UNIVERZITA MATEJA BELA V BANSKEJ BYSTRICI
FAKULTA PRÍRODNÝCH VIED

Použitie techniky získavania tvaru z tieňovania na analýzu dezénu pneumatiky a podrážky

Diplomová práca

6f090f1d-13e5-4462-a32c-781a90a35465

2016 **Bc. Vladimír Mlynárčik**

UNIVERZITA MATEJA BELA V BANSKEJ BYSTRICI
FAKULTA PRÍRODNÝCH VIED

Použitie techniky získavania tvaru z tieňovania na analýzu dezénu pneumatiky a podrážky

Diplomová práca

6f090f1d-13e5-4462-a32c-781a90a35465

Študijný program: Aplikovaná informatika

Študijný odbor: Aplikovaná informatika

Pracovisko (katedra, inštitút, . . .): KIN FPV – Katedra informatiky

Vedúci diplomovej práce: Mgr. Michal Vagač, PhD.

Banská Bystrica, 2016 **Bc. Vladimír Mlynárčik**

Univerzita Mateja Bela v Banskej Bystrici
Fakulta prírodných vied

3700982331

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta:	Bc. Vladimír Mlynárčik
Študijný program:	aplikovaná informatika (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor:	aplikovaná informatika
Typ záverečnej práce:	Magisterská záverečná práca
Jazyk záverečnej práce:	slovenský
Sekundárny jazyk:	anglický

Názov: Použitie techniky získavania tvaru z tieňovania na analýzu dezénu pneumatiky a podrážky

Anotácia: Analyzujte algoritmy získavania tvaru z tieňovania (shape from shading) z pohľadu skúmania tvaru dezénu pneumatiky a podrážky obuvi. Vytvorte a vyhodnoťte experimenty s reálnymi dátami.

Vedúci:	Mgr. Michal Vagač, PhD.
Katedra:	KIN FPV - Katedra informatiky
Vedúci katedry:	PaedDr. Mgr. Vladimír Siládi, PhD.
Dátum zadania:	10.04.2015

Dátum schválenia: 14.05.2015

prof. Ing. Miroslav Svítek, Dr.
garant študijného programu

Čestné prehlásenie

Čestne prehlasujem, že som túto diplomovú prácu vypracoval samostatne s použitím uvedených zdrojov.

V Železnej Breznici, dňa 3.5.2016 .....

Poďakovanie

Touto cestou chcem poďakovať Mgr. Michalovi Vagačovi, PhD., vedúcemu mojej bakalárskej práce, za pomoc, čas a inšpiráciu pri písaní práce.

Abstrakt

MLYNÁRČIK, Vladimír: Použitie techniky získavania tvaru z tieňovania na analýzu dezénu pneumatiky a podrážky. [Diplomová práca] / Bc. Vladimír Mlynárčik. – Univerzita Mateja Bela v Banskej Bystrici. Fakulta prírodných vied; Katedra informatiky. – Školiteľ: Mgr. Michal Vagač, PhD. Stupeň odbornej kvalifikácie: Magister. – Banská Bystrica: FPV UMB, 2016, 45 s.

Cieľom práce je zistiť, či je možné využiť techniky získavania tvaru z tieňovania na analýzu obrázkov dezénov pneumatík a podrážok topánok. Práca je rozdelená do troch kapitol. Prvá kapitola sa zaoberá teoretickou stránkou získavania tvaru z tieňovania. Je v nej prezentovaných niekoľko prístupov k problému získavania tvaru z tieňovnania. Druhá kapitola sa zoberá implementáciou Pentlandovho a Tsaiovho a Shahovho prístupu k problému získavania tvaru z tieňovnania do programu. Program je napísaný v Jave s využitím OpenCV. V tretej, poslednej kapitole sú prezentované výsledky experimentov s vytvoreným programom na vzorkách dezénov pneumatík a podrážok topánok.

Kľúčové slová: získavanie tvaru z tieňovania, Pentland, Tsai, Shah, Java, OpenCV, analýza obrazu, počítačové videnie

Abstract

MLYNÁRČIK, Vladimír: Using Shape from Shading technique to analyze tire profile and shoe sole. [Master thesis] / Bc. Vladimír Mlynárčik. – Matej Bel University Banská Bystrica. Faculty of Natural Sciences; Department of computer sciences. – Supervisor: Mgr. Michal Vagač, PhD. Degree of Qualification: Master. – Banská Bystrica: FPV UMB, 2016, 45 s.

The goal of this work is to find out the possibility of using shape from shading techniques to analyze images of tire profiles and shoe soles. Work is devided into three chapters. The first chapter is being concerned with theory of shape from shading problem. We are presenting here several approaches to shape from shading problem. The second chapter is being concerned with the way of how to implement Pentland's and Tsai and Shah's approach to shape from shading problem to the program. The program is written in Java and uses OpenCV. In the third, the last chapter we are presenting results of our experiments with created program on a samples of tire profiles and shoe soles.

Keywords: shape from shading, Pentland, Tsai, Shah, Java, OpenCV, image analysis, computer vision

Obsah

Zoznam obrázkov

Zoznam symbolov a skratiek

SFS – Získavanie tvaru z tieňovania

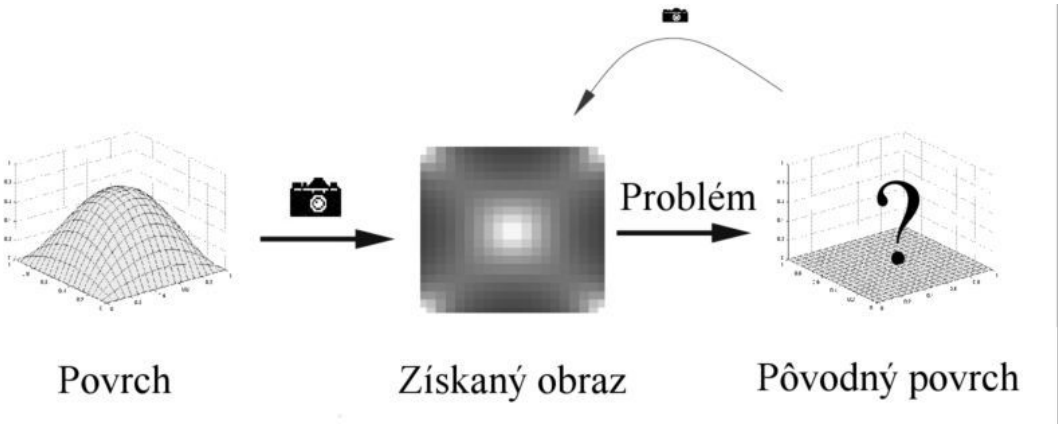
Úvod

Ak vytvoríme fotografiu ľubovolnej scény v trojrozmernom svete dostaneme dvojrozmerný obraz tejto scény. Tento obraz je tvorený povrchom, ktorý nasnímal snímač. Ten v podstate zachytil jas vyžarovaný z tohoto povrchu. Inak povedané, jeho tieňovanie. Pri tomto prevode však prichádzame o informácie o tvare povrchu. To ako získať tvar povrchu (3D), z jediného obrazu (2D), riešia techniky získavania tvaru z tieňovania (*angl. Shape From Shading*), ďalej len SFS. Problém SFS je znázornený na obrázku 1.

Táto práca je rozdelená do troch kapitol. Prvá kapitola sa zaoberá teóriou, ktorá je potrebná na to, aby sme boli schopní získať tvar povrchu z jeho tieňovania. Najprv objasníme pojmy ako odrazivosť a mapa odrazivosti. Potom prejdeme k všeobecnému rozdeleniu rôznych prístupov k problému SFS. Následne detailnejšie rozoberieme tri prístupy. Prvý Hornov, druhý Pentlandov a tretí Tsaiov a Shahov prístup.

Druhá kapitola opisuje spôsob implementácie vybraných prístupov k SFS v jazyku Java s využitím knižnice OpenCV. Táto kapitola je rozčlenená na dve časti. V prvej ukážeme ako sme implementovali Pentlandov prístup. V druhej Tsaiov a Shahov. Tieto dva prístupy sme vybrali hlavne kvôli tomu, že sú oproti Hornovmu prístupu jednoduchšie na implementáciu. Súčasne zastupujú dve vetvy všeobecného rozdelenia, ktoré sú najčastejšie používané a vylepšované v rôznych vedeckých prácach.

V tretej kapitole predstavíme výsledky, ktoré sme získali, použitím vytvorených programov, zo sady vzoriek obrázkov dezénov pneumatík a podrážok topánok.



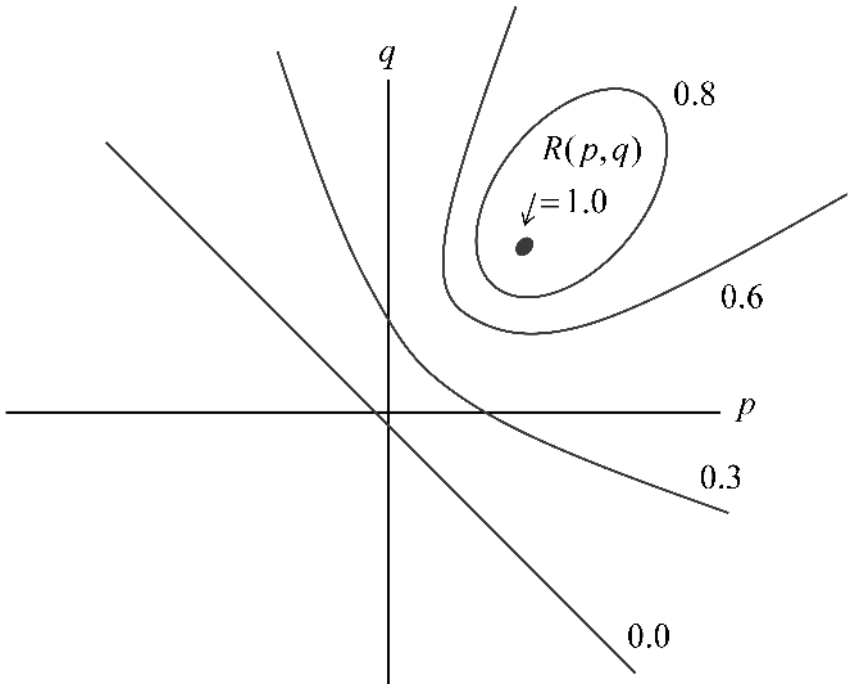
Obr. 1: Problém SFS [8].

1 Teória

Mnoho algoritmov SFS predokladá, že povrch zachytený na obraze má Lambertovskú odrazivosť. To znamená, že je dokonale matný a difúzny, takže odráža dopadajúce svetlo rovnomerne na všetky smery. Tento model je pomenovaný po Johannovi Heinrichovi Lambertovi, ktorý ho objavil. V takomto obraze je intenzita bodu (pixelu), označme ju E v bode (x, y) daná touto rovnicou.

$$\begin{aligned} E(x,y) &= R(p,q) \\ &= \frac{\cos(\sigma) + p\cos(\tau)\sin(\sigma) + q\sin(\sigma)\sin(\tau)}{\sqrt{1+p^2+q^2}} \end{aligned} \tag{1.1}$$

Premenná σ je sklon zdroja svetla, τ je jeho náklon a p, q sú parciálne derivácie povrchu z podľa osí x a y. Veľmi praktickým dôsledkom Lambertovskej odrazivosti je fakt, že vnímaný jas povrchu nezávisí na pozorovacom uhle. Takže ak poznáme orientáciu povrchu a pozíciu zdroja svetla môžeme vypočítať mapu odrazivosti R(p,q). Čo je vlastne projekcia gradientu povrchu na očakávané hodnoty intenzít pixelov obrazu [1]. Príklad mapy odrazivosti je na obrázku 2.



Obr. 2: Príklad mapy odrazivosti pre Lambertovský povrch [1].

Každá krivka na obrázku 2 odpovedá určitému jasu. Pre každý bod týchto kriviek platí, že

jednoznačne určuje gradient povrchu v zložkách p a q, ktorého výsledkom je daná odrazivosť, respektíve jas v tomto bode. Veľa, aj keď nie všetky prístupy k riešeniu problému SFS využívajú mapu odrazivosti, keďže tá ponúka spôsob, akým priamo vypočítať rozsah všetkých možných orientácií povrchu pre každý jeho bod. Algoritmy SFS môžeme rozdeliť na základe prístupu k problému SFS do štyroch skupín:

SFS s prístupom šírenia Tento prístup k riešeniu problému SFS využíva šírenie informácií o tvare povrchu smerom preč od počiatočných singulárnych bodov. Využíva sa skutočnosť, že v singulárnom bode poznáme jeho vzdialenosť od kamery (snímača). Historicky bol prvý krát použitý Hornom [3].

SFS s lokálnym prístupom Pri tomto prístupe sa vzdialenosť bodu povrchu od kamery vypočíta na základe veľkosti intenzity jasu tohoto bodu a jej prvej a druhej derivácie vzhľadom na priestor. Predpokladá sa tiež, že povrch je v každom svojom bode guľovitý a polomer zakrivenia sa v každom bode mení. Prvý krát bol použitý Pentlandom [6].

SFS s lineárnym prístupom SFS s lineárnym prístupom je založené na lineárnej aproximácií funkcie odrazivosti. Príkladom tohoto prístupu je práca Tsaia a Shaha [7].

SFS s prístupom minimalizácie Pri tomto prístupe sa trojrozmerný tvar povrchu získava minimalizáciou funkcie energie celého obrazu. Vychádzame z toho, že poznáme len intenzitu každého pixela. Z tejto informácie sa algoritmus SFS snaží určiť hodnoty vertikálneho a horizontálneho gradientu v danom bode povrchu. Keďže máme dve neznáme a jednu známu premennú, tak musíme zaviesť nejaké ďalšie podmienky optimalizácie. Tie môžu byť nasledovné [1]:

Podmienka jasu sa snaží minimalizovať celkovú odchýlku jasu vo vytvorenom 3D obraze k pôvodnému obrazu. Minimalizuje sa táto funkcia:

$$\int\int (I(x,y) - R(x,y))^2 dxdy \quad .$$

Pričom I je pôvodný obraz a R je vytvorený obraz.

Podmienka veľkosti gradientu sa snaží minimalizovať rozdiel medzi intenzitou gra-

dientu pôvodného obraz a vytvoreného obrazu podľa vzťahu

$$\int\int (R_x(x,y) - I_x(x,y))^2 + (R_y(x,y) - I_y(x,y))^2 dxdy \quad .$$

Rx a Ry sú gradienty vytvoreného obrazu pozdĺž osí x a y. Podobne Ix a Iy sú gradienty pôvodného obrazu pozdĺž osí x a y.

Podmienka plynulosti sa snaží minimalizovať rýchlosť s akou sa mení gradient povrchu v smeroch osí x a y. Využíva funkciu energie, ktorá penalizuje prudké zmeny v orientácií povrchu. Tá má tvar:

$$\int\int (p_x^2 + p_y^2 + q_x^2 + q_y^2) dxdy \quad .$$

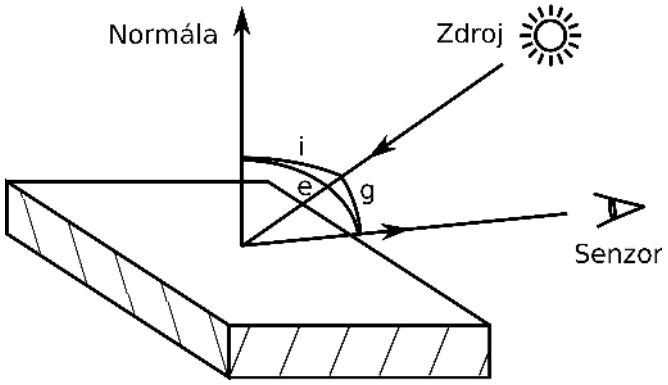
Pričom p_x, p_y, q_x a q_y sú parciálne derivácie zložiek gradientu p a q podľa osí x a y.

1.1 Hornov prístup k SFS

1.1.1 Funkcia odrazivosti

Majme nasledujúci model, ako na obrázku 3. Na určitú časť povrchu objektu dopadá lúč svetla, časť z neho sa pohltí materiálom a časť vyžiari naspäť do priestoru. Nás zaujíma najmä dopadajúci lúč a vyžiarený lúč. Označme veľkosť tej časti povrchu kam dopadol dopadajúci lúč ako dS. Písmenom i označme uhol pod akým dopadal dopadajúci lúč. Podobne písmenom e označme uhol, pod ktorým sa odrazil lúč vyžiarený. Uhly meriame vzhľadom na normálu vztýčenú v mieste dopadu. Ďalej ak zavedieme I_1 ako intenzitu dopadajúceho svetla na jednotku plochy kolmo na dopadajúci lúč. Potom množstvo svetla dopadajúceho na miesto dopadu vypočítame takto $I_1 * cos(i) * dS$ [4][3].

Podobne I_2 je intenzita vyžiareného lúča na jednotku priestorového uhla na jednotku plochy kolmo na vyžiarený lúč. Takže množstvo svetla zachyteného oblasťou ležiacou pod priestorovým uhlom, označme ho dW, na povrchu bude $I_2 * cos(e) * dS * dW$. Pri takto zavedených premenných potom môžeme funkciu odrazivosti definovať ako pomer I_2 k I_1, a teda výsledný



Obr. 3: Znázornenie premenných vo funkcií odrazivosti [4].

vzorec vyzerá nasledovne [4].

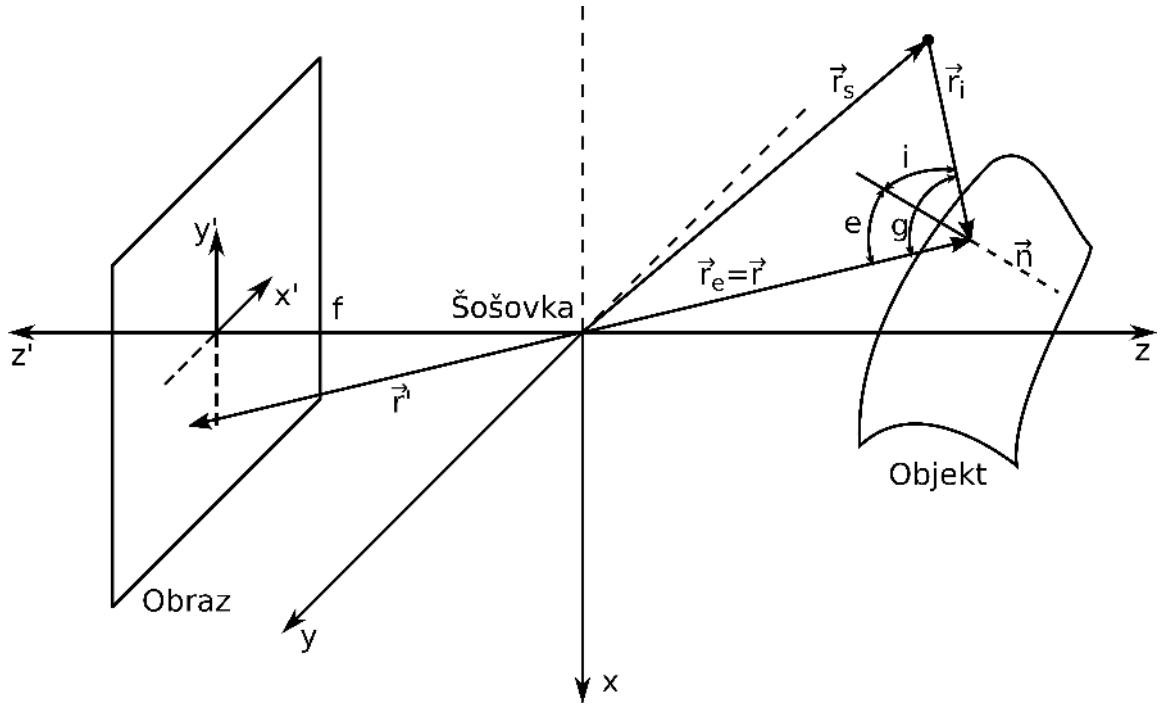
$$\phi(i, e, g) = \frac{I_2}{I_1} \tag{1.2}$$

Pre väčšinu povrchov platí, že funkcia odrazivosti závisí na farbe svetla. Zrkadlová zložka odrazeného svetla, čo je svetlo odrazené od povrch ešte pred tým ako do neho preniklo, sa nemení. Ale difúzna zložka sa sfarbí podľa mikroštruktúry daného materiálu. Preto pri návrhu funkcie odrazivosti môžeme vychádzať aj zo štruktúry daného povrchu. Preto aj napr. nasledujúce funkcie môžeme považovať za správne [3].

$$\phi(i, e, g) = \frac{I_2}{I_1} * \pi$$
$$\phi(i, e, g) = \frac{I_2}{I_1} * \cos(e)$$
$$\phi(i, e, g) = \frac{I_2}{I_1} * \cos(i)$$

1.1.2 Analytická formulácia problému SFS

Vychádzajme z toho, že poznáme len výsledný obraz, pozíciu svetelného zdroja pri snímaní a pozíciu kamery. V akomkoľvek bode obrazu vieme vypočítať uhol g (obrázok 4) zo známej polohy svetelného zdroja a pozície kamery (snímača, šošovky). Čo nevieme určiť sú uhly i a g, keďže nepoznáme normálu v danom bode obrazu. Existujú ale aj také body kde je normála známa, nazývame ich singulárne body. Tie sa dajú využiť pri hľadaní počiatočných podmienok. Ak sa ale chceme pohnúť z nášho bodu musíme v ňom poznať gradient, alebo aspoň jednu jeho zložku v smere osi x, alebo y [4][9].



Obr. 4: Projekcia v zobrazovacom systéme [3]

Pre osvetlenie obrazu platí nasledujúca rovnica:

$$A(r)\phi(I, E, G) = b(r') \tag{1.3}$$

$A(x, y, z) = t \cdot a(x, y, z)$.

t je pomer osvetlenia obrazu k luminiscencií objektu.

$a(x, y, z)$ je intenzita dopadajúceho svetla.

$r = (x, y, z)$ je bod na objekte a $r' = (x', y', f)$ je obraz toho bodu v obraze.

f je kolmá vzdialenosť roviny obrazu od šošovky.

$b(x', y')$ je intenzita osvetlenia nameraná v bode obrazu so súradnicami (x', y').

p a q sú parciálne derivácie z vzhľadom na osi x a y.

$I = \cos(i)$, $E = \cos(e)$ a $G = \cos(g)$.

Dá sa dokázať, že táto rovnica je nelineárna parciálna diferenciálna rovnica prvého rádu dvoch nezávislých premenných x a y, tvaru $F(x, y, z, p, q) = 0$. Dôkaz uvedený nebude. Rovnicu 1.3 môžeme rozpísať do sústavy piatich ekvivalentných obyčajných diferenciálnych

rovníc takto:

$$\begin{aligned} \dot{x} &= F_p, \qquad \dot{y} = F_q, \qquad \dot{z} = pF_p + qF_q \\ \dot{p} &= -F_x - pF_z, \qquad \dot{q} = -F_y - qF_z \end{aligned} \tag{1.4}$$

Bodka nad premennými značí deriváciu podľa s. Pričom s je parameter meniaci sa so vzdialenosťou pozdĺž charakteristického pruhu a dolný index značí parciálnu deriváciu [4].

Charakteristický pruh alebo len charakteristika, je krivka, ktorú získame nasledovným postupom. Nachádzame sa v určitom bode. V tomto bode určíme zložku gradientu v smere osi x alebo y. Potom sa v tomto smere posunieme o malý krok a tento postup opakujeme.

Rovnicu 1.3 môžeme vynásobiť ľubovolným nenulovým číslom λ bez toho, aby sme zmenili výsledný povrch.

$$\lambda = \frac{1}{\sqrt{(F_p^2 + F_q^2 + (pF_p + qF_q)^2)}}$$

Preto sústava diferenciálnych rovníc 1.4 môže vyzerať aj ako sústava 1.5. Líšiť sa bude len v hodnotách s v ktoromkoľvek bode povrchu. Parameter s pri takto stanovenom λ nám dá dĺžku oblúka charakteristiky [4].

$$\begin{aligned} \dot{x} &= \lambda F_p, \qquad \dot{y} = \lambda F_q, \qquad \dot{z} = \lambda(pF_p + qF_q) \\ \dot{p} &= \lambda(-F_x - pF_z), \qquad \dot{q} = \lambda(-F_y - qF_z) \end{aligned} \tag{1.5}$$

Z tejto sústavy je vidieť, že ak je menovateľ v λ rovný 0, tak celý výraz nemá zmysel. To sa stane, ak sa nachádzame v singulárnom bode, alebo na neurčitej hrane. Vtedy $F_p = F_q = 0$.

Zatiaľ uvedené rovnice sú všeobecné a dosť zložité. Avšak môžeme ich zjednodušiť zavedením určitých podmienok. Zaveďme niekoľko nových premenných. Nech A je vektor s trojicou súradníc. $|A|$ je veľkosť vektora A. $\hat{A}$ je jednotkový vektor prislúchajúci A a vypočítame ho $\hat{A} = A/|A|$. Skalárny súčin $A \bullet B = A\,B^T$ a výsledkom je matica rozmeru 3×3. Ďalej parciálna derivácia podľa vektora je vektor s tromi súradnicami a značiť ju budeme dolným indexom. Jej zložky vypočítame parciálnou deriváciou vzhľadom na každú súradnicu tohoto vektora. V prípade, že potrebujeme vypočítať deriváciu vektora podľa vektora, tak výsledkom bude matica rozmeru 3×3. Prvý riadok tejto matice vznikne deriváciou podľa prvej, druhý podľa druhej a tretí podľa tretej súradnice vektora. Tiež budeme potrebovať

vypočítať parciálnu deriváciu skalárneho súčinu $X = \hat{A} \bullet \hat{B}$ [4]. Potom $X_A = \left(\frac{1}{A}\right)\left(\hat{B} - X\hat{A}\right)$. Zo vzťahov vyššie vyplýva napr.

$$A_A = \hat{A} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Zaveďme ďalej normálu smerujúcu do vnútra povrchu v bode r ako $n = (-p, -q, 1)$ a $r_s = (x_s, y_s, z_s)$ ako pozíciu zdroja svetla. Potom vektor dopadajúceho lúča je $r_i = r - r_s$ a vektor lúča vyžiareného $-r_e = -r$. Pre I, E a G platí nasledovné:

$$I = \hat{n} \bullet \hat{r}_i,$$
$$E = \hat{n} \bullet \hat{r}_e,$$
$$G = \hat{r}_i \bullet \hat{r}_e$$

Z vyššie uvedeného vyplývajú nasledujúce vzťahy [4].

$$I_r = I_{r_i} = \left(\frac{1}{r_i}\right)(\hat{n} - I\hat{r}_i) \tag{1.6}$$

$$I_n = \left(\frac{1}{n}\right)(\hat{r}_i - I\hat{n}) \tag{1.7}$$

$$E_r = E_{r_e} = \left(\frac{1}{r_e}\right)(\hat{n} - E\hat{r}_e) \tag{1.8}$$

$$E_n = \left(\frac{1}{n}\right)(\hat{r}_e - E\hat{n}) \tag{1.9}$$

$$G_r = G_{r_i} + G_{r_e} = \left(\frac{1}{r_e}\right)(\hat{r}_i - G\hat{r}_e) + \left(\frac{1}{r_i}\right)(\hat{r}_e - G\hat{r}_i) \tag{1.10}$$

$$G_n = 0 \tag{1.11}$$

Pri rôznych pozíciách zdroja svetla a kamery voči sebe navzájom môžeme získať zjednodušenia rovníc 1.6 - 1.11. Ďalej uvedieme niektoré, najčastejšie situácie.

Vzdialený svetelný zdroj: Zdroj svetla zviera s objektom len veľmi malý uhol, prípadne lúče svetla vychádzajúce zo zdroja sú takmer rovnobežné a pretínajú sa až v nekonečne. Vtedy $A_r \bullet r_i = 0$, prípadne ak je zdroj svetla veľmi ďaleko, tak $A_r = 0$. Ak nahradíme r_i s kr_i a $k \to \infty$, tak potom $I_r = 0$, $E_r = 0$, $G_n = 0$. Vzťahy pre výpočet I_n

a E_n sa nemenia. (Pozri vzťahy 1.7, 1.9). $G_r = \left(\frac{1}{r_e}\right)(\hat{r_i} - G\hat{r_e})$ [4].

Vzdialená kamera: Kamera zviera malý uhol s objektom. Podobne ako pri vzdialenom svetelnom zdroji. Ak nahradíme r_e s kr_e a $k \rightarrow \inf$, tak potom I_r, I_n a E_n sa nemenia. (Pozri vzťahy 1.6, 1.7, 1.9). $E_r = 0$, $G_n = 0$ a $G_r = \left(\frac{1}{r_i}\right)(\hat{r_e} - G\hat{r_i})$ [4].

Vzdialený svetelný zdroj a vzdialená kamera: Toto je v praxi najviac využívaná situácia. $I_r = 0$, $E_r = 0$, $G_r = 0$, $G_n = 0$, I_n a E_n sa nemenia [4].

Svetelný zdroj na kamere: Teraz sa $r_i = r_e$, $I = E$ a $G = 1$. $I_r = E_r$, a to sa rovná vzťahu 1.6, alebo 1.8. Je jedno ktorý vzťah sa použije, keďže $r_i = r_e$. Podobne $I_n = E_n$, vzťahy na výpočet sú 1.7, 1.6. $G_r = 0$ a $G_n = 0$ [4].

Vzdialený svetelný zdroj na vzdialenej kamere: Ide o najjednoduchšiu situáciu. $I_r = E_r = G_r = 0$, $I_n = E_n$ a vzťahy na výpočet sú 1.7 alebo 1.9. $G_n = 0$ [4].

Rovnomerné osvetlenie: Je ekvivalentné situácii keď máme bodový svetelný zdroj na kamere, avšak je nutné zmeniť funkciu odrazivosti [4].

Keďže rovnica 1.3 je nelineárna parciálna diferenciálna rovnica prvého rádu môžeme ju napísať aj ako $F(x, y, z, p, q) = A(r)\phi(I, E, G) - b(r') = 0$. Z tejto rovnice poznáme $A(r)$, $\phi(I, E, G)$ a $b(r')$. Nepoznáme F_x, F_y, F_z, F_p, F_q. Z predchádzajúcich zistení vieme tieto neznáme získať z F_r a F_n, keď platí nasledovné.

$$F_r = A(r)\phi_r(I, E, G) + A_r(r)\phi(I, E, G) - b_r(r') \tag{1.12}$$

$$F_n = A(r)\phi_n(I, E, G) \tag{1.13}$$

Tiež platí, že $\phi_r(I, E, G) = \phi_{(I,E,G)}(I, E, G)_r$ a $\phi_n(I, E, G) = \phi_{(I,E,G)}(I, E, G)_n$. A zo vzťahov 1.6 - 1.11 vyplýva, že $(I, E, G)_r$ a $(I, E, G)_n$ sú nasledovné matice rozmeru 3×3 [4].

$$(I, E, G)_r = \begin{pmatrix} \left(\frac{1}{r_i}\right)(\hat{n} - I\hat{r_i}) \\ \left(\frac{1}{r_e}\right)(\hat{n} - E\hat{r_e}) \\ \left(\frac{1}{r_e}\right)(\hat{r_i} - G\hat{r_e}) + \left(\frac{1}{r_i}\right)(\hat{r_e} - G\hat{r_i}) \end{pmatrix} = \begin{pmatrix} \frac{1}{r_i} & -\frac{I}{r_i} & 0 \\ \frac{1}{r_e} & 0 & -\frac{E}{r_e} \\ 0 & \left(\frac{1}{r_e} - \frac{G}{r_i}\right) & \left(\frac{1}{r_i} - \frac{G}{r_e}\right) \end{pmatrix} \begin{pmatrix} \hat{n} \\ \hat{r_i} \\ \hat{r_e} \end{pmatrix}$$

$$(I,E,G)_n = \begin{pmatrix} \left(\frac{1}{n}\right)(\hat{r}_i - I\hat{n}) \\ \left(\frac{1}{n}\right)(\hat{r}_e - E\hat{n}) \\ 0 \end{pmatrix} = \begin{pmatrix} -\frac{I}{n} & \frac{1}{n} & 0 \\ -\frac{E}{n} & 0 & \frac{1}{n} \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \hat{n} \\ \hat{r}_i \\ \hat{r}_e \end{pmatrix}$$

V derivácii F_n už poznáme všetky premenné, no v F_r stále ostávajú dve neznáme $b_r(r')$ a A_r. Prvú z nich vypočítame podľa tohoto vzťahu $b_r(r') = b_{r'} \bullet b'_r$. A keď tento vzťah rozpíšeme dostávame

$$(bx, by, bz) = \left(\frac{f}{z}\right)\left(b_{x'}, b_{y'}, -\left[\left(\frac{x}{z}\right)b_{x'} + \left(\frac{y}{z}\right)b_{y'}\right]\right) \tag{1.14}$$

Pre pripomenutie f je kolmá vzdialenosť roviny obrazu od kamery (snímača, šošovky), $b(r')$ je intenzita osvetlenia nameraná v bode r' obrazu, a teda $b_{x'}$ a $b_{y'}$ sú zložky intenzity osvetlenia pozdĺž osí roviny obrazu x' a y'. Alebo inak, parciálne derivácie intenzity osvetlenia b v bode r' obrazu podľa x' a y'. Tieto hodnoty netreba počítať, získame ich priamo z obrazu. Druhá neznáma A_r sa dá považovať za konštantu v oblasti objektu na obraze, alebo klesá z druhou mocninou nasledovne. Ak $A = \left(\frac{r_l}{r_i}\right)^2$, tak $A_r = -2\left(\frac{r_l^2}{r_i^4}\right)r_i$ a r_i je vektor dopadajúceho svetla, a r_l je dĺžka vektora dopadajúceho svetla do singulárneho bodu [2][3][4].

1.1.3 Počiatočné podmienky SFS

Keďže existuje množstvo povrchov, ktoré by mohli byť riešením musíme vybrať jeden konkrétny. Ten určíme na základe počiatočnej krivky, ktorou musí prechádzať povrch. Zaveďme $x = x(t)$, $y = y(t)$, $z = z(t)$, potom hľadaná krivka musí spĺňať nasledovné.

$$\begin{aligned} z'(t) &= px'(t) + qy'(t) \\ F[x(t), y(t), z(t), p(t), q(t)] &= 0 \end{aligned} \tag{1.15}$$

Apostrof predstavuje deriváciu podľa t. Sústava nelineárnych rovníc 1.15 nám umožňuje nájsť $p(t)$ a $q(t)$ na počiatočnej krivke.

Počiatočnú krivku vypočítame priamo z obrazu pomocou singulárnych bodov. Singulárne body sú body s najväčším alebo najmenším jasom. Čiže sú to najjasnejšie alebo najtmavšie body. To, ktoré z nich vyberieme, na určenie počiatočnej krivky, závisí na funkcií odrazivosti.

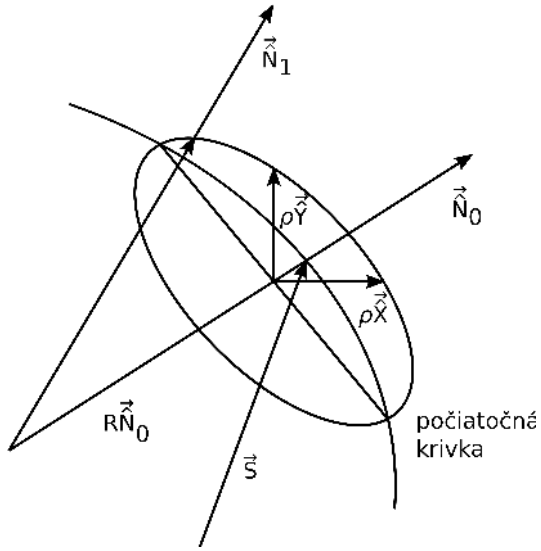
Veľkosti uhlov i a e, pre ktoré je odrazivosť jedinečným globálnym maximom alebo minimom zodpovedajú singulárnym bodom. Preto sa táto metóda nedá použiť, ak povrch neobsahuje také časti kde nevznikajú singulárne body. Tie nevzniknú preto, lebo neexistujú hodnoty i a e, ktoré nadobúdajú globálne minimum alebo maximum [3][4].

Po nájdení singulárnych bodov by sme mali zistiť ich vzdialenosť od kamery. Tú však zisťovať nemusíme, lebo nás zaujíma len relatívna vzdialenosť. Avšak riešenie sa samostatne nepohne zo singulárneho bodu. Musíme mu poskytnúť informáciu o zakrivení povrchu. O tom či je konvexný, alebo konkávny v danom singulárnom bode. V prípade, že singulárny bod je súčasne sedlovým bodom použijeme iný, blízky, singulárny bod pre výpočet počiatočnej krivky. Ak máme singulárny bod o ktorom vieme, že nie je sedlovým bodom a povrch v jeho okolí je konvexný (konkávny) odhadneme polomer zakrivenia R, pričom na presnosti nám veľmi nezáleží. Je tomu tak preto, že ρ, čo je vzdialenosť, o ktorú sa posunieme v smere od singulárneho bodu, je oveľa menšie ako R. A teda ak aj R nie je presné, pozícia počiatočnej krivky sa zmení len minimálne. S týmito informáciami môžeme vytvoriť na singulárnom bode guľový vrchlík nasledovným spôsobom 5 [4].

Zavedieme S ako vektor smerujúci od kamery k singulárnemu bodu. ρ zvolíme podľa uváženia. $\hat{N}_0$ je normála vypočítaná v singulárnom bode a stred guľového vrchlíka je v $S - R\hat{N}_0$. Ďalej

$$
\begin{aligned}
R_1^2 &= R^2 - r^2 \\
S_1 &= S + (R_1 - R)\hat{N}_0 \\
X &= \hat{y} \times \hat{N}_0, \hat{y} = (0, 1, 0) \\
Y &= \hat{N}_0 \times \hat{X} \\
T(t) &= \rho(\hat{X}\cos(2\pi t) + \hat{Y}\cos(2\pi t)), 0 \leq t < 1
\end{aligned}
$$

Body na počiatočnej kružnici vypočítame nasledovne $S_1 + T(t)$. A keďže potrebujeme približne určiť počiatočné hodnoty p a q vypočítame normálu N_1 takto $N_1(t) = R_1\hat{N}_1 + T(t)$. Stačia nám približné hodnoty, nakoľko celý postup je iteračný a tieto hodnoty sú len počiatočné.



Obr. 5: Konštrukcia počiatočnej krivky v okolí singulárneho bodu [4]

1.2 Pentlandov prístup k SFS

Pentlandov prístup k riešeniu problému SFS sa líši od predošlých prístupov. Pentland sa snaží získať tvar povrchu pomocou predpokladov založených čisto na funkcií odrazivosti, zatiaľ čo predošlé prístupy sú založené na predpokladoch o tvare povrchu. Tie vyžadujú buď veľký počet iterácií, alebo integrovanie. Aj keď výsledky týchto techník sú oveľa presnejšie.

Povrch z nech je funkciou (x, y), takže $z = z(x, y)$. Ďalej Pentland predpokladá, že povrch spĺňa tieto tri požiadavky:

- je Lambertovský (ideálne matný, ideálne difúzny) - odráža dopadajúce svetlo rovnomerne na všetky smery
- je osvetlený vzdialenými, bodovými zdrojmi svetla
- povrch sám na sebe nevytvára tiene

Ďalej $z < 0$ v oblasti, z ktorej sa snažíme získať tvar povrchu. Tiež predpokladá, že obraz vznikol ortografickou projekciou do roviny x, y. Nech τ je náklon zdroja svetla a σ je sklon zdroja svetla. Náklon τ je uhol, ktorý zviera vektor zdroja osvetlenia s osou x a sklon σ je uhol, ktorý zviera vektor zdroja osvetlenia s osou z. Potom $L = (x_L, y_L, z_L) = (\cos\tau \sin\sigma, \sin\tau \sin\sigma, \cos\sigma)$ je jednotkový vektor zdroja osvetlenia [5].

Pomocou týchto predpokladov potom Pentland definuje normalizovanú intenzitu obrazu I

v bode (x, y) ako:

$$I(x, y) = \frac{p \cos\tau \sin\sigma + q \sin\tau \sin\sigma + \cos\sigma}{\sqrt{(p^2, q^2 + 1)}} \quad , \tag{1.16}$$

čo je vlastne rovnica 1.1. Pričom platí, že p a q označujú zakrivenie povrchu v bode (x, y) v smere osí x a y. Vypočítame ich cez parciálne derivácie nasledovne: $p = \frac{\partial z}{\partial x}(x, y)$, $q = \frac{\partial z}{\partial y}(x, y)$. Taylorovým rozvojom rovnice 1.16 v $p = p_0$ a $q = q_0$ a ignorovaním členov vyššieho rádu dostávame

$$I(x, y) \approx \cos\sigma + p \cos\tau \sin\sigma + q \sin\tau \sin\sigma - \frac{\cos\sigma}{2}(p^2 + q^2) \quad . \tag{1.17}$$

Jedným z Pentlandových predpokladov o povrchu je to, že povrch je Lambertovský. Za tohoto predpokladu, a ak $p_0 = q_0 = 0$, tak rovnica 1.17 sa zredukuje na

$$I(x, y) \approx \cos\sigma + p \cos\tau \sin\sigma + q \sin\tau \sin\sigma \quad . \tag{1.18}$$

Ďalej Pentland aplikuje na obe strany predchádzajúcej rovnice Fourierovu transformáciu. Kde komplexné Fourierovo spektrum $F_z(f, \theta)$ povrchu $z(x, y)$ je rovnica 1.19. $m_z(f, \theta)$ v tejto rovnici je magnitúda na pozícii (f, θ) vo Fourierovej rovine. ϕ_z je fáza. Transformácie p a q sa vypočítajú podľa vzťahov 1.20 a 1.21 [5].

$$F_z(f, \theta) = m_z(f, \theta) e^{i\phi_z(f,\theta)} \tag{1.19}$$

$$F_p(f, \theta) = 2\pi f \cos\theta m_z(f, \theta) e^{i(\phi_z(f,\theta)+\pi/2)} \tag{1.20}$$

$$F_q(f, \theta) = 2\pi f \sin\theta m_z(f, \theta) e^{i(\phi_z(f,\theta)+\pi/2)} \tag{1.21}$$

Ak $|p|, |q| < 1$ a ak je zdroj svetla $\pm 30°$ od pozície kamery, tak v rovnici 1.17 prevládne lineárny člen. Inak je kvadratický člen zanedbateľný. V takom prípade Fourierova transformácia obrazu $I(x, y)$ je

$$F_I(f, \theta) = 2\pi f \sin\theta m_z(f, \theta) e^{i(\phi_z(f,\theta)+\pi/2)} [\cos\theta \cos\tau + \sin\theta \sin\tau] \tag{1.22}$$

V prípade keď prevládne kvadratický člen v rovnici 1.17, tak dôjde k efektu zdvojenia

frekvencií [5].

Z rovnice 1.22 vyplýva, že ak máme daný smer osvetlenia, vieme priamo určiť Fourierovu transformáciu povrchu F_z. Takže, ak uvažujeme Fourierovu transformáciu obrazu ako

$$F_I(f,\theta) = m_I(f,\theta)e^{i\phi_I(f,\theta)} \quad , \tag{1.23}$$

tak Fourierova transformácia povrchu je

$$F_z(f,\theta) = \frac{m_I(f,\theta)e^{i(\phi_I(f,\theta)-\pi/2)}}{2\pi f \sin\sigma[\cos\theta\cos\tau + \sin\theta\sin\tau]} \quad . \tag{1.24}$$

1.2.1 Odhad pozície zdroja osvetlenia

Vo všetkých doterajších rovniciach sme predpokladali, že poznáme smer kde sa nachádza zdroj osvetlenia. Avšak reálne sme smer osvetlenia nepoznali. Spôsob akým získať smer zdroja osvetlenia si ukážeme teraz.

Metóda je založená na odhade smeru osvetlenia z distribúcie derivácií obrazu ako funkcie smeru obrazu. Predpokladá sa, že smerovanie povrchu v každom jeho mieste je štatisticky rovnomerné. Smer osvetlenia získame z rovnice 1.25. (x_L^*, y_L^*) sú nenormalizované súradnice x a y vektora smeru osvetlenia, β je matica smerov (dx_i, dy_i) rozmeru $2 \times n$ a dI_i je stredná magnitúda $\frac{dI(x,y)}{dx_i} + \frac{dI(x,y)}{dy_i}$.

$$(x_L^*, y_L^*) = (\beta^T\beta)^{-1}\beta^T(dI_1, dI_2, \ldots, dI_n) \tag{1.25}$$

Ak nahradíme magnitúdu prvej derivácie magnitúdou m_i z Fourierovej transformácie, tak rovnicu 1.25 môžeme zapísať aj ako

$$(x_L^*, y_L^*) = (\beta^T\beta)^{-1}\beta^T(m_1, m_2, \ldots, m_n) \quad . \tag{1.26}$$

Keď sme vypočítali (x_L^*, y_L^*) môžeme vypočítať súradnice vektora zdroja osvetlenia podľa rovnice 1.27, ak

$$k = \sqrt{E(dI^2) - E(dI)^2}$$

a $E(dI)$ je hodnota $\frac{dI}{dx_i} + \frac{dI}{dy_i}$ cez všetky smery i.

$$L(x_L, y_L, z_L) = \left[\frac{x_L^*}{k}, \quad \frac{y_L^*}{k}, \quad \sqrt{1 - x_L^2 - y_l^2} \right] \tag{1.27}$$

1.3 Tsaiov a Shahov prístup k SFS

Tento prístup sa podobá Pentlandovému. Pentland sa snaží lineárne aproximovať mapu odrazivosti cez p a q pomocou Taylorovho rozvoja funkcie odrazivosti 1.16. Tsai a Shah lineárne aproximujú tiež mapu odrazivosti, ale cez hĺbku Z. Pre výpočet p a q používajú ich diskrétnu aproximáciu pomocou metódy konečných diferencií. Ďalším rozdielom je možnosť použiť túto metódu, s úpravami, aj na iné ako Lambertovské povrchy [7].

Tsai a Shah najprv svoju metódu použili na Lambertovských povrchoch. Tu platí pre funkciu odrazivosti rovnica 1.1. Pre pripomenutie vyzerá nasledovne.

$$\begin{aligned} E(x, y) &= R(p, q) \\ &= \frac{\cos(\sigma) + p\cos(\tau)\sin(\sigma) + q\sin(\sigma)\sin(\tau)}{\sqrt{1 + p^2 + q^2}} \end{aligned} \tag{1.28}$$

Ako už bolo spomenuté zložky gradientu obrazu p a q sa vypočítajú ich diskrétnou aproximáciou, takže:

$$p = \frac{\partial Z}{\partial x} = Z(x, y) - Z\,(x - 1, y) \quad , \tag{1.29}$$

$$q = \frac{\partial Z}{\partial y} = Z(x, y) - Z\,(x, y - 1) \quad . \tag{1.30}$$

Rovnicu 1.28 môžeme pre daný bod (x, y), obrazu E prepísať do tvaru:

$$0 = f(E(x, y), Z(x, y), Z(x - 1, y), Z(x, y - 1)) \tag{1.31}$$

$$= E(x, y) - R(Z(x, y) - Z(x - 1, y), Z(x, y) - Z(x, y - 1)) \quad . \tag{1.32}$$

Premenná Z v tejto rovnici označuje mapu hĺbok, teda vzdialenosti jednotlivých bodov od kamery. Lineárnou aproximáciou rovnice 1.32 pre mapu hĺbok Z^{n-1}, kde n označuje n-tú

iteráciu, dostávame nasledujúcu rovnicu [7].

$$
\begin{aligned}
0 &= f(E(x,y), Z(x,y), Z(x-1,y), Z(x,y-1)) \\
&\approx f(E(x,y), Z^{n-1}(x,y), Z^{n-1}(x-1,y), Z^{n-1}(x,y-1)) \\
&\quad + (Z(x,y) - Z^{n-1}(x,y)) \frac{\partial f(E(x,y), Z^{n-1}(x,y), Z^{n-1}(x-1,y), Z^{n-1}(x,y-1))}{\partial Z(x,y)} \\
&\quad + (Z(x-1,y) - Z^{n-1}(x-1,y)) \frac{\partial f(E(x,y), Z^{n-1}(x,y), Z^{n-1}(x-1,y), Z^{n-1}(x,y-1))}{\partial Z(x-1,y)} \\
&\quad + (Z(x,y-1) - Z^{n-1}(x,y-1)) \frac{\partial f(E(x,y), Z^{n-1}(x,y), Z^{n-1}(x-1,y), Z^{n-1}(x,y-1))}{\partial Z(x,y-1)}
\end{aligned}
\tag{1.33}
$$

Prípadne vektorový zápis, ak $a_{x,y} = \frac{\partial}{\partial Z(x,y)} f(E(x,y), Z^{n-1}(x,y), Z^{n-1}(x-1,y), Z^{n-1}(x,y-1))$ a

$$
\begin{aligned}
b_{x,y} &= -f(E(x,y), Z^{n-1}(x,y), Z^{n-1}(x-1,y), Z^{n-1}(x,y-1)) \\
&\quad + Z^{n-1}(x,y) * \frac{\partial}{\partial Z(x,y)} f(E(x,y), Z^{n-1}(x,y), Z^{n-1}(x-1,y), Z^{n-1}(x,y-1)) \\
&\quad + Z^{n-1}(x-1,y) * \frac{\partial}{\partial Z(x-1,y)} f\ (E(x,y), Z^{n-1}(x,y), Z^{n-1}(x-1,y), Z^{n-1}(x,y-1)) \\
&\quad + Z^{n-1}(x,y-1) * \frac{\partial}{\partial Z(x,y-1)} f(E(x,y), Z^{n-1}(x,y), Z^{n-1}(x-1,y), Z^{n-1}(x,y-1))
\end{aligned}
$$

je nasledujúca rovnica:

$$
\left(0, \ldots, a_{x,y-1}, 0, \ldots, a_{x-1,y}, a_{x,y}, 0, \ldots\right) \begin{pmatrix} Z_{1,1} \\ \vdots \\ Z_{x,y} \\ \vdots \\ Z_{N,N} \end{pmatrix} = b_{x,y} \quad . \tag{1.34}
$$

Z tejto rovnice vyplýva, že ak máme obrázok rozmeru $N \times N$, tak existuje N^2 takýchto rovníc, ktoré vytvoria lineárnu sústavu $AZ = B$. A je matica rozmeru $N^2 \times N^2$ a Z s B sú vektory rozmeru $N^2 \times 1$. Tsai a Shah na vyriešenie tejto sústavy použili Jakobiho iteratívnu metódu, ktorá výpočet zjednodušuje a hlavne zrýchľuje. Pri danej počiatočnej aproximácií Z^0 sa každá hodnota hĺbky rieši postupne v v každej iterácií. Takže napr. $Z(x,y)$ v n-tej iterácií sa vypočíta z hodnôt $Z^{n-1}(x,y)$ v predchádzajúcej iterácii pre všetky $Z(i,j)$, pre ktoré

platí, že $i \neq x$ a $j \neq y$. Keď nahradíme $Z^{n-1}(x-1,y)$ za $Z(x-1,y)$ a $Z^{n-1}(x,y-1)$ za $Z(x,y-1)$ v rovnici 1.33, tak sa táto rovnica zredukuje nasledovne [7].

$$\begin{aligned} 0 &= f(Z(x,y)) \\ &\approx f(Z^{n-1}(x,y)) + (Z(x,y) - Z^{n-1}(x,y))\frac{d}{dZ(x,y)} f(Z^{n-1}(x,y)) \quad . \end{aligned} \tag{1.35}$$

Pre výpočet mapy hĺbok v n-tej iterácii potom môžeme použiť vzťah 1.36 [7].

$$Z^n(x,y) = Z^{n-1}(x,y) + \frac{-f(Z^{n-1}(x,y))}{\frac{d}{dZ(x,y)} f(Z^{n-1}(x,y))} \tag{1.36}$$

Kde

$$\frac{df(Z^{n-1}(x,y)}{dZ(x,y)}) = -1 * \left(\frac{p_s + q_s}{\sqrt{p^2+q^2+1}\sqrt{p_s^2+q_s^2+1}} - \frac{(p+q)(pp_s+qq_s+1)}{\sqrt{(p^2+q^2+1)^3}\sqrt{p_s^2+q_s^2+1}} \right) \quad . \tag{1.37}$$

Počiatočná hodnota $Z^0(x,y) = 0$ a ostatné neznáme sa vypočítajú nasledovne:

$$p_s = \frac{\cos\tau \sin\sigma}{\cos\sigma} \quad , \quad q_s = \frac{\sin\tau \sin\sigma}{\cos\sigma} \quad .$$

2 Vlastná implementácia

Táto časť práce sa zaoberá vlastnou programovou implementáciou dvoch prístupov získavania tvarov z tieňovania. Prvý je Pentlandov a druhý Tsaiov a Shahov. Programy sú napísané v jazyku Java, s využitím knižnice OpenCV. Výsledné grafy sú vytvorené v programe Gnuplot.

Program tvorí niekoľko tried. Najpodstatnejšou z nich je trieda Sfs.java, v ktorej sa nachádzajú oba algoritmy, tak Pentlandov, ako Tsaiov a Shahov.

Oba programy boli vytvárané na zostave s len 4GB RAM pod OS Ubuntu. V prvotnej verzii algoritmov, najmä v prípade Tsaiovho a Shahovho prístupu dochádzalo k vysokej spotrebe pamäte, až na hranicu 3GB spotrebovanej RAM programom pri spracovaní obrázku pneumatiky o rozmere 1000×1000 pixelov, pri 200 iteráciách. To následne viedlo k swapovaniu pamäte na disk, čo spomaľovalo celú zostavu až na takú úroveň, že algoritmus nebol schopný skončiť v rozumnom čase (okolo 2 hodín). Operačný systém úplne vymrzol a nereagoval na nič. Preto sme pristúpili k zníženiu pamäťovej náročnosti najjednoduchším možným spôsobom, a to zmenšením počtu používaných premenných. Súčasne sa znížil počet iterácií v základnom nastavení programu na 90. Po tejto úprave bol algoritmus schopný úspešne skončiť do pár minút. Pre istotu bol ale zväčšený aj swap oddiel z pôvodných 2GB na 6GB.

Obe metódy majú spoločné vstupné a výstupné premenné. Spoločné sú v tom zmysle, že vstupom do obidvoch metód je matica hodnôt, ktoré reprezentujú obrázok a výstupom je taktiež matica hodnôt.

Vstupný obrázok, z ktorého chceme získať tvar z jeho tieňovania vieme previesť do takejto matice použitím OpenCV takto:

```
Mat m = Imgcodecs.imread(pathToImage,
                         Imgcodecs.CV_LOAD_IMAGE_GRAYSCALE);
```

Výstup je pre spracovanie v Gnuplot prevedený z reprezentácie maticou do CSV súboru. Tento súbor obsahuje 3 stĺpce oddelené tabulátorom na každom riadku. Prvý stĺpec označuje súradnicu x, druhý súradnicu y a tretí súradnicu z. Ukážka časti vygenerovaného CSV je na obrázku 6.

Ďalej prejdeme k opisu vybraných algoritmov. Pri každom z nich uvedieme pseudokód a

```
x  y   z
0  0  -4.705788612365723
0  1  -5.006088733673096
0  2  -4.105187892913818
0  3  -3.504586935043335
0  4  -4.405488014221191
0  5  -7.10819149017334
0  6  -7.708792209625244
0  7  -13.714798927307129
0  8  -27.828916549682617
```

Obr. 6: Ukážka formátovania výstupného CSV súboru.

krok po kroku prejdeme celým algorimom implementovaným v Jave. Každý krok bude sprevádzaný ukážkov zdrojového kódu a jeho opisom.

2.1 Implementácia Pentlandovho prístupu k SFS

Pentlandov prístupu k SFS môžeme zhrnúť do nasledujúceho pseudokódu:

1. Načítaj obrázok ako maticu hodnôt.
2. Vypočítaj sklon a náklon zdroja osvetlenia.
3. Vypočítaj Fourierovu transformáciu obrazu F_e.
4. Z Fourierovej transformácie obrazu vypočítaj Fourierovu transformáciu povrchu F_z.
5. Vypočítaj inverznú Fourierovu transformáciu povrchu.
6. Ulož výsledný povrch.

Keďže zo pseudokódu nie je úplne jasné ako jednotlivé kroky naprogramovať, tak ich teraz opíšeme podrobnejšie. Opis budú sprevádzať ukážky zdrojového kódu.

Ako už bolo spomenuté povinným vstupom je matica hodnôt reprezentujúca vstupný obrázok. Takýto obrázok je následne konvertovaný do odtieňov šedej a jednotlivé hodnoty pixelov sú pretypované na typ double, aby sa zvýšila presnosť výpočtu.

```
System.loadLibrary( Core.NATIVE_LIBRARY_NAME )
E = Utils.convertColorToGray64FC1(E);
```

Obrázok je následne normalizovaný tak, aby všetky pixely mali maximálnu hodnotu 1.

```
Core.divide(E, new Scalar(Core.minMaxLoc(E).maxVal), E);
```

V ďalšom kroku je nutné zistiť sklon σ a náklon τ zdroja svetla.

```
sceneParams = Utils.estimateSceneParameters(E);
```

Následne vypočítame optimálnu výšku a šírku obrazu pre Fourierovu transformáciu a chýbajúce pixely, čo do šírky a výšky, doplníme nulami. Takáto úprava sa robí preto, že Fourierova transformácia je najrýchlejšia vtedy, ak sú rozmery vstupného obrazu deliteľné dvomi, tromi alebo piatimi.

```
int optRows = Core.getOptimalDFTSize(E.rows());
int optCols = Core.getOptimalDFTSize(E.cols());
Core.copyMakeBorder(E, E, 0, optRows - E.rows(), 0,
                    optCols - E.cols(), 0, Scalar.all(0));
```

Teraz musíme vypočítať ω_x a ω_y, ktoré budeme používať pri výpočte Fourierovej transformácie povrchu z Fourierovej transformácie obrazu.

```
MeshGrid meshgrid = Utils.meshgrid(1, E.cols(), 1, E.rows());
Mat x = meshgrid.X,
    y = meshgrid.Y;
Mat wx = new Mat(),
    wy = new Mat();
Core.multiply(x, new Scalar(2*Math.PI/E.cols()), wx);
Core.multiply(y, new Scalar(2*Math.PI/E.rows()), wy);
```

Pred výpočtom Fourierovej transformácie obrazu je nutné pridať do obrazu ďalší kanál. Zatiaľ sme pracovali len s jedným, teraz budú dva. V prvom sa uloží reálna, v druhom imaginárna časť komplexného čísla.

```
ArrayList<Mat> channels = new ArrayList<Mat>();
channels.add(E);
channels.add(Utils.initZeros(E));
Core.merge(channels, E);
Core.dft(E, E);
```

Máme vypočítanú Fourierovu transformáciu obrazu, aby sme z nej dokázali vypočítať Fourierovu transformáciu povrchu musíme pred tým pridaný kanál oddeliť a s oboma pracovať nezávisle. Keďže Java ani OpenCV nedokáže priamo pracovať s komplexnými číslami v tvare $a + bi$, musíme všetky základné operácie (+, -, *, /) realizovať po častiach podľa vzťahov, ktoré pre jednotlivé operácie s komplexnými číslami platia. Nám bude stačiť súčet a delenie komplexných čísel. Súčet dvoch komplexných čísel vypočítame $(a+bi)+(c+di) =$

$(a+c)+(b+d)i$ a ich podiel je $\frac{a+bi}{c+di} = \left(\frac{ax+bd}{c^2+d^2}\right)+\left(\frac{bc-ad}{c^2+d^2}\right)i$. Treba si uvedomiť, že aj keď pracujeme s komplexnými číslami, tak tieto sú uložené vo forme matíc. Preto sa využívajú OpenCV metódy *Core.add*, *Core.divide*. Fourierovu transformáciu povrchu z Fourierovej transformácie obrazu potom vypočítame nasledovne.

$$F_z = \frac{F_e}{-i\omega_x \cos(tilt)\sin(slant) - i\omega_y \sin(tilt)\sin(slant)}$$

```
Core.split(E, channels);
Core.multiply(wx, new Scalar(Math.cos(sceneParams.tilt)*
   Math.sin(sceneParams.slant)*-1), wx); // -i*wx*cos(tilt)*sin(slant)
Core.multiply(wy, new Scalar(Math.sin(sceneParams.tilt)*
   Math.sin(sceneParams.slant)*-1), wy); // -i*wy*sin(tilt)*sin(slant)

// sucet len imaginarnych casti, realna cast = 0 nema zmysel ju pocitat
Mat tmp1 = new Mat();
Core.add(wx, wy, tmp1);
Core.multiply(channels.get(1), tmp1, wx);
Mat tmp12 = new Mat();
Core.pow(tmp1, 2, tmp12);
Core.multiply(channels.get(0), tmp1, wy);
Core.divide(wy, tmp12, wy);

//realna cast vypocitana vo wx
Core.divide(wx, tmp12, wx);

// imaginarna cast vypocitana vo wy
Core.multiply(wy, new Scalar(-1), wy);
```

Aby sme z Fourierovej transformácie povrchu dostali hodnoty vzdialeností alebo hĺbok povrchu musíme vypočítať inverznú Fourierovu transformáciu. Na to musíme opäť spojiť kanály reprezentujúce reálnu a imaginárnu časť komplexného čísla v matici.

```
channels = new ArrayList<Mat>();
channels.add(wx);
channels.add(wy);
Core.merge(channels, E);
Core.idft(E, E);
```

Aby sme boli schopný komplexné čísla v matici nejako rozumne zobraziť vypočítame ich veľkosti podľa vzorca $\sqrt{Re^2 + Im^2}$. Po tomto kroku už máme výslednú mapu hĺbok alebo vzdialeností. Čiže výsledný tvar povrchu získaný z jeho tieňovania.

```
channels = new ArrayList<Mat>();
Core.split(E, channels);
E = new Mat();
Mat ReS = new Mat(),
    ImS = new Mat();
Core.pow(channels.get(0), 2, ReS);
Core.pow(channels.get(1), 2, ImS);
Core.add(ReS, ImS, ReS);
Core.sqrt(ReS,E);
return E;
```

2.2 Implementácia Tsaiovho a Shahovho prístupu k SFS

Pseudokód pre Tsaiov a Shahov prístup je nasledovný.

1. Načítaj obrázok.
2. Vypočítaj sklon a náklon zdroja osvetlenia.
3. Inicializuj povrchu Z a derivácie povrchu podľa osí x a y na nulu.
4. Vypočítaj súradnice zdroja osvetlenia.
5. V cykle opakuj nasledujúce kroky kým nedosiahneš zadaný počet iterácií.
   a Vypočítaj mapu odrazivosti R.
   b Ak je niektorá z hodnôt mapy odrazivosti záporná priraď jej nulu.
   c Vypočítaj funkciu f.
   d Vypočítaj $\frac{df}{dZ}$.
   e Vypočítaj nový povrch Z.
   f Vypočítaj nové hodnoty p a q, derivácií povrchu podľa osí x a y.
6. Aplikuj prahovanie.
7. Ulož výsledný povrch.

Podobne ako v prípade implementácie Pentlandovho prístupu k SFS je vstupom matica hodnôt, ktorá reprezentuje vstupný obrázok. Tiež platí, že prvým krokom je konverzia do odtieňov šedej a pretypovanie na typ double. Taktiež sa vypočíta sklon a náklon zdroja svetla. Keďže tieto kroky sú rovnaké v oboch algoritmoch, aj ich kód je rovnaký, a teda nie je nutné ho znovu uvádzať. Avšak už nasledujúci krok je iný. V tomto kroku sa inicializujú počiatočné hodnoty matíc p, q a Z na nulu.

```
Mat p = Utils.initZeros(E);
Mat q = Utils.initZeros(E);
Mat Z~= Utils.initZeros(E);
```

Potom sa vypočíta normalizovaný smer osvetlenia.

```
double ix = Math.cos(sceneParams.tilt) * Math.tan(sceneParams.slant);
double iy = Math.sin(sceneParams.tilt) * Math.tan(sceneParams.slant);
```

Vytvoria sa ostatné potrebné premenné, uvádzané sú už po znížení ich počtu kvôli pamäťovej náročnosti.

```
Mat R = new Mat();
Mat p2q21 = new Mat();
Mat sqrtp2q21 = new Mat();
Mat f = new Mat();
Mat df_dZ = new Mat();
Mat tmp = new Mat();
Mat tmp1 = new Mat();
Mat tmp2 = new Mat();
```

Teraz začína cyklus v ktorom sa budú vykonávať jednotlivé iterácie. Ako prvé sa v každej iterácií vypočíta mapa odrazivosti R z hodnôt p, q predchádzajúcej iterácie a z hodnôt sklonu σ, a náklonu τ. Súčasne sa overí, či mapa odrazivosti obsahuje nejaké záporné hodnoty. Ak áno, tak tieto sú prepísané na nulu. Mapa odrazivosti musí byť všade nezáporná.

```
for (int k~= 1; k~<= maxIter; k++) {
   Core.multiply(p, new Scalar(Math.cos(sceneParams.tilt) *
   Math.sin(sceneParams.slant)), tmp1);
   Core.multiply(q, new Scalar(Math.sin(sceneParams.tilt) *
   Math.sin(sceneParams.slant)), tmp2);
   Core.add(tmp1, tmp2, tmp1);
   Core.add(tmp1, new Scalar(Math.cos(sceneParams.slant)), tmp1);
   Core.pow(p, 2, tmp);
   Core.pow(q, 2, tmp2);
   Core.add(tmp, tmp2, tmp);
   Core.add(tmp, new Scalar(1), p2q21);
   Core.sqrt(p2q21, sqrtp2q21);
   Core.divide(tmp1, sqrtp2q21, R);
   Core.max(R, new Scalar(0), R);
```

Ako ďalšie vypočítame funckiu f ako rozdiel medzi pôvodným obrazom E a mapou odrazi-

vosti R. $f = E - R$. Následne vypočítame deriváciu f podľa Z použitím tejto rovnice:

$$\frac{df}{dZ} = \frac{(p+q)(ixp+iyq+1)}{\sqrt{(1+p^2+q^2)^3}\sqrt{1+ix^2+iy^2}} - \frac{ix+iy}{\sqrt{1+p^2+q^2}\sqrt{1+ix^2+iy^2}} \quad .$$

```
Core.subtract(E, R, f);
Core.add(p, q, tmp);  // (p+q)
Core.multiply(p, new Scalar(ix), tmp1); // ix*p
Core.multiply(p, new Scalar(iy), tmp2); // iy*q
Core.add(tmp1, tmp2, tmp1); // ix*p + iy*q
Core.add(tmp1, new Scalar(1), tmp1); // ix*p + iy*p +1
Core.multiply(tmp, tmp1, tmp); // ((p+q) *(ix*p + iy*q + 1))
Core.pow(p2q21, 3, tmp1); // (1 + p^2 + q^2)^3
Core.sqrt(tmp1, tmp1); // sqrt((1 + p^2 + q^2)^3)
Scalar sqrtix2iy21 = new Scalar(Math.sqrt(1 + Math.pow(ix, 2) + Math.
    pow(iy,2))); // sqrt(1 + ix^2 + iy^2)
Core.multiply(tmp1, sqrtix2iy21, tmp1); //(sqrt((1 + p^2 + q^2)^3)*
    sqrt(1 + ix^2 + iy^2))
Core.divide(tmp, tmp1, tmp);
Core.multiply(sqrtp2q21, sqrtix2iy21, tmp1); //(sqrt(1 + p^2 + q^2)*
    sqrt(1 + ix^2 + iy^2))
Core.divide(ix+iy, tmp1, tmp2); //(ix+iy)./(sqrt(1 + p^2 + q^2)* sqrt(1
     + ix^2 + iy^2));
Core.subtract(tmp, tmp2, df_dZ);
```

Vypočítame novú mapu hĺbok $Z = Z - df/dZ$. Aby sme predišli deleniu nulou pripočítame k derivácii f podľa Z veľmi malé kladné číslo blízke nule.

```
Core.add(df_dZ, new Scalar(Math.ulp(1)), tmp);  // aby sa nedelilo 0
Core.divide(f, tmp, tmp); // f/(df_dZ)
Core.subtract(Z, tmp, Z); // Z~= Z~- f/(df_dZ)
```

Teraz môžeme vypočítať nové p a q. Vytvoríme si dve nové mapy hĺbok z vypočítanej mapy hĺbok. A to tak, že najprv posunieme Z o jeden stĺpec do prava a hodnotám v prvom stĺpci zľava priradíme nulu. Takto získame posun, zmenu, deriváciu Z v smere osi x. Podobne získame posun Z v smere osi y. Nebudeme však posúvať stĺpce, ale riadky a do prvého riadku zhora zapíšeme nuly. Potom $p = Z - Z_x$ a $q = Z - Z_y$.

```
Mat Zx = Mat.zeros(1, E.cols(), CvType.CV_64FC1),
    Zy = Mat.zeros(E.rows(), 1, CvType.CV_64FC1);
List<Mat> mlv = new ArrayList<>();
mlv.add(Zx);
mlv.add(Z.rowRange(1,Z.rows()));
List<Mat> mlh = new ArrayList<>();
mlh.add(Zy);
mlh.add(Z.colRange(1, Z.cols()));
```

```
Core.vconcat(mlv, Zx);
Core.hconcat(mlh, Zy);

// vypocet novych p a q
Core.subtract(Z, Zx, p);
Core.subtract(Z, Zy, q);
}
```

Po vykonaní zadaného počtu iterácií získame tvar povrchu z tieňovania obrazu. Pri experimentovaní s týmto algoritmom sme zistili, že je vhodné použiť prahovanie výsledného povrchu, keďže niektoré hodnoty skresľovali výsledný graf. A to tak, že extrémne zväčšovali rozsah osi z potrebný pre zobrazenie povrchu. Hodnoty povrchu tak tvorili len zlomok tohoto rozsahu a splynuli v jednoliatu rovinu. Po aplikovaní prahovania s dolným prahom -1000 a horným prahom 1000 sa povrch zobrazil podľa očakávaní pri všetkých použitých vzorkách pneumatík.

```
Imgproc.threshold(Z, Z, tresh.lower, 0, Imgproc.THRESH_TOZERO);
Imgproc.threshold(Z, Z, tresh.upper, 0, Imgproc.THRESH_TOZERO_INV);
```

3 Výsledky

V tejto časti práce ukážeme výsledky, ktoré sme dosiahli pri experimentovaní s vytvorenými programami nad sadou vzoriek dezénov pneumatík a podrážok topánok. Experimenty vyhodnotíme. Pri dezénoch pneumatík sa používajú názvy ako žliabky a figúry. V prípade žliabkov je asi jasné, že sa jedná o drážky v dezéne a figúry sú vyvýšené plôšky, ktoré sa dotýkajú vozovky.

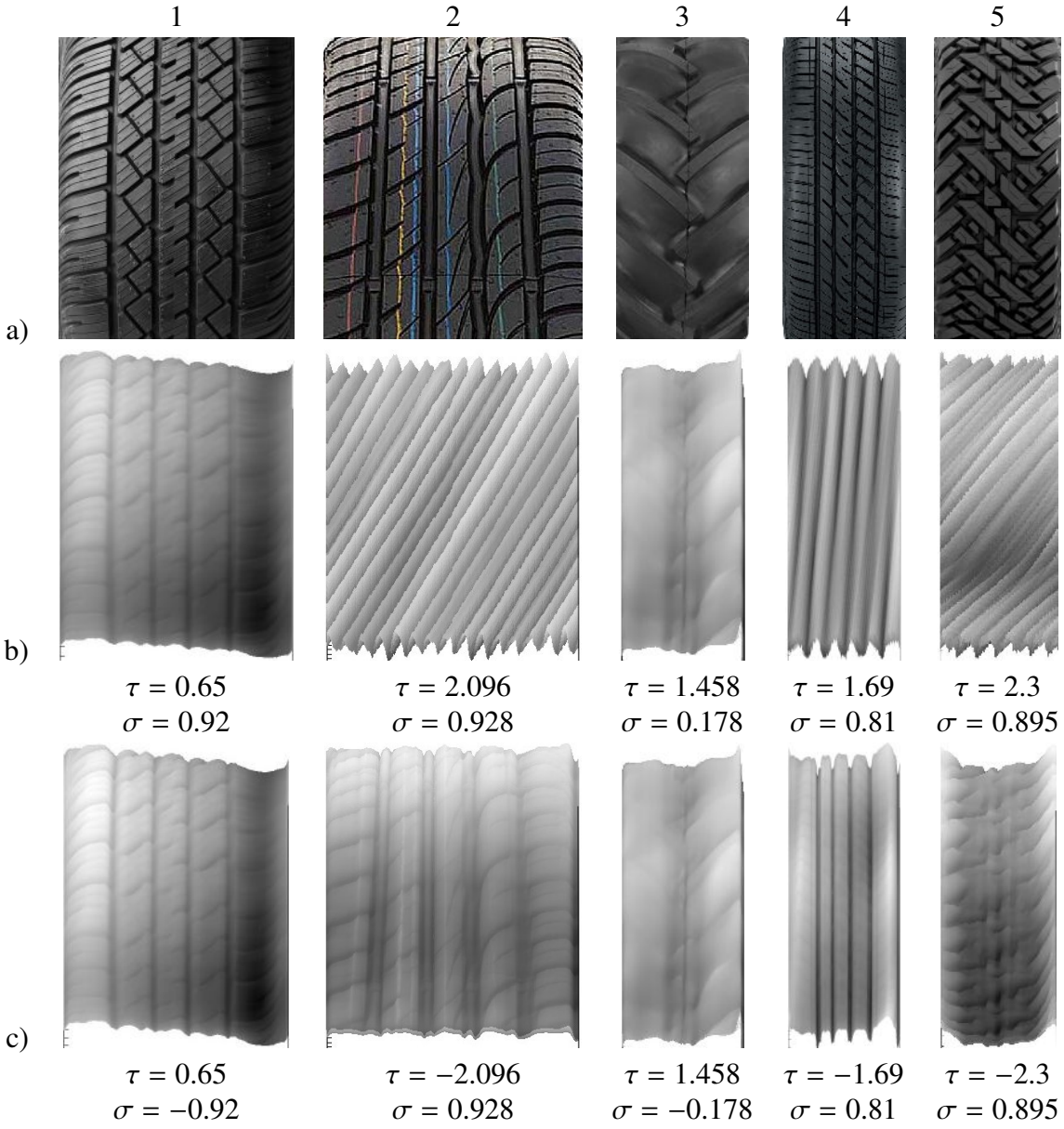
3.1 Pentlandov prístup

Pentlandov prístup sme implementovali do programu. Pomocou tohoto programu sme získali výsledky, ktoré sa teraz pokúsime analyzovať.

Na obrázkoch 7a sú zobrazené použité vzorky dezénov pneumatík, z ktorých sme sa snažili získať tvar z ich tieňovania. Prvotné výsledky môžeme vidieť na obrázkoch 7b. Na väčšine z týchto obrázkov (okrem 7b1 a 7b3) môžeme pozorovať, že tvar, ktorý bol nájdený, nie je taký ako vidíme na zodpovedajúcich obrázkoch 7a. Nachádza sa tu len neurčité vlnenie. Pravdepodobne kombinácia funkcií sínus a kosínus. Je možné, že tento jav je spôsobený zlým odhadom sklonu a náklonu zdroja svetla, prípadne vysokou úrovňou šumu. Už spomínané obrázky 7b1 a 7b3 však nevyzerajú vôbec zle. Na prvý pohľad môžeme vidieť, že tvar dezénu približne zodpovedá tomu, aký je na obrázkoch 7a1 a 7b1. Väčšina žliabkov a figúr dezénu je správne lokalizovaná. Tiež je vidieť aj výškový rozdiel medzi žliabkami a figúrami dezénu. Nie je síce veľmi zreteľný, ale je tam. Najmä v prípade vzorky č.1 (obrázok 7b1). Pri ďalších experimentoch sa zameriame hlavne na obrázky dezénov pneumatík, kde Pentlandova metóda nebola úspešná. No aj naďalej budeme pracovať so všetkými vzorkami dezénov. Budeme sledovať aký vplyv na výsledný tvar budú mať zmeny, ktoré na týchto vzorkách alebo v parametroch nášho programu urobíme.

Najprv sme skúsili zmeniť sklon a náklon zdroja osvetlenia. Konkrétne sme použili pôvodné hodnoty, ktoré sme upravili prenásobením -1. Čiže pozícia svetla sa takto preklopila buď cez os x, alebo os z. Pri všetkých vzorkách bolo takto možné nájsť pozíciu svetla, kedy sa namiesto neurčitého vlnenia objavil tvar dezénu. Pozícia svetla je uvedená vždy pod obrázkom na obrázkoch 7c. Tvar dezénu opäť len približne zodpovedá tomu skutočnému. Žliabky aj figúry sú väčšinou správne lokalizované, no výškové rozdiely nie sú až tak zre-

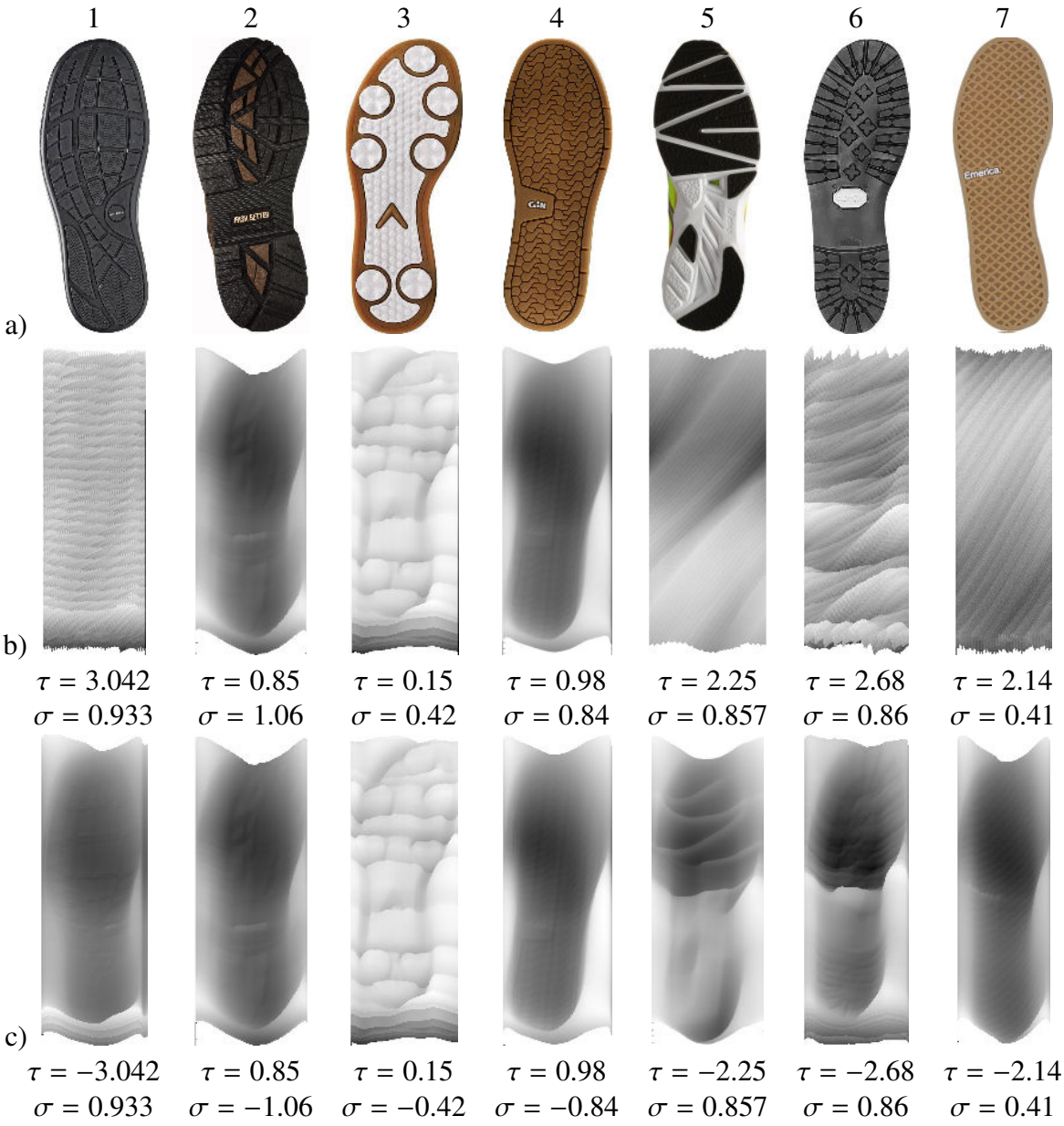
teľné. Na obrázkoch 7c3 – 5 môžeme pozorovať, že tvar dezénu nebol získaný správne. Mnoho figúr buď chýba úplne, alebo sú len naznačené. Podobne žliabky sa dajú len ťažko identifikovať. V prípade obrázka 7c4 sú jasne viditeľné len vertikálne žliabky, ktoré sa tiahnu celým dezénom.



Obr. 7: Výsledky aplikácie Pentlandovho prístupu na dezény pneumatík.

Keďže sa naša práca, okrem získavania tvaru z dezénu pneumatík, zaoberá aj získavaním tvaru z podrážok topánok, tak sme experimentovali aj s nimi. Výsledky môžeme vidieť na obrázku 8. Obrázky 8a zobrazujú použité vzorky podrážok. Podobne ako pri pneumatikách nie všetky výsledky boli správne. Len zo vzoriek 2, 3 a 4 (obrázky 8b2 – 4) sme získali tvar podrážky na prvý pokus. Tento ale nie je detailný a množstvo drážok a vyvýšených

plôch úplne chýba. Na obrázkoch 8b1, 8b5, 8b6 a 8b7 opäť pozorujeme len akési vlnenie. Po zmene sklonu alebo náklonu zdroja osvetlenia sme aj z týchto vzoriek dokázali získať tvar podrážky. Avšak drážky sú nevýrazné, podobne ako vyvýšené plôšky. Takže určitý tvar, ktorý zodpovedá skutočnému, môžeme pozorovať, ale chýbajú mu detaily.



Obr. 8: Výsledky aplikácie Pentlandovho prístupu na podrážky topánok.

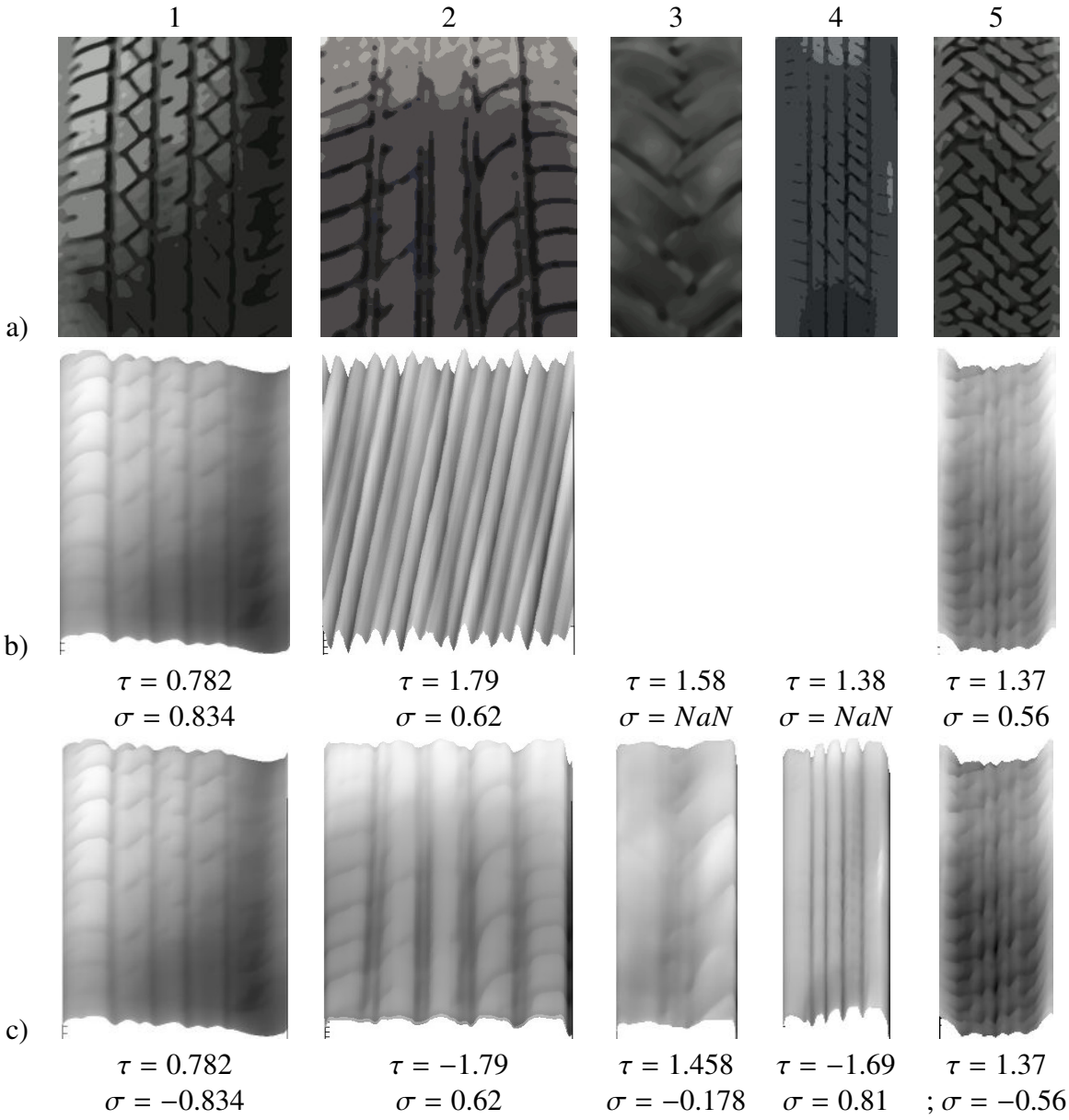
V ďalšom experimente sme sa snažili znížiť množstvo detailov, preto sme vzorky manuálne upravili. Upravené vzorky môžeme vidieť na obrázkoch 9a. Výsledky bez ďalších úprav na obrázkoch 9b. Len na vzorke 1 môžeme vidieť, že tvar získaného povrchu približne zodpovedá skutočnému tvaru. Oproti obrázku 7b1 si môžeme všimnúť, že sa stratili niektoré žliabky. V prípade vzorky 2 (obrázok 9b2) môžeme opäť vidieť len neurčité vlnenie. Vzorka

5 (obrázok 9b5) sa ale oproti výsledkom bez akejkoľvek úpravy (obrázok 7b5) zmenila a namiesto vlnenia tu môžeme vidieť náznak tvaru dezénu. Jedná sa skutočne len o náznak, lebo väčšina figúr chýba a rovnako chýbajú aj žliabky. Vzorky 3 a 4 sa nepodarilo získať bez ďalších úprav. Sklon zdroja svetla nebol v obidvoch prípadoch vypočítaný, a teda tvar povrchu nebol nájdený. Upozorňujeme, že to nie je chyba. Pri manuálnom zadaní približnej pozície zdroja svetla sme boli schopný získať výsledky. Použili sme pred tým zistenú pozíciu zdroja svetla z výsledkov na obrázkoch 8c3 a 8c4. Takto získané výsledky sú zobrazené na obrázkoch 10c3 a 10c4. Rovnako sme postupovali aj v prípade ostatných vzoriek a získali sme tak výsledky na obrázkoch 8c1, 8c2 a 8c5. Na nich si môžeme všimnúť, že sú takmer zhodné s tými, ktoré sme získali rovnakým spôsobom zo vzoriek bez úpravy. Niektoré žliabky chýbajú alebo nie sú menej výrazné. Preto figúry, ktoré by mali byť rozdelené týmito žliabkami sú teraz prepojené a tvoria na mnohých miestach jednoliate plochy.

V prípade vzoriek podrážok sme postupovali rovnako ako v prípade dezénov a znížili sme množstvo detailov, ktoré obsahujú. Takto upravené vzorky môžeme vidieť na obrázkoch 10a. Prejdime k dosiahnutým výsledkom. Na vzorkách 1, 2 a 6 opäť pozorujeme iba vlnenie. Po zmene hodnôt sklonu, prípadne náklonu zdroja osvetlenia môžeme na obrázkoch 10c vidieť, že sa tvar podrážky objavil. No v prípade všetkých vzoriek je len slabo viditeľný. Najviditeľnejšie sú hlavne tie drážky a vyvýšené plochy, ktoré sú dostatočne viditeľné aj na pôvodných vzorkách. Napr. na obrázku 10c7 je jasne viditeľná mriežka, ktorá tvorí podrážku. Podobne na obrázku 10c3 sú viditeľné kruhy po obvode podrážky.

3.2 Tsaiov a Shahov prístup

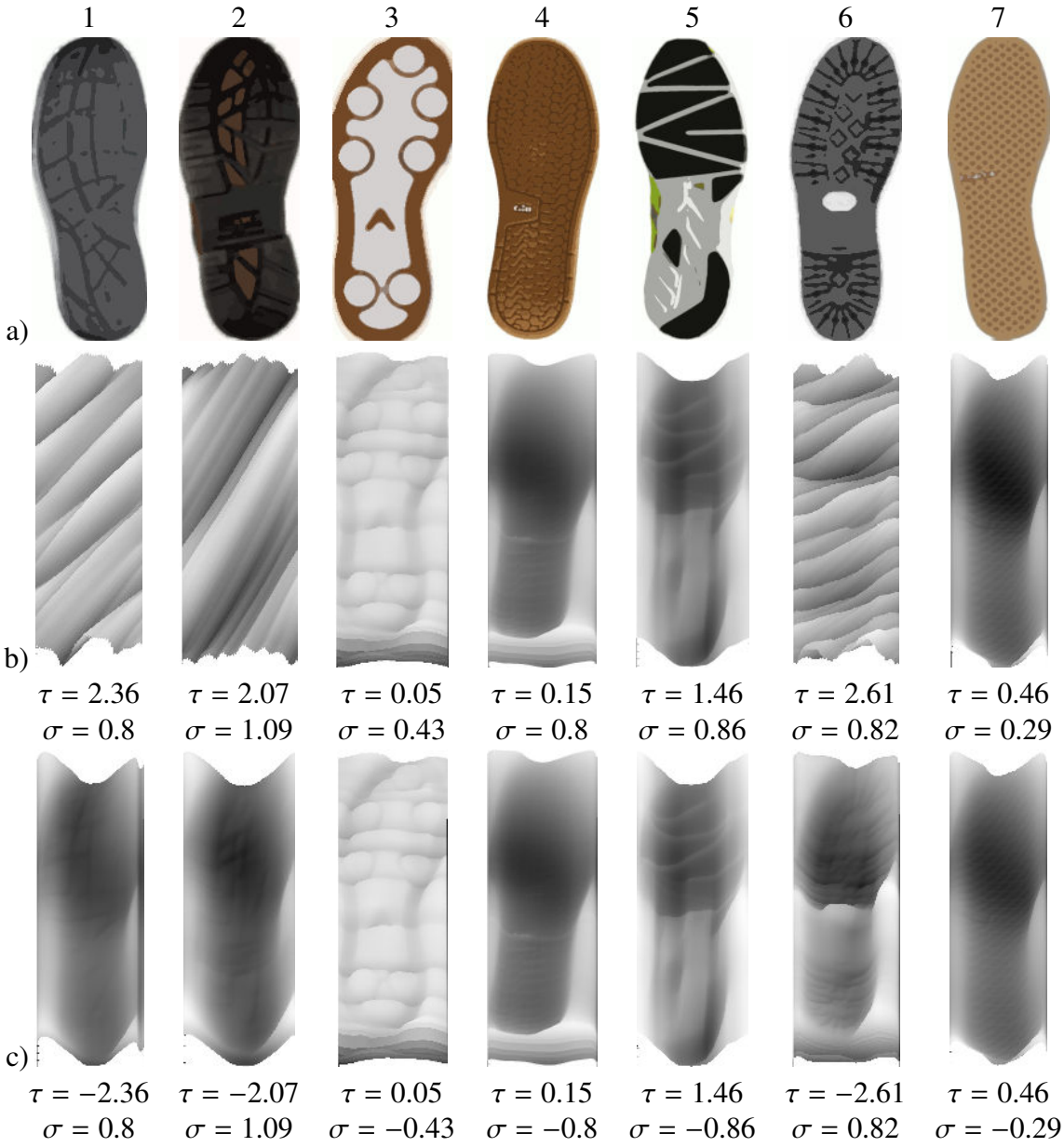
Podobne ako v prípade Pentlandovho prístupu aj tu sme použili rovnakú sadu vzoriek dezénov pneumatík. Pre lepšiu názornosť sa nachádzajú aj na obrázkoch 11a. Aplikovaním programu vytvoreného na základe Tsaiovho a Shahovho prístupu, na tieto vzorky, sme získali výsledky na obrázkoch 11b. Ako môžeme vidieť všetky žliabky a figúry na všetkých vzorkách sú správne lokalizované. Taktiež je zreteľný výškový rozdiel medzi žliabkami a figúrami. Aj keď výsledky sú zatiaľ uspokojivé pokúsime sa ich zlepšiť. Postupne budeme zvyšovať počet iterácií z pôvodných 90 na 150 a 200 pričom budeme sledovať vzniknuté povrchy jednotlivých vzoriek dezénov.



Obr. 9: Výsledky aplikácie Pentlandovho prístupu na upravené dezény pneumatík .

Ako je možné vidieť na obrázkoch 11c dezény sú po 150 iteráciách na pohľad svetlejšie. Okraje žliabkov a figúr sú hladšie a lepšie rozlíšiteľné. V žliabkoch sa nachádza menej tieňov, ktoré vrhajú figúry. Celkovo teda zvýšenie počtu iterácií zlepšilo výsledok. Iba v prípade obrázka 11c5 sme zaznamenali že do popredia vystúpili žliabky namiesto figúr. Tento jav môže byť spôsobený tým, že pri technikách získavania tvaru z tieňovania, algoritmus nevie rozlíšiť, medzi konvexným a konkávnym povrchom. Takže niektoré povrchy získame akoby vyvrátené z vnútra von. S rovnakým javom sa ešte stretneme aj pri ďalších výsledkoch.

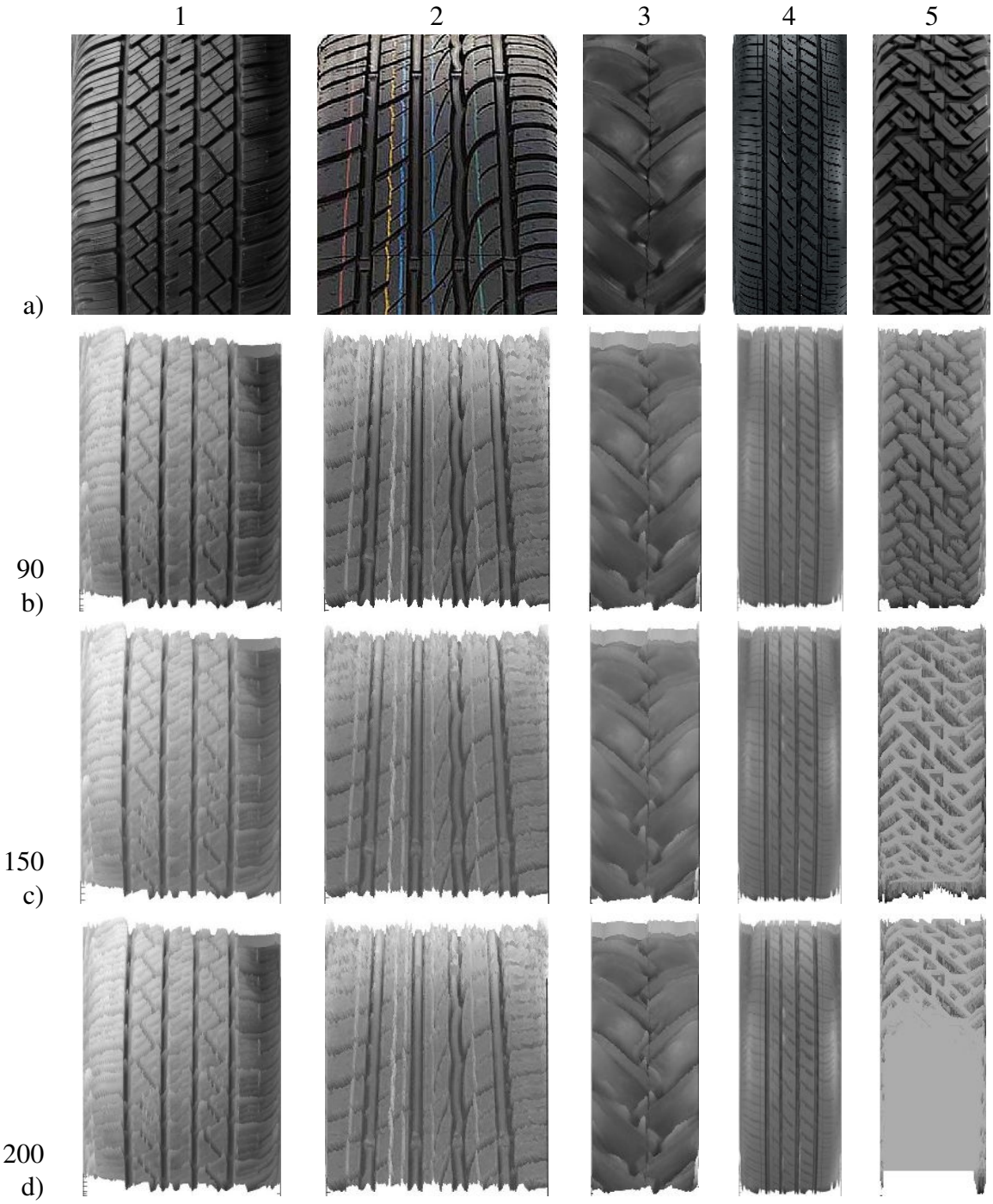
Obrázky 11d zobrazujú výsledky po 200 iteráciách. Výsledky sú v podstate rovnaké ako pri



Obr. 10: Výsledky aplikácie Pentlandovho prístupu na upravené podrážky topánok.

150 iteráciách, aj keď záleží od vzorky. Určite najhorší výsledok sme dosiahli v prípade obrázku 11d5. Tu sme stratili veľkú časť tvaru dezénu. Pri ďalšom zvyšovaní počtu iterácií sa tvar dezénu stratil úplne, tieto výsledky už ale neuvádzame.

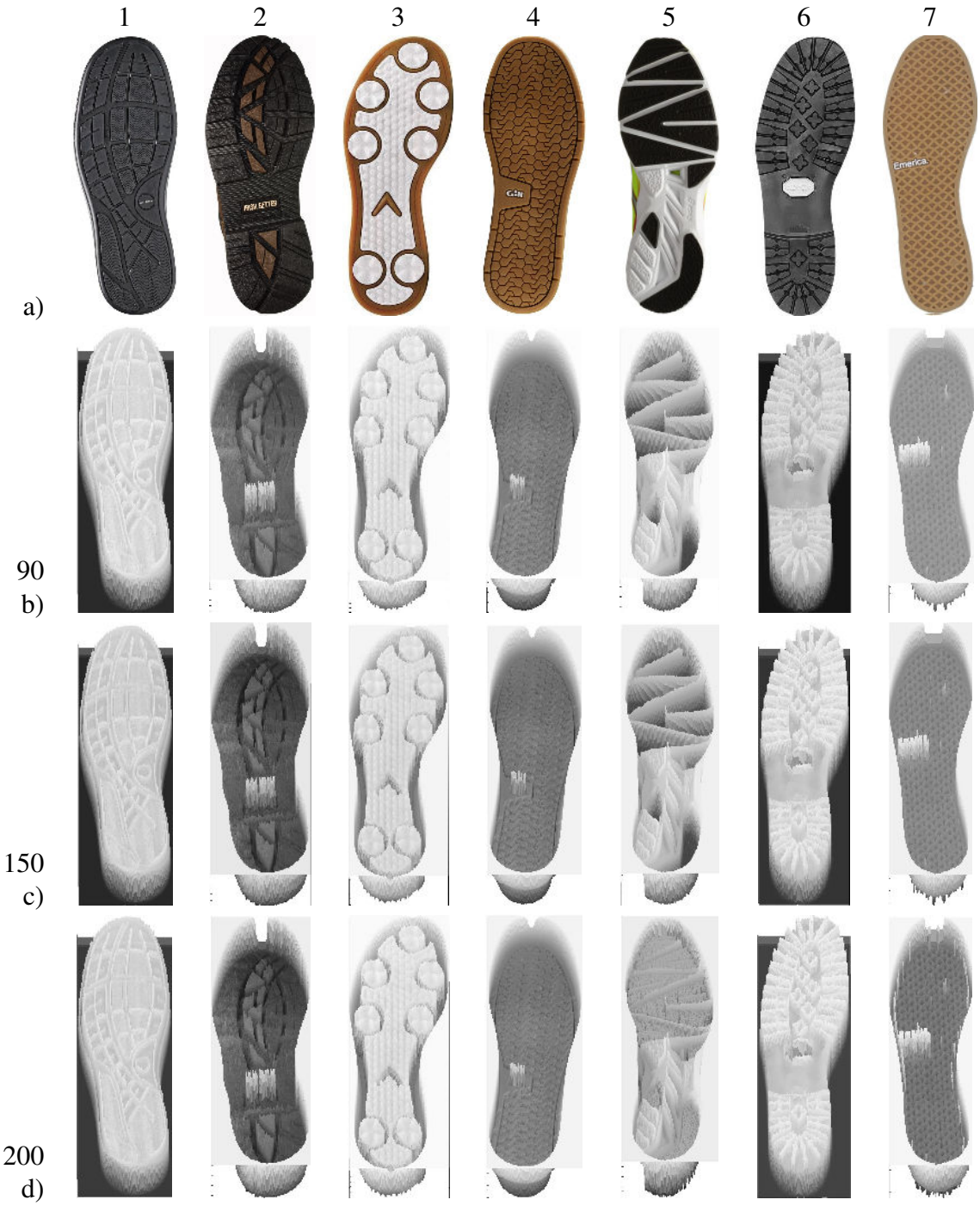
Ďalej sa budeme zaoberať experimentami so získavaním tvaru z podrážok topánok. Výsledky môžeme vidieť na obrázkoch 12b – d. V prípade vzoriek 1 a 6 došlo k javu, kedy sa povrch vyvrátil z vnútra von, podobne ako v prípade piatej vzorky dezénu (obrázky 11c5 a 11d5). Tvar dezénu vzoriek sa zvyšovaním počtu iterácií nemenil, ako môžeme vidieť na obrázkoch 11b – d. Súčasne na týchto obrázkoch môžeme pozorovať, že všetky drážky sú jasne vi-



Obr. 11: Výsledky aplikácie Tsaiovho a Shahovho prístupu na dezény pneumatík.

diteľné, podobne ako vyvýšené plôšky. Tiež si môžeme všimnúť, že v prípade podrážok s menšou hĺbkou drážok sa táto hĺbka premietla aj do výsledného tvaru podrážky (napr. porovnanie vzoriek 1 a 2 cez všetky iterácie). V prípade vzorky 5 (obrázky 11b5, 11c5 a 11d5) môžeme pozorovať, že zvyšovaním počtu iterácií sa postupne menilo množstvo zachytených detailov podrážky tejto vzorky. Pri 90 a 150 iteráciách sú jasne viditeľné len tie najviditeľnejšie drážky. No už pri 200 iteráciách sa do tvaru podrážky premietli aj menej

viditeľné priehlbiny na jej povrchu.



Obr. 12: Výsledky aplikácie Tsaiovho a Shahovho prístupu na podrážky topánok.

Záver

Cieľom tejto práce bolo zistiť, či je možné využiť techniky získavania tvaru z tieňovania na analýzu obrázkov dezénov pneumatík a podrážok topánok.

V teoretickej časti práce sme zhrnuli niekoľko prístupov, ktoré sa používajú keď potrebujeme získať tvar povrchu z jeho tieňovania. S potrebným teoretickým základom sme sa v druhej kapitole snažili implementovať Pentlandov a Tsaiov a Shahov prístup do programu. To sa nám aj podarilo a výsledný program vo forme zdrojového kódu, aj s jeho opisom, sme uviedli v druhej kapitole. Následne sme experimentovali s vytvoreným programom a sadou vzoriek dezénov pneumatík a podrážok topánok. Experimenty sme prezentovali v tretej kapitole. Celkovo z týchto experimentov vyplynulo, že techniky získavania tvaru z tieňovania sa dajú celkom úspešne použiť. Tak na získanie tvaru dezénu pneumatiky, ako na získanie tvaru podrážky topánky. Ako lepší z dvojice prístupov, ktoré sme implementovali sa javí byť Tsaiov a Shahov prístup. Nakoľko pri všetkých použitých vzorkách sme boli schopný pomocou tohoto prístupu získať tvar povrchu danej vzorky, či už šlo o dezény pneumatík, alebo podrážky topánok. Pentlandov prístup bol veľmi závislý na presnom určení pozície zdroja osvetlenia. V prípadoch, kedy výpočet približného smeru zdroja osvetlenia bol správny, alebo sme ho upravili manuálne, sme aj pomocou tohoto prístupu získali tvar povrchu použitých vzoriek. Tento povrch bol ale menej výrazný, výškové rozdiely neboli tak zreteľné ako v prípade Tsaiovho a Shahovho prístupu. Avšak jeho výhodou bola oveľa nižšia pamäťová a nižšia časová náročnosť.

Prácou do budúcna by mohlo byť zlepšenie najmä pamäťovej náročnosti programu implementujúceho Tsaiov a Shahov prístup. Prípadne vylepšenie algoritmu, ktorý počíta približný smer zdroja osvetlenia.

Zoznam bibliografických odkazov

[1] Blair, Z.: *Towards automatic 3D reconstruction of pitched roofs in monocular satellite/aerial images*. Diplomová práca, Simon Fraser University, 2012.

[2] Horn, B. K.: Understanding Image Intensities. In *Readings in Computer Vision*, úprava M. A. Fischler; O. Firschein, San Francisco (CA): Morgan Kaufmann, 1987, ISBN 978-0-08-051581-6, s. 45 – 60, doi:http://dx.doi.org/10.1016/B978-0-08-051581-6.50011-8.
URL http://www.sciencedirect.com/science/article/pii/B9780080515816500118

[3] Horn, B. K. P.: Shape From Shading: A Method for Obtaining the Shape of a Smooth Opaque Object From One View. Technická správa, Department of Electrical Engineering, Massachusetts Institue of Technology, Cambridge, MA, USA, November 1970.
URL http://people.csail.mit.edu/bkph/AIM/AITR-232-OPT.pdf

[4] Horn, B. K. P.: *Shape from Shading*. Cambridge, MA, USA: MIT Press, 1989, ISBN 0-262-08183-0, s. 123–171.
URL http://dl.acm.org/citation.cfm?id=93871.93877

[5] Pentland, A.: Shape Information From Shading: A Theory About Human Perception. In *Computer Vision., Second International Conference on*, Dec 1988, s. 404–413, doi: 10.1109/CCV.1988.590017.

[6] Pentland, A. P.: Local Shading Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, ročník PAMI-6, č. 2, March 1984: s. 170–187, ISSN 0162-8828, doi:10.1109/TPAMI.1984.4767501.

[7] Ping-Sing, T.; Shah, M.: Shape from shading using linear approximation. *Image and Vision Computing*, ročník 12, č. 8, 1994: s. 487 – 498, ISSN 0262-8856, doi: http://dx.doi.org/10.1016/0262-8856(94)90002-7.
URL http://www.sciencedirect.com/science/article/pii/0262885694900027

[8] Prados, E.; Faugeras, O.: *Handbook of Mathematical Models in Computer Vision*, kapitola Shape From Shading. Boston, MA: Springer, 2006, ISBN 978-0-387-28831-4, s.

375–388, doi:10.1007/0-387-28831-7_23.
URL http://dx.doi.org/10.1007/0-387-28831-7_23

[9] Introduction to Recovering Scene Geometry. In *Readings in Computer Vision*, úprava M. A. Fischler; O. Firschein, San Francisco (CA): Morgan Kaufmann, 1987, ISBN 978-0-08-051581-6, s. 13 – 20, doi:http://dx.doi.org/10.1016/B978-0-08-051581-6.50007-6.
URL http://www.sciencedirect.com/science/article/pii/B9780080515816500076

Prílohy

Príloha A

Systémová dokumentácia

Príloha B

Používateľská príručka

Príloha C

Sprievodné CD/DVD

Obsah sprievodného CD/DVD

```
.
├── sfs.bat - spúšťací skript pre Windows
├── sfs.jar - program
├── sfs.sh - spúšťací skript pre Linux
└── sfs.zip - zabalený zdrojový kód programu
    └── SFS
        ├── .classpath
        ├── .project
        └── src
            ├── Main.class
            ├── MeshGrid.class
            ├── SceneParameters.class
            ├── Sfs.class
            ├── SfsType.class
            ├── Treshold.class
            └── Utils.class
```