoritem. Od pregledanih psevdokod je še karnajdirektnejša tista, opisana v Maurer et al. [1]. Citirani bosta njihovi tabeli procedur **ComputeFT** in **VoronoiFT**.

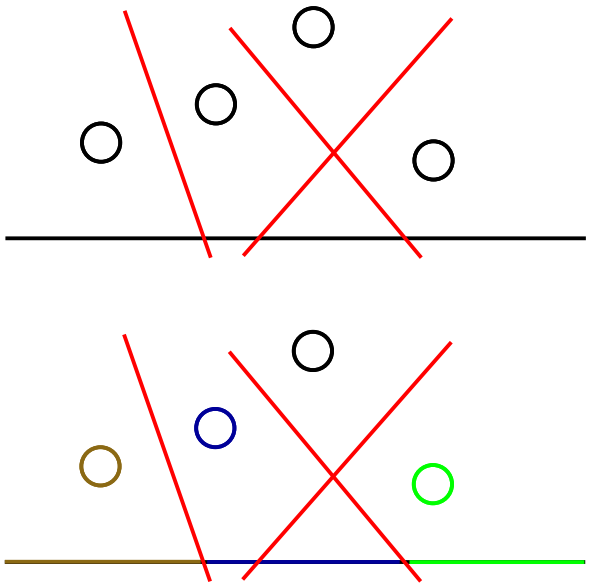


Slika 3.5: Voronojski Diagram

Na sliki 3.5 je primer Voronojskega diagrama. Za primer dveh točk je bil v predhodnem razdelku razrešen kot stranski produkt predstavitve lastnosti Evklidske razdalje (Slika 2.8). Zaključek dosežen je bilo dognanje, da prostor med točkama razdalji enakomerno loči ravnina. Razdelek lasten vsaki od dveh točk je njej prileżni, tako ločen, polprostor. Razdelek lasten točki glede na množino drugih je konveksen poligon. To sledi, ker za točke razdelka ne sme veljati, da ležijo v polprostoru bližjem kateri koli drugi točki diagrama; Tedaj bi namreč bile v razdelku tiste druge točke. Da je razdelek formiran, morajo biti izločene vse take točke. Izločeni so polprostori *oz.* napravljena je operacija preseka prostora z množico polprostorov, kar rezultira v konveksnem poligonu.

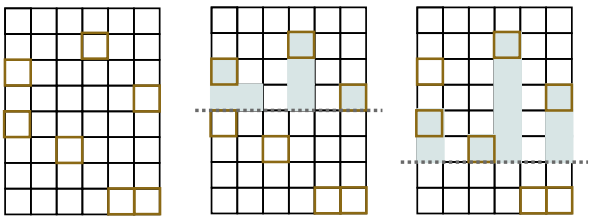
Breu et al. [9] pri predstavitvi konstrukcije Voronojskih diagramov pokažejo, da je preverjanje članstva izbrane točke v prostoru (Iskanje točke diagrama, v katere razdelku se izbrana točka nahaja) operacija časovne kompleksnosti $O(s \log s)$. V tem izrazu je s število točk diagrama. Presenetljivo, ista časovna meja velja tudi ko je diagram že predhodno konstruiran, in ostaja samo problem poizvedbe v diagramu. Kot tehnika v

pomoč nižanju časovne kompleksnosti se pokaže delna konstrukcija diagrama. Podatkovna struktura na kateri je možna opravitev poizvedb, ni v nobeni točki med pogonom algoritma izgrajena. Namesto tega, algoritem stremi k zgolj pridobitvi ekvivalentnih rezultatov na specifičnem območju diagrama. *ti.* druga lastnost Evklidske razdalje s slike 2.8 se pri tem početju pokaže uporabna.



Slika 3.6: Vrstica Delne Konstrukcije

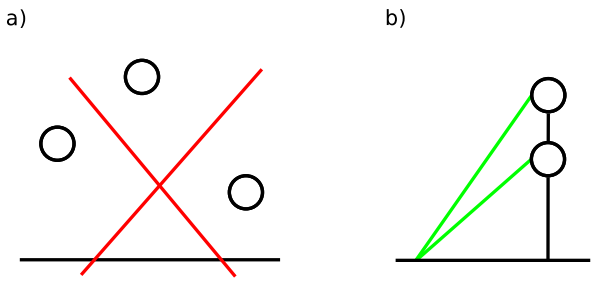
Slika 3.6 prikazuje vrstico, in več točk v soseščini. Označbe so podobne tistim slike 2.8 $\boldsymbol{d}$. Kjer sta prej bili zgolj dve točki, je zdaj med vsakim zaporednim parom točk z rdečo barvo izrisana razmejitvena površina. Razmejitvena površina loči premico na dva dela, eden od katerih je bliže levi točki, drugi katerih pa desni. Del premice nahajajoč se bliže določeni od točk, je del med razmejitvenimi površinami prejšnjega in naslednjega para katerih član je ta določena točka. Primer dodelitve po bližini z obarvitvijo prikazuje pomožna podslika zgoraj.



Slika 3.7: Slapovni Model

Uporaba delne konstrukcije vrste je predstavljena na simplifikiranem primeru slike 3.7. Na mreži so razporejene točke, razdalja do katerih je cilj izračuna. Vrstice mreže

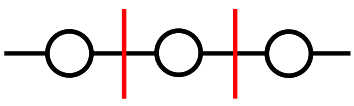
so obdelane ena za drugo, na sliki je prikazano stanje pri obdelavi četrte in šeste. Za razliko od "pravega" algoritma, so upoštevane razdalje samo od točk nad ciljno vrstico, podnje točke so ignorirane. Izraz slapovni model je bil izbran zato, ker pot do premice pravokotno iz njej najbliže točke, prikazana osenčeno (Na sliki plavo) in z nekaj domišljije, med potekom procesiranja vrstic spominja na padec vode.



Slika 3.8: Vrstica - Dodatno

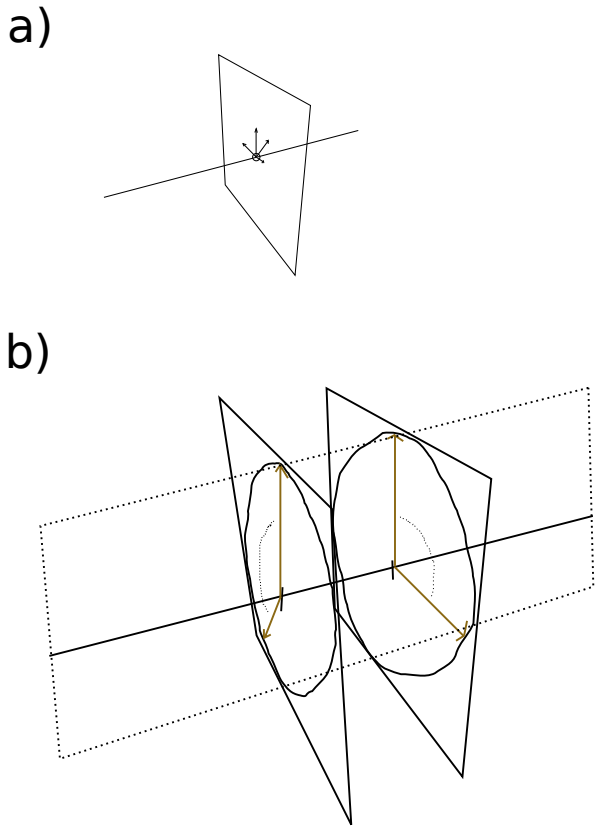
Z stanjem te množice točk nad vrstico, je na tej pognana dodelitev po bližini, kot s predhodne slike 3.6. Pomembna ugotovitev je, da je ta množica točk zadostna. Če je izbrana točka nad vrstico v določenem stolpcu tej vrstici najbližja, potem je v množici. Če izbrana točka v stolpcu vrstici ni najbližja, potem obstaja druga, bližja točka. Ta primer prikazuje slika 3.8 ***b*** . Je le vizualizacija posledice prve dodatne lastnost Evklidske razdalje: Vrstica teče v eni koordinati, točki se ujemata v drugi koordinati, ena od teh je vrstici bliže. Tedaj je bližja točka vrstici bliže v vsaki točki. Bolj oddaljena točka torej nima vpliva na rezultat dodelitve in izpade. Medtem ko je zadostna, pa množica točk nad vrstico ni potrebna. V posebnem primeru prikazanem na sliki ***a*** , presek ločnih površin sredinski točki ne dodeli nobenega odseka vrstice. Take točke pravtako ne vplivajo na rezultat dodelitve. Relevantni del algoritma zato premore mehanizem za njih izločitev.

Slapovni model je mogoče z določenimi razširitvami popraviti na izračun točne Evklidske transformacije razdalje.



Slika 3.9: Vrstica v Eni Dimenziji

Za začetek velja ugotovitev, ki razreši enodimenzionalni primer. Dvodimenzionalna rešitev omogoča poljubno postavitev točk diagrama. Postavljene so tako lahko neposredno na premico katere dodelitev računamo. Enodimenzionalni primer je zgolj poseben primer dvodimenzionalnega.



Slika 3.10: Dimenzijski Vrtež v Ravnino

Podobno prikazanemu na sliki 3.10 ***a*** je bilo omenjeno v razdelku o dodatnih lastnostih Evklidske razdalje. Podprostor vseh točk pravokotnih na premico v določeni točki, je množica vektorjev, ki prostor z smernega vektorja premice dopolni do ortonormalne baze. Izvršimo lahko (Razdalje ohranjajočo) rotacijo, ki smerni vektor premice ohranja, pri tem pa deluje na podprostoru. Vektor podprostora z poljubnimi koordinatami $v = (v_1, \ldots, v_{n-1})$ lahko zavrti tako, da ohrani zgolj eno koordinato $v' = (v'_1, 0, \ldots, 0)$. Vektor katerega prehaja koordinata v'_1 je z pravilno vnaprejšnjo izbiro baze podprostora lahko poljuben. Za potrebe te predstavitve je ugodna interpretacija smeri kot "navzgor". Na sliki ***b*** sta dva vektorja, vsak iz podprostora lastne točke premice, zavrtena v to navzgornjo smer. Ostane prostor parameteriziran z le dvema koordinatama: V smeri premice in v navzgornji smeri. Premici *oz.* vrstici je tako moč dodelitev prirediti po obstoječem dvodimenzionalnem postopku.

Dovolj je poznati dolžino vektorjev v podprostoru, ki dobo vrteni, smer po vrtenju je vedno navzgor. Dolžino pa lahko dobimo tako, da izračunamo Evklidsko transformacijo razdalje v tem podprostoru. Ta zahteva izračun Evklidske transformacije razdalje v podprostoru še nižje dimenzionalnosti, *itd.* . Ko proces doseže drugo dimenzijo, uporabimo dvodimenzionalni postopek iz predhodnih razdelkov. Obdelava v prvi dimenziji

pa je kot prej opisano zgolj poseben primer druge.

Jedro na Voronojskih diagramih osnovanega algoritma je redukcija vseh operacij na operacijo dodelitve vrstici, in specializirano rešitev dvodimenzionalnega problema.

Tabela 3.2: Maurer et al. Procedura **ComputeFT**

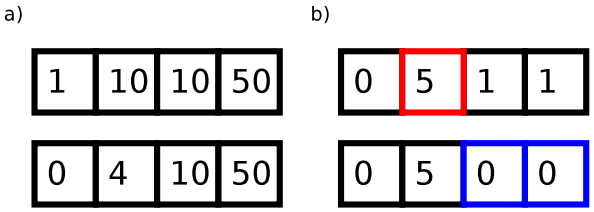
```
1.  if d = 1 then        /* Compute F_{d-1} */
2.    for i_1 ← 1 to n_1 do
3.      if I(i_1, j_2, ..., j_k) = 1 then
4.        F(i_1, j_2, ..., j_k) ← (i_1, j_2, ..., j_k)
5.      else
6.        F(i_1, j_2, ..., j_k) ← φ
7.      endif
8.    endfor
9.  else
10.   for i_d ← 1 to n_d do
11.     COMPUTEFT(d − 1, i_d, j_{d+1}, ..., j_k)
12.   endfor
13. endif
14. for i_1 ← 1 to n_1 do        /* Compute F_d */
15.   ...
16.     for i_{d-1} ← 1 to n_{d-1} do
17.       VORONOIFT(d, i_1, ..., i_{d-1}, j_{d+1}, ..., j_k)
18.     endfor
19.   ...
20. endfor
21. return
```

Tabela 3.2 citira proceduro **ComputeFT**, v izvorni psevdokodi algoritma članka [1]. Vrstice 1–10 so posebnost. V vsaki rekurzivni seriji klicev **ComputeFT** tečejo natanko enkrat, tip preden se algoritem izvrši v prvi dimenziji, in zgolj služijo inicializaciji polja.

Vrstice 10–14 sprožijo rekurzijo na podprostoru v točki $(\mathbf{v}, i_d, j_{d+1}, \ldots, j_k)$ trenutno obdelovane premice. Pri tem i_d variira, da preteče vse točke premice. $\mathbf{v}$ je seznam dolžine d, dimenzije od 1 do d predstavljajo dimenzije podprostora. Preostale koordinate, indeksi j, podajo koordinato točke na premici.

Ob vrsticah 14–21 opozarjam na nerodno predstavitev: Zahtevano številu vgnez-

denih for stavkov je $1 \ldots d-1$. Ko je dosežena globina rekurzije pri kateri d zavzame vrednost 1 torej $1 \ldots 0$. Zdi se, da je končni cilj tega dela algoritma formiranje koordinatne sekvence dolžine k, z uporabo indeksov i in j in številom d, torej: $[i_1 \ldots i_{d-1}, d, j_{d+1} \ldots j_k]$. V implementaciji $d = 1$ generira en sam klic procedure **VoronoiFT**, in nobene zanke.



Slika 3.11: Števec

Implementacija dinamično spremenljivega zank se v programskem jeziku C++ ne prilega direktno kakemu gradniku. Za izvedbo na očiten neposreden način bi mogoče bilo kvečjemu uporabiti generacijo kode z makro jezika LISP ... Implementiran je bil preprost števec, zasnovan po principu s slike 3.11. Slika ***a*** prikazuje meje števca, in primer začetnega stanja. Te ustrezajo $n_1 \ldots n_{d-1}$ psevdokode **ComputeFT**. Števke so iz začetnega stanja inkrementirane. Potek inkrementa:

1. Inkrement poteka tako da je $n = 1$-ta števka najprej inkrementirana.
2. Če števka zdaj presega mejo na indeksu n, se n poveča za 1. Potek nadaljuje s 1. korakom.
3. Če števka meje ne presega, je *repni* del števca postavljen na vrednost nula.

Slika ***b*** prikazuje rokovanje z presegom meje *oz.* overflowom. Med inkrementom z začetnega stanja je rdeče obarvana števka prva, ki meje ne preseže. Modro obarvani del se nato postavi na nič. Druga procedura opisana v Maurer et al. je **VoronoiFT**, klicana kot pomožna rutina **ComputeFT**. Implementira proces dodelitve po bližini premici, nanjo posebnih pripomb ni.

Glede podatkovnih struktur velja omeniti uporabo konvencije indeksiranja polj, pri kateri v polju dolžine $n+1$ prvi element dobi indeks 1, zadnji pa n. To je v navzkrižju tako z načinom indeksiranja, nativnem C-jevskim programskim jezikom, kot z znanim spisom E. W. Dijkstre [6]. Omeniti velja oteženo (In napakam izpostavljeno) sestavljanje sekvenc, in računanje njih dolžine. V programskem jeziku C se implementacija polj indeksiranih začetno z indeksiom 1, izkaže kot težavna zaradi obskurnosti v standardu.

Razširjeni "trik" prikazan v tabeli 3.3 je pravzaprav napaka. Razdelek 6.5.6 : 8 standarda C99 [7], povzet:

Tabela 3.3: Naivna Implementacija 1-Indeksiranih Polj v C

```
#include <stdlib.h>

elt_t * OneBased(size_t len) {
  elt_t *mem = malloc(len * sizeof *mem);
  elt_t *mem_shifted = &(mem - 1);
  return mem_shifted;
}

int main(int argc, char **argv) {
  elt_t *array = OneBased(5);
  array[1]; // Indeksira prvi element

  return EXIT_SUCCESS;
}
```

Če kazalec števnik in rezultat oba kažeta na elemente istega polja, ali enega elementa po koncu polja, evaluacija naj ne rezultira v presegu (Overflow); Drugače je program nedefiniran.

Ker je formiran kazalec na element pred prvim zaseženim z klicem na *malloc*, je program nedefiniran. Presenetljivo je morda, da standard preprečuje samo formiranje kazalca, in ne le *dereferenciranja*.

Tako je prehod polja v navzprednji smeri dovoljen, v vzvratni smeri pa nedefiniran. Ilustrativni primer se nahaja na tabeli 3.4. Odpovedati se tako moramo uporabi nativnih tipov polj, ali se sprijazniti z prekomerno alokacijo elementov - Z njo dosežemo, da je indeks vedno v mejah. Pri implementaciji algoritma je bila uporabljena prva rešitev. Izgrajen je bil podatkovni tip, ki v ozadju uporablja polje, operacije nad katerim pa so zaslužne za pravilno transformacijo prejetih indeksov, v indekse polja. Ta pristop na učinkovitosti pridobi pri poljih višjih dimenzij. Če na primer dvodimenzionalno sliko predstavimo kot zgolj polje polj, in vsakega od teh prekomerno alociramo, se izguba prostora nabira. Z dodatnim nivojem abstrakcije, pa je lahko celoten potreben pomnilnik zasežen v enem kosu, točne velikosti.

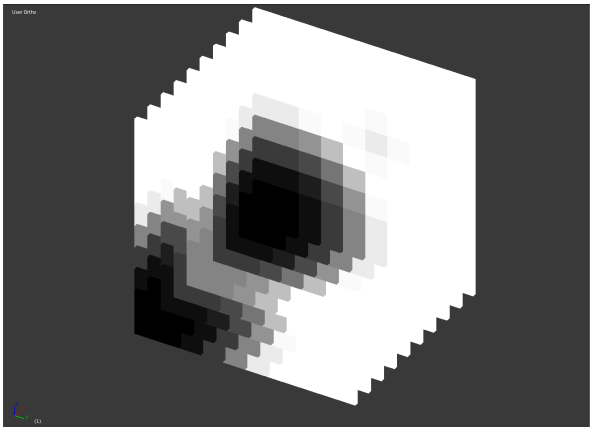
Pri preiskovanju člankom sem pogrešal vizualizacijo Evklidskih transformacij razdalj, aplicirano na tridimenzionalnih primerih. Navadno so predstavljeni samo dvodimenzionalni, ali izbrane rezine “izsekane” iz tridimenzionalnega primera (Denimo medicinsko skeniranje možganov, od katerega je prikazana ploskev v določeni ravnini). Za sam rezultat dvodimenzionalne transformacije so včasih uporabljene tri dimenzije:

Tabela 3.4: Primer Prehoda Polja

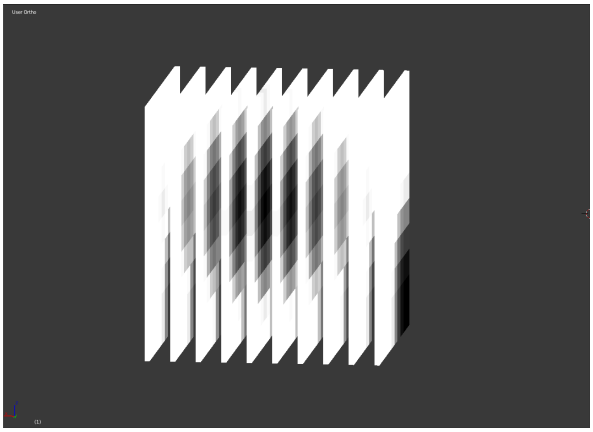
```
#include <stdlib.h>

int main(int argc, char **argv) {
  char elt;
  char *a, *b;

  // Ok, po stavku: a == (&elt) + 1
  for (a = &elt; a < (&elt) + 1; a++)
    ;

  // Nedefinirano, po stavku: b == (&elt) - 1
  for (b = elt; b >= (&elt); b--)
    ;

  return EXIT_SUCCESS;
}
```

Dve ravninski za lokacijo, in tretja, višina, za dobljeno vrednost razdalje. Tako je teže pridobiti intuicijo in povratno informacijo o delovanju posameznih algoritmov. Posebno deceptiven je članek J. Wang et al. [4]. Podane, v vzorcu obarvane, slike kipa, sfere in prostorske razporeditve kock, dajo slutiti da gre za rezultate obdelave tridimenzionalnega primera, z razdaljo mapirano v barvne odtenke. Vendar podrobnejši ogled pokaže, da so odtenki zgolj linearen barvni gradient. Kot rezultat Evklidske transformacije razdalje bi slike bile lahko interpretirane, če v kot na začetku praznega prostora, postavimo en sam voksel vrednosti ospredja objekta. Nato izračunamo transformacijo razdalje. Vrednosti namesto z eno cifro (Številom enot ozadja do najbližje-edine točke ospredja) razbijemo na tri smeri: Vertikalno, horizontalno, globinsko. Vsaki od teh smeri pripišemo eno osnovno barvo, in točke pobarvamo z mešanjem teh, proporcionalno na smerne vrednosti. Kot končni korak nato na prostoru opravimo presek z prostornino testnega objekta. Dosti bolj nazorne so *ti.* digitalne sfere, narisane v Borgefors et al. [3]. Konceptualno so rezultat situacije ko je okrog enega samega voksla ospredja, z formo zapolnjena površina katere transformacija razdalje je izbrana konstantna vrednost. Različne metrike razdalje rezultirajo v različnih oblikah, bolje prilegajoče se Evklidski metriki se približajo sferični obliki.

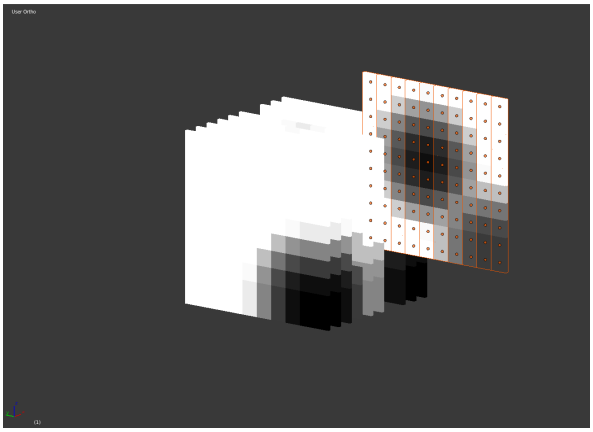
Novejše različice odprtokodnega 3D animacijskega programa Blender podpirajo skriptiranje v programskem jeziku Python. Mogočje je kreirati objekte in jim spre-



Slika 3.12: Pogled od Spodaj, s Strani



Slika 3.13: Pogled od Spredaj



Slika 3.14: Izvzetek

minjati različne lastnosti. V namene pomoči pri vizualizaciji je bila implementirana taka skripta. Testni vhodni podatki so najprej procesirani z algoritmom Evklidske

transformacije razdalje, nato s pomočjo skripte uvoženi v program Blender. Ker niso izvedene kakršnekoli posebne optimizacije, in je iz podatkov prenešena celotna mreža element za elementom, sistem za velike slike odpove. Mreža velikosti (n_1, n_2, n_3) kjer so števila n_i zelo visoka, namreč vsebuje približno n^3 elementov, raste z kubičnim členom dolžine stranic. Na slikah 3.12, 3.13 in 3.14 je primer manipulacije z uvoženim objektom.

Implementirana algoritma sta bila primerjana glede vidikov učinkovitosti in korektnosti - Rezultat algoritma Borgefors et al. je približen, za uporabo v praktične primere je koristno, poleg napakčnih karakteristik v najslabšem možnem primeru, pridobiti predstavo o razlikah generiranih na kakšnem generičnem problemu. Tabela 3.5 zbira testne rezultate.

Tabela 3.5: Rezultati

Velikost (št. vokslov)	Maurer (ms)	Borgefors (ms)
$50 \times 50 \times 50$	899	7.197
	871	7.101
	877	7.197
$100 \times 100 \times 100$	6.685	57.875
	6.679	58.412
	6.758	57.859
$100 \times 200 \times 200$	52.720	469.402
	52.788	466.030
	52.796	461.661

Pri izbiri tega testnega primera je bil vodilni namen primerjati - Izvorna hipoteza je, da se po delovanju kot opisanem, algoritma ne razhajata pretirano. Medtem ko Borgefors v prehodih vsak element obdela na simetričen način, pa faza dodelitve področij vrstici Maurer et al. zahteva količino procesiranja proporcionalno z številom točk vrstice za katere velja, da obstaja vsaj eden element ospredja v podprostoru (Za premico paralelno dimenziji d, dimenzije $d-1$) ortogonalnem premici v tej točki. Ni torej v obeh primerih vseeno kakšne so vhodne vrednosti vokslov danih v obdelavo. Maurer et al. je z izbiro testnega primera namenoma penaliziran. Pogoji ki sprožijo računsko zahtevne poti se, v treh dimenzijah praktično rečeno, izkažejo za zahtevo po prisotnosti vokslov vrednosti ospredja, v vsaki vrstici, nato v vsaki rezini prostora. Ta zasedenost vokslov je dosežena z generiranjem primitivnih elementov velikosti $2 \times 2 \times 2$ voksla. Generiranje se ponavlja, dokler ni zaseden določen delež celotnega volumna prostora; V primeru tabele 3.5 delež meri 50 odstotkov. Pozicioniranje generiranih elementov je opravljeno naključno.

Izkaže se, da kljub temu nasprotku, meritve časa izvajanja kažejo v prid algoritmu Maurer et al.. To je moč pripisati implementaciji različice Borgefors et al. uporabljajoč *ti.* D-Evklidsko metriko. Druge v njihovi ekspoziciji predstavljene metrike kot sta Chamfer in Hyperoctogonal, so prilagojene učinkoviti izvršitvi (Zahtevajo denimo ne operacij z plavajočo vejico), pri čemer je žrtvovana korektnost. Na žalost je zgornja meja odstopanja od idealnih vrednosti, pri teh metrikah funkcija površine rezine v dvodimenzionalnem primeru $((M-1)\times(M-1)$, kjer je M najvišja od izračunanih globalnih razdalj, $\max(\{|FT_i(a)| : a \in I\})$. V najslabšem primeru je M dolžina stranice vhodne slike I. Pogoj na konfiguraciji vokslov, da ta rezultira v najslabši vrednosti M, je: Voksel vrednosti ospredja v enem robu, voksel vrednosti ozadja v drugem. Vezna črta tako razporejenega para vokslov je diagonala pravokotnika, najdaljša ravna črta mogoča v pravokotniku.), oziroma funkcija volumna prostora v trodimenzionalnem $((M-1)\times(M-1)\times\dots)$. D-Evklidska metrika je edina od naštetih s strani Borgefors et al. [3, TABLE 1], katere odstopanje je konstanta. Konstanta je obenem nizka (Citirani vrednosti sta -0.29 in -0.09, glede na podrazličico), in v velikosti vhodne slike neodvisna. Te druge metrike torej ne zadostijo, uporabljena je D-Evklidska.

Specifično mesto ki izstopa, po pregledu algoritma z pozornostjo na računsko zahtevnost, je izračun minimum kandidatne množice lokalnih razdalj pri propagaciji. Posebnost implementacije D-Evklidske metrike na tem mestu je namreč uporaba vektorskega tipa za hrambo stanja vokslov med obdelavo. Na teh vektorskih vrednostih morajo v namene razpoznave minimuma z vršitvijo primerjanja biti neprestano opravljani izračuni kvadrata razdalje. Kvadrat razdalje v namene primerjave zadošča. Neenakost $\sqrt{a_1^2+b_1^2} < \sqrt{a_2^2+b_2^2}$, za tu uporabljane realne a in b, velja tudi po korenjenju obeh strani. Tako sta prihranjena vsaj izračuna dvojice kvadratnih korenon na primerjavo. Izrazi a^2+b^2 so sicer za iste vrednosti a in b izračunani večkrat, kar navaja na priložnosti glede memoizacije ali medpomnjenja vmesnih rezultatov. Propagacija razdalje v prehodih preko slike načeloma lahko v vsakem koraku vrednost voksla zamenja. Ugotovljeno bi torej moralo biti, katere regije bodo ponovno uporabljene in katerih izračune se tako izplača medpomniti. Ni očitno kako bi to lahko izvedli, brez investiranja procesne moči primerljive magnitude z optimizacijo prihranjeni.

Da je to mesto zaslužno za večinski delež izračunov v dejanski implementaciji, je bilo potrjeno z pomočjo profilinga. Tradicionalna tehnika za profiling programov je instrumentacija: Na želena meritvena mesta je dodana koda, ki ob izvršitvi hrani statistiko o številu klicev, pretečenem času, *itd.* . Ta koda je lahko vstavljena s strani prevajalnika programskega jezika. Razširjeni prevajalniki C++ premorejo zahtevano funkcionalnost generiranja kode, in podporo izvršnega okolja. Izvršitev instrumentiranih funkcij zadobi določen overhead, navadno kratek fiksen dodaten pretečen čas na vsak klic funkcije. Ker je za omenjeno izvedbo računanja minimuma in Evklidske raz-

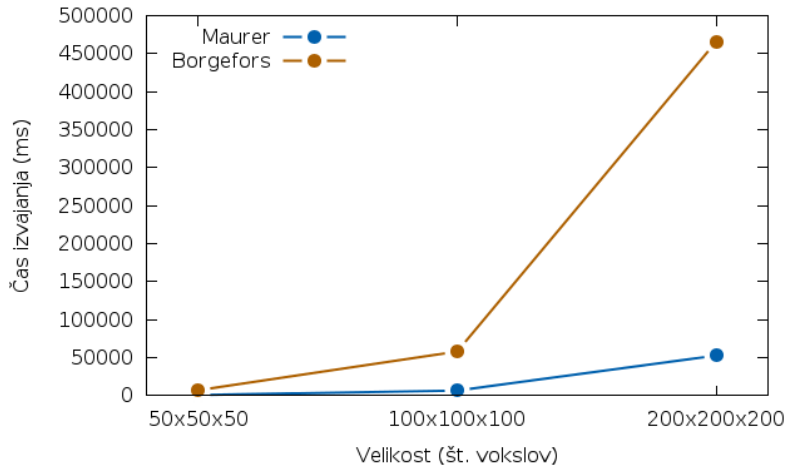
dalje same potrebna koda zelo kratka - Pričakujemo lahko, da nje optimiziran prevod meri zgolj peščico strojnih inštrukcij - obstaja nevarnost, da trčimo ob dno, kjer je čas izvajanja dominiran s strani instrumentacije kode. Instrumentacijska rešitev specifično GCC prevajalnika je tehnično pomanjkljiva. Ker optimizacija za kompleksne procesorske arhitekture (Podpirajoč denimo avtomatično paralelizacijo, out-of-order execution) zahteva premešanje generiranih strojnih inštrukcij, je velik izziv doseči tako postavitev instrumentacijskih klicev, ki natančno loči posamezno funkcijsko enoto. Poleg tega je opazno, z pregledom programa v razhroščevalniku, da je vstavitev instrumentacijske kode GCC omejena na zgolj eno na vsak *ti.* skladovni okvir (Stack frame). Zelo zaželena optimizacija inlininga je tako pokvarjena - Čas izvajanja bo merjen vključno z funkcijo, tarčo inlininga. Profiler uporabljen nad testi je bil program imenovan "Very Sleepy". Ta je primer implementacije *ti.* statističnega profilinga. Povzeto z spletne strani, opis avtorja programa koncizno predstavi princip delovanja:

> Je neinvaziven profiler, ki omogoča statističen profiling aplikacij C++ preko zajema vsebine registra kazalca inštrukcij. Profiler uporablja tehniko, pri kateri profiling teče v ločeni niti od programa samega. V vsakem določenem časovnem intervalu zajame vrednost registra, in iz nje izpelje vrstico kode trenutno v izvršitvi.

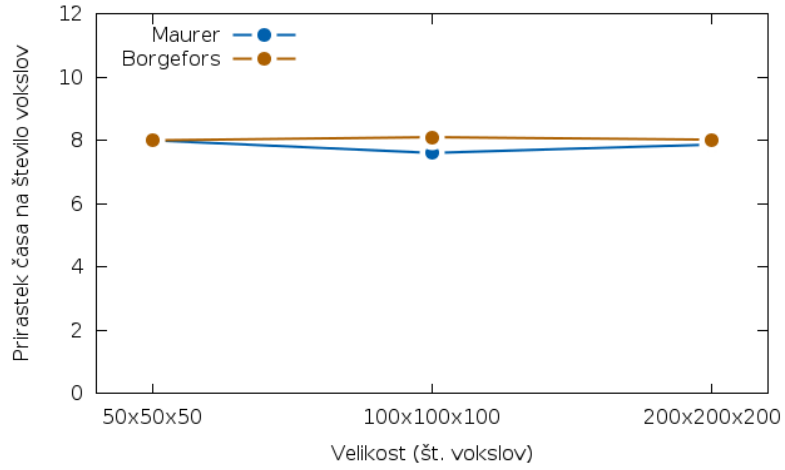
Izogne se tako nekaterim težavam tradicionalnega profilinga. Rezultati so kot pričakovani. Program približno 20 odstotkov časa porabi pri alokaciji pomnilnika. To je v mejah običajnega za C++ programe, ki kakorkoli uporabljajo objektno naravnane gradnike, podatkovne strukture standardne knjižnice, *itd.* . Pri bolj agresivni optimizaciji programa bi večina pomnilnika lahko bila prealocirana, in reciklirana namesto uporabe splošne alo- in dealo- kacijske funkcionalnosti. Približno 80 odstotkov je porabljenih za računanje minimuma (Merjeno inkluzivno. Klic, ki alocira pomnilnik znotraj te funkcije, torej prispeva k tej števki. Seštevek procentuaž je zato lahko višji od 100). Adicija vektorjev in izračun formule Evklidske razdalje spadata pod to kategorijo. Preostanek časa program preteče v prehodih skozi polja, pomožnih rutinah, dekodiranju vhodnih podatkov iz datoteke, in podobnih opravilih.

Ni presenečenje, da so na tabeli 3.5 prikazani rezultati konsistentno slabši za algoritem Borgefors et al. Testi so bili ponovljeni trikrat. Ker se pokaže manjek kakršnihkoli relevantnih variabilnosti v času izvajanja, tri ponovitve verjetno zadostijo.

Na sliki 3.15 so podatki tabele prikazani grafično. Zdi se lahko, da slika namiguje na to, da izvajalni čas algoritma Borgefors et al. narašča hitreje od Maurer et al., vendar je zavedljiva. Rezultati vizualizirani na način s slike 3.16 namreč pokažejo da je časovni prirastek sorazmeren s številom vokslov. Čas izvajanja se med trenutnim in naslednjim testom poveča za skoraj točno konstanten faktor 8. Ker je v vsakem



Slika 3.15: Rezultati: Vizualizacija



Slika 3.16: Rezultati: Prirastki

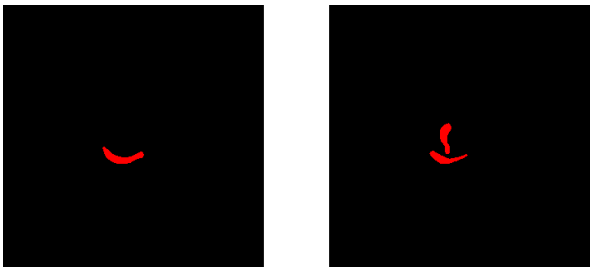
od sledočih si testov stranica prostora dvakrat daljša sledi (Enačba), in rezultata se ujemata:

$$N = 2 * a \times 2 * a \times 2 * a = (2 * 2 * 2) \times \cdots = 8 \times \ldots$$

Z vidika korektnosti izvajanja je dovolj opazovati rezultate Borgefors et al.. Kot približen algoritem je edini od implementiranih, ki je podvržen aproksimacijam. Pri prvih dveh testih manjšega števila vokslov, do odstopanj ni prišlo. Šele pri tretjem, na prostoru velikosti $200 \times 200 \times 200$, je po večih regeneriranjih naključne razpostavitve prišlo do odstopanja, in sicer vrednosti enega samega voksla v prvi vrstici prve rezine. Odstopanje je velikosti $|\sqrt{2} - 1|$ in se sklada z ugotovitvami zapisanimi v [3, p. 9]: "Maksimalno odstopanje od prave Evklidske razdalje je neodvisno od volumna ... Izračunana vrednost je vedno večja ali enaka pravi vrednosti."

Z prejšnjimi ugotovitvami je mogoče argumentirati preferenco glede uporabe predstavljenih algoritmov. Bistveno hitrejši in nenatančnosti prost algoritem Maurer et al. se je za boljšega izkazal v vsakem primeru.

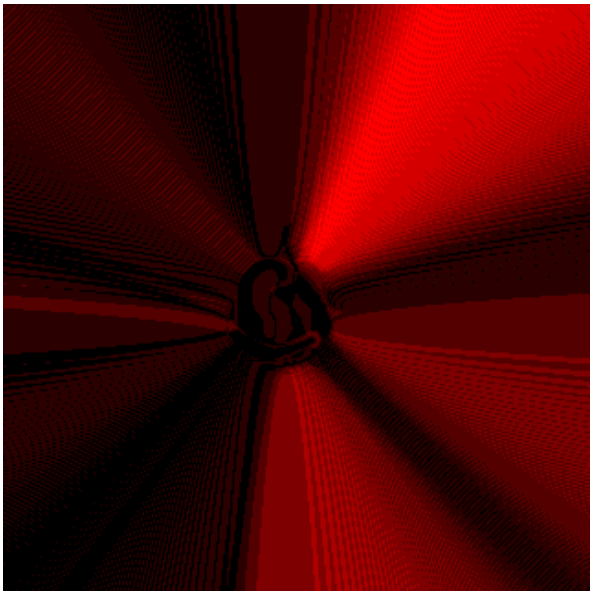
V komplement sintetičnemu testu sta algoritma bila uporabljena na primeru medicinske slike. Rentgenski posnetek je bil zajet z pomočjo naprave uporabljajoč tehnologijo CT (Računalniška tomografija). Pri tej so s senzorjem z različnih zornih kotov zajeti podatki uporabljeni za rekonstrukcijo tridimenzionalne forme določenih telesnih organov. V primerih ko je skeniranje vršeno v namene predoperacijskega planiranja posega, lahko informacijska tehnologija igra podporno vlogo pri različnih vizualizacijah. Relevantna v tem razdelku je prikaz in primerjanje označb obodov (Countour). Množica ekspertov samostojno segmentira posamezno slikovno rezino, tako da vanjo vriše enega ali več obodov, po kriteriju prisotnosti določene patološke lastnosti. Dobljene ročne segmentacije so nato primerjane. Namen je dodatno informacijo, ki jo v sistem vnese vnesejo dodatne redundantne ponovitve, izrabiti kot orodje za zgodnje odkrivanje razhajanj v analizi patološkega stanja. Evklidska transformacija razdalje je lahko uporabljena kot eden od načinov primerjave. Preprost, tu implementiran, algoritem sestoji zgolj iz izračuna transformacije na paru segmentacij, in temu sledečemu odštetju vrednosti na mestih, ustrezajočih istoležnim prostorskim vokslom.



Slika 3.17: Izbrana regija segmentacije

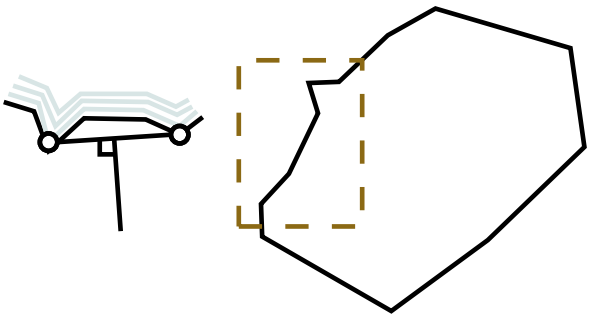
Slika 3.17 prikazuje rezino slike. Izbrana je bila približno na sredini zajetega intervala. Ustreza ker vsebuje prisotnost tako delov pripadajočih referenčni sliki in odsotnih v drugi, kot obratno. Različno je število zaključenih področij, katerih druga slika vsebuje dve.

Glede na to, da segmentacija slike hrani zgolj dve možni stanji (Znotraj označenega področja, zunaj označenega področja), obstajata dva očitna načina dodelitve vrednosti ozadja in ospredja, zahtevanih s strani algoritma Evklidske transformacije razdalje. Prvi način, prikazan na sliki 3.18, točkam v označitvi dodeli vrednost ozadja. Do teh so razdalje računane v preostanku slike. Odtenek rdeče je interpretiran kot razlika med vrednostima razdalje dodeljenima referenčni in drugi sliki v določeni točki. Črna so področja enake oddaljenosti. Oddaljenost je enaka vsaj v preseku obeh označitev



Slika 3.18: Absolutna diferenca regije

- Nula. V razdelku o lastnostih razdalje je bil predstavljen primer področij enakosti razdalje generiran s strani dveh točk. Področja so polprostori, meja dveh področij je premica. Po prvi lastnosti so, kot nazorneje prikazano na primeru poprepreproščenega *ti.* slapovnega algoritma, pomembni zgolj *najbližji voksli* (Če med dvema točkama potegnemo premico, je razdalja od točk u in v, skozi kateri vodi pravokotnica z istega mesta premice, do poljubne točke na premici vedno krajša za točno določeno od teh, po pravokotnici bližjo). Slika 3.19 prikazuje izsek večjega objekta. Z svetlim odtenkom

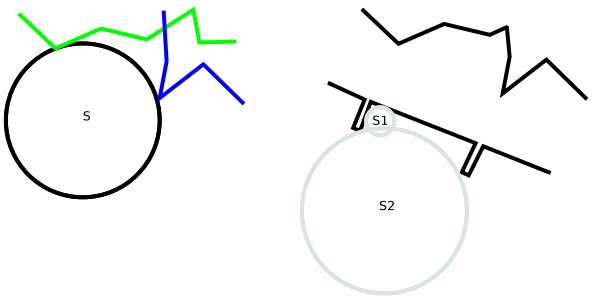


Slika 3.19: Krajišči na poligonskem objektu

modre barve so označene točke, ki na regijo na premico vežočo krajišči nimajo vpliva. Če je premica premo pomaknjena v smeri vstran objekta, se nevpliv teh točk ohranja. Notranje točke so tako izločene, ostanejo samo še vplivi robov.

V primeru dveh objektov - In bolj natančno v primeru dveh objektov katerih vrednosti razdalje Evklidske transformacije se odštejejo, je intuitivno pogoj, ki določi ako

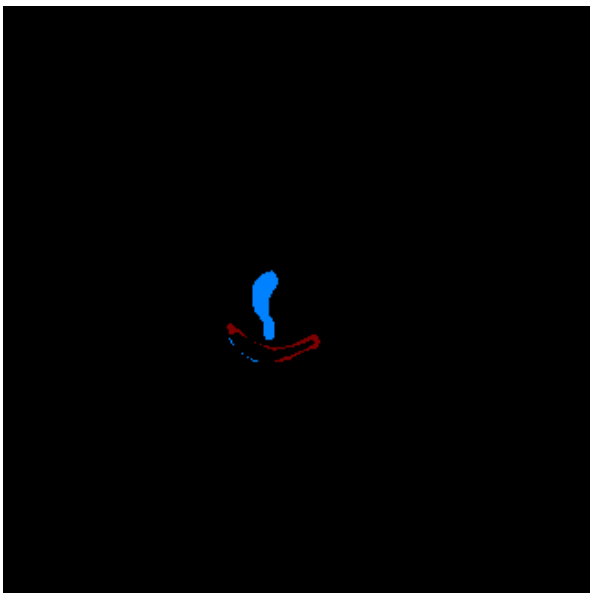
določena točka S leži v področju enakomerne oddaljenosti, naslednji (Slika 3.20): Če iz



Slika 3.20: Bližina dveh predmetov

središča v izbrani točki S potegnemo krožnico z polmerom enakim razdalji do najbližje točke prvega objekta, točka leži v področju enakomerne oddaljenosti, če krožnica seka drugi objekt v eni sami točki. V preostanku slike 3.20 sta roba objektov, ki sta prikazana, v namen preproščenja združena v enega. Ta enotni rob je nato dodatno porezan, da ostanejo samo relevantni elementi. S_1 in S_2 označujeta dve izbrani točki, eno zelo blizu, drugo bolj oddaljeno od roba. Vidno je, da za S_1 obstaja veliko sosednjih točk v katerih je zgoraj omenjeni pogoj izpolnjen. Poleg dveh "vrhov", je lahko v namene izpolnitve uporabljeno vezno dno. Kurvatura krožnice S_2 temu v razliko preprečuje postavitev točke tako, da krožnica hkrati seka enega od vrhov in dno. V poštev pridejo zgolj položaji ko sta sekana oba vrhova. Zdi se, da lahko pričakujemo, da bo v neposredni bližini objekta sorazmerno obilje dodatnih točk, enakomerno oddaljenih od obeh objektov. V daljavi pa, da njih število upada, in položaj področij določa razpostavitev vrhov v obema objektoma. Rezultat 3.18 postane tako laže upravičljiv. Opazne so večje črne cone na okolici kjer v neobdelanih podatkih ležijo objekti. Od objektov vodi manjše število "cevi" valjastih oblik. Morda bi pričakovali, zaradi direktne interpretacije problema Evklidske transformacije razdalje kot vzorčenjem voronojskega diagrama, da bodo generirana področja konveksni poligoni, in njih meje premice. Vendar se to, k razlagi čegar pripomore prejšnja ekspozicija, ne zgodi.

Od te nepregledne metode za podporo segmentaciji z uporabo Evklidske transformacije lahko preidemo na njej dualno: Namesto vrednosti ozadja, točkam v označitvi dodeli vrednost ospredja. Že na prvi pogled se slika 3.21 zdi jasnejša. Dodatna obarvanost (Modro), je rezultat spremembe v interpretaciji razlike pridobljene z odštevkom vrednosti v posamezni točki. Zdaj je beležena *smer* deficita vrednosti. Rdeč odtenek obarvanja pripada področjem katerih razlika med vrednostjo na referenčni sliki ter na drugi sliki je pozitivna, modri odtenek pa kjer je razlika negativna. V danem primeru je glede na referenčno sliko segmentator začrtal območja preobširno. Ločeni večji skupek je obarvan kontrastno in takoj ločljiv. Manjši ostanki modrega odtenka so prisotni po



Slika 3.21: Negirana diferenca regije

sledi roba spodnje regije, levo. Tudi tam je na ročno segmentirani sliki nastal eksces. Rdeča kontura označuje področja ki v segmentacijo niso bila zajeta, črna izvotljenost na sredini pa je področje kjer se segmentatorju z referenco uspe ujeti. Zajete so tako pomembne informacije o razlikah med primerjanima segmentacijama.

4 Zaključek

Implementirana in predstavljena sta bila dva algoritma namenjena računanju Evklidske transformacije razdalje. Specifična sta bila izbrana ker sta si medsebojno v večji meri kontrastirata: Eden je točen, drugi približen. Eden uporablja za implementacijo zelo preprost način propagacije razdalj, drugi zahteva bolj intenzivne podatkovne strukture za vzpostavitev Voronojskega diagrama. Do izraza razlike v asimptotični časovni kompleksnosti ne pridejo: Algoritem Borgefors et al. zahteva število prehodov procesiranja, eksponentno dimenziji problema, 2^n. Ker pa je cilj rešitev problemov specifično v treh dimenzijah, konstantnih osem prehodov ohranja linearnost. Več časa je namenjenega analizi delovanja Voronojskega algoritma, posebej delovanju dodatnih lastnosti razdalje, zahtevanih od uporabljanih metrik.

Avtor menim, da bi bilo potencialno zanimivo raziskati *ti.* šibkost zaznano v algoritmu Borgefors et al. bolj podrobno. Z nekaj sreče je bila modaliteta napake pravilno diagnosticirana kot pomanjkljivost algoritma. Bolj verjetno pa je posledica napake implementaciji ali razumevanju postopka z moje strani.

Literatura

[1] C. R. Maurer, R. Qi in V. Raghavan, A Linear Time Algorithm for Computing Exact Euclidean Distance Transforms of Binary Images in Arbitrary Dimensions, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25 (2003), 265–270.

[2] K. C. Ciesielski, X. Chen, G. J. Grevera in J. K. Udupa, Linear Time Algorithms for Exact Distance Transform, *J Math Imaging Vis* 39 (2010), 193–209.

[3] G. Borgefors, Distance Transformations in Arbitrary Dimensions, *Computer Vision, Graphics, and Image Processing* 27 (1984), 321–345.

[4] Y. Tan in J. Wang, Efficient Euclidean distance transform algorithm of binary images in arbitrary dimensions, *Pattern Recognition* 46 (2013), 230–242.

[5] L. Euler, General formulas for the translation of arbitrary rigid bodies, *Theoria motus corporum rigidorum* 9 (1790), 449–460.

[6] E. W. Dijkstra, Why numbering should start at zero, *EWD-numbered documents* (1982), 0–$\infty + 1$.

[7] American National Standards Institute, *Working Paper for Draft Proposed International Standard for Information Systems: Programming Language C++*, 1995.

[8] Y-B. Jia, J. B. Kuipers, *Quaternions and Rotation Sequences*, 1999.

[9] H. Breu, J. Gil, D. Kirkpatrick in M. Werman, Linear Time Euclidean Distance Transform Algorithms, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17 (1995), 529–533.