

Obrazová analýza radiálně symetrických vzorků  
Image analysis of radial symmetrical samples

Tomáš Dřímál

---

Bakalářská práce  
2008

 Univerzita Tomáše Bati ve Zlíně  
Fakulta aplikované informatiky

---

Univerzita Tomáše Bati ve Zlíně  
Fakulta aplikované informatiky  
Ústav aplikované informatiky  
akademický rok: 2007/2008

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Tomáš DRÍMAL**  
Studijní program: **B 3902 Inženýrská informatika**  
Studijní obor: **Informační technologie**

Téma práce: **Obrazová analýza radiálně symetrických vzorků**

Zásady pro vypracování:

1. Literární rešerše odpovídajících metod obrazové analýzy.
2. Návrh algoritmů.
3. Vývoj programu pro obrazovou analýzu v jazyce Delphi.
4. Analýza vzorků.

Rozsah práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. Ponížil P., Pavlínek V., Kitano T., Dřímal T.: Image analysis of radial symmetrical samples. In: Revetria R. et al. (eds.) System science and simulation in engineering, proceedings of 6th WSEAS international conference on System science and simulation in engineering (ICOSSSE 07), Venice, Italy, November 21-23, 2006, p. 330-333. ISBN: 978-960-6766-14-5.
2. Hader D.P.: Image Analysis: Methods and Applications, CRC Pres, 2001, ISBN 0-84-930239-0.
3. Gonzalez R.C., Woods R.E.: Digital Image Processing, Prentice Hall, 2007.

Vedoucí bakalářské práce:

**doc. RNDr. Petr Ponížil, Ph.D.**

Ústav fyziky a mater. inženýrství

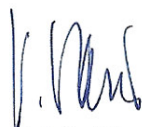
Datum zadání bakalářské práce:

**20. února 2008**

Termín odevzdání bakalářské práce:

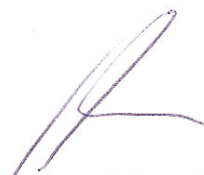
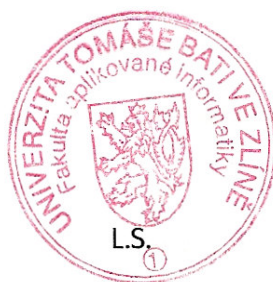
**5. května 2008**

Ve Zlíně dne 20. února 2008



prof. Ing. Vladimír Vašek, CSc.

*děkan*



doc. Ing. Ivan Zelinka, Ph.D.

*ředitel ústavu*

## **ABSTRAKT**

Cílem bakalářské práce na téma Obrazová analýza radiálně symetrických vzorků bylo nalézt metody obrazové analýzy radiálně symetrických obrázků, které vznikly z elektroeologických vzorků. Tyto poznatky pak použít pro vytvoření programů v IDE Delphi a v Javě, které analyzují radiálně symetrické vzorky ve formě bitmapových obrázků. V praktické části pak jsou zhodnoceny výsledky analýz jednotlivých vzorků.

Klíčová slova: Obrazová analýza, radiálně symetrický vzorek, Delphi, Java

## **ABSTRACT**

The purpose of this bachelor thesis on a given subject Image analysis of radial symmetrical samples was to find methods of image analysis of radial symmetrical images, which are developed from electrorheological samples. After, this knowleges use in designing the applications on IDE Delphi and Java which analysis the radial symmetrical samples represented by bitmap images. The result of analysis of each samples are in the practise part of this bachelor thesis.

Keywords: Image analysis, radial symmetrical sample, Delphi, Java

*Rád bych na tomto místě poděkoval všem, kteří mi pomohli nebo umožnili vytvořit tuto bakalářskou práci. Z odborného hlediska je to na prvním místě vedoucí této práce doc. RNDr. Petr Ponižil, Ph.D., který se po celou dobu psaní textu i souvisejících programů o danou problematiku zajímal a mou práci kontroloval. Dík také patří doc. Dr. Ing. Vladimíru Pavlínkovi, který je vedoucím grantu číslo 202/06/0419 České grantové nadace s názvem „Tvorba elektrorheologických struktur a jejich charakterizace“, jež dal vzniknout i této bakalářské práci. Také děkuji Prof. Takeshi Kitanovi, Ph.D. za přípravu elektrorheologických vzorků.*

Prohlašuji, že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků, je-li to uvolněno na základě licenční smlouvy, budu uveden jako spoluautor.

Ve Zlíně 4. května 2008

.....

Tomáš Dřímál

# Obsah

Úvod	8
<b>I Teoretická část</b>	<b>9</b>
<b>1 Teoretický rozbor obrazové analýzy radiálně symetrických vzorků</b>	<b>10</b>
1.1 Vznik kruhově symetrických vzorků . . . . .	10
1.1.1 Úvod do elektroeologie . . . . .	10
1.1.2 Experiment . . . . .	12
1.2 Obrazová analýza radiálně symetrických vzorků . . . . .	12
1.2.1 Hledání středu . . . . .	13
1.2.2 Transformace . . . . .	15
1.2.3 Výsledky . . . . .	17
<b>2 Použité programovací nástroje</b>	<b>19</b>
2.1 Delphi 7 . . . . .	20
2.2 Java . . . . .	21
2.2.1 Vývojové prostředí pro práci s Javou . . . . .	22
<b>II Praktická část</b>	<b>24</b>
<b>3 Program Analyzér kruhových vzorků</b>	<b>25</b>
3.1 Uživatelský návod pro program Analyzér kruhových vzorků . . . . .	25
3.1.1 Hledání středu . . . . .	25

---

3.2	Analyzér kruhových vzorků z programátorského hlediska . . . . .	29
3.2.1	Otevření obrázku . . . . .	29
3.2.2	Hledání středu . . . . .	30
3.2.3	Rozdělení na sektory a výpočet průměrů . . . . .	32
<b>4</b>	<b>Program Cross-Corel</b>	<b>35</b>
4.1	Uživatelský návod pro ovládání programu Cross-Corel . . . . .	35
4.2	Cross-Corel z programátorského hlediska . . . . .	37
	<b>Závěr</b>	<b>40</b>
	<b>Závěr v angličtině</b>	<b>41</b>
	<b>Seznam použité literatury</b>	<b>42</b>
	<b>Seznam použitých symbolů a zkratek</b>	<b>43</b>
	<b>Seznam obrázků</b>	<b>44</b>
	<b>Seznam příloh</b>	<b>46</b>

## Úvod

Tato práce se zabývá obrazovou analýzou radiálně symetrických struktur. Symetrické struktury vznikly zesíťováním suspenze kapalného polymeru s určitým obsahem skleněných kuliček o velikosti v řádech desítek  $\mu m$  vložených do vnějšího elektrického pole o vysoké intenzitě, řádově v  $kV/mm$ . Pak byly tyto suspenze vloženy do rotačního viskozimetru a byly zaznamenány výsledky při různých smykových rychlostech.

Obrazová analýza se týká těchto naskenovaných vzorků, které mají radiální strukturu. U těchto struktur však nelze najít jednoduše střed rotace, a proto je hledání středu provním úkolem této obrazové analýzy.

Protože jsou jednotlivé vzorky relativně lehce deformovatelné, tak je potřeba obrazovou analýzou zjistit, jestli nejsou deformované, a případně tuto deformaci eliminovat. Proto je pak naskenovaný vzorek rozdělen na sektory a analýza pracuje s každým radiálním sektorem zvlášť.

Výsledkem celé analýzy je závislost průměrných jasů ve vzorku na vzdálenosti od středu.



# I. Teoretická část

# 1 Teoretický rozbor obrazové analýzy radiálně symetrických vzorků

## 1.1 Vznik kruhově symetrických vzorků

V této části je vysvětleno, jak vznikly radiálně symetrické vzorky a jaký bude zvolen postup při obrazové analýze.

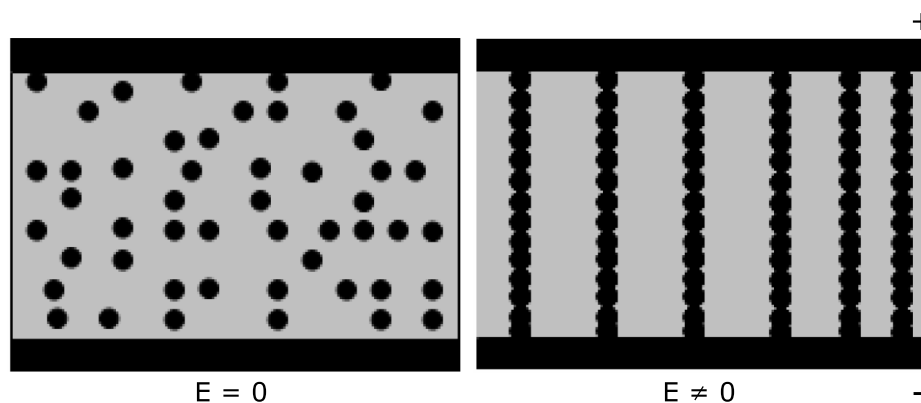
### 1.1.1 Úvod do elektoreologie

Elektoreologické kapaliny jsou systémy, jejichž reologické vlastnosti (viskozita, průtažnost ve smyku) mohou být řízeny vnějším elektrickým polem. Při dosavadním zkoumání elektoreologie suspenzí pevných látek v kapalinách byly pozorovány 2 jevy: kladný a záporný elektoreologický efekt. Kladný znamená, že u elektoreologické kapaliny vložené do vnějšího elektrického pole začne plynule růst viskozita, nebo tato kapalina okamžitě ztuhne. Záporný efekt znamená, že naopak viskozita u kapaliny po vložení do elektrického pole klesá. Z toho vyplývá, že využití elektoreologických kapalin je široké [2].

Pozitivním elektoreologickým efektem se poprvé zabýval před asi padesáti lety W.M.Winslow [1]. Jeho práce dala vzniknout dalším vyčerpávajícím studiím [2, 3, 4], které se zabývaly vysvětlením daného mechanismu, přípravou elektoreologické kapaliny s optimální efektivitou a možným využitím elektoreologického efektu.

Elektoreologické kapaliny jsou obvykle suspenze pevných částic v elektricky nevodivých kapalinách (minerální nebo rostlinný olej). První generace těchto kapalin se sestávala z organických či anorganických látek bez vody, nebo s určitým množstvím vody či jiného spouštěče. Poslední generace elektoreologických kapalin je založena na suspenzi elektricky vodivých polymerů a nanočástic různých povah, s ohledem

na jejich možnou polarizovatelnost. Elektroreologický efekt je příčinou meziplošné polarizace rozptýlených částic [3, 4], vyvolané externím elektrickým polem o vysoké intenzitě ( $E$  v řádech kV/mm). Polarizované částice jsou orientovány ve směru elektrického pole a vytváří struktury (řetězce), které zvyšují tuhost původní kapaliny (obrázek 1.1). Po vložení suspenze do elektrického pole je při nízké smykové rych-



Obrázek 1.1: Zobrazení řetězcové struktury elektroreologické suspenze bez vnějšího elektrického pole a s vnějším elektrickým polem

losti (těsně po překročení možné průtažnosti ve smyku) viskozita vysoká. S rostoucí smykovou rychlostí však viskozita klesá. To způsobují smykové síly, které zapříčiňují rozpad vytvořených struktur. Při vysokých smykových rychlostech je pak viskozita nízká.

Rozložení řetězců částic v elektrickém poli je viditelné pod optickým mikroskopem. Při vysvětlování mechanismu reorganizace těchto struktur v elektrickém poli většina studií doposud zdůrazňovala závislost reologie a viskoelastického chování (gradient závislosti viskozity, smykového napětí a viskoelastických modulů)[3]. Jiné se zase zabývaly průtažností ve smyku elektroreologické kapaliny ve vztahu k chemické povaze a fyzickým vlastnostem rozptýlených částic vzhledem k jejich polarizovatelnosti (tvar a velikost částic, stejnosměrná a střídavá elektrická vodivost, permitivita a dielektrické ztráty). Výsledky těchto studií však neposkytují důkaz o změnách rozmístění částic po vložení do vnějšího elektrického pole [4].

Naproti tomu řízené optické zobrazení elektoreologických suspenzí v rotačním viskozimetru, prezentované v [5, 6], ukazuje, že se částice přeskupují do lamelárních nebo prstencovitých struktur závislých na vnějším elektrickém poli, které mají minimální energetické ztráty. Tyto experimenty ukazují cestu k objasnění možností řízení tokových mechanismů elektoreologických struktur.

### 1.1.2 Experiment

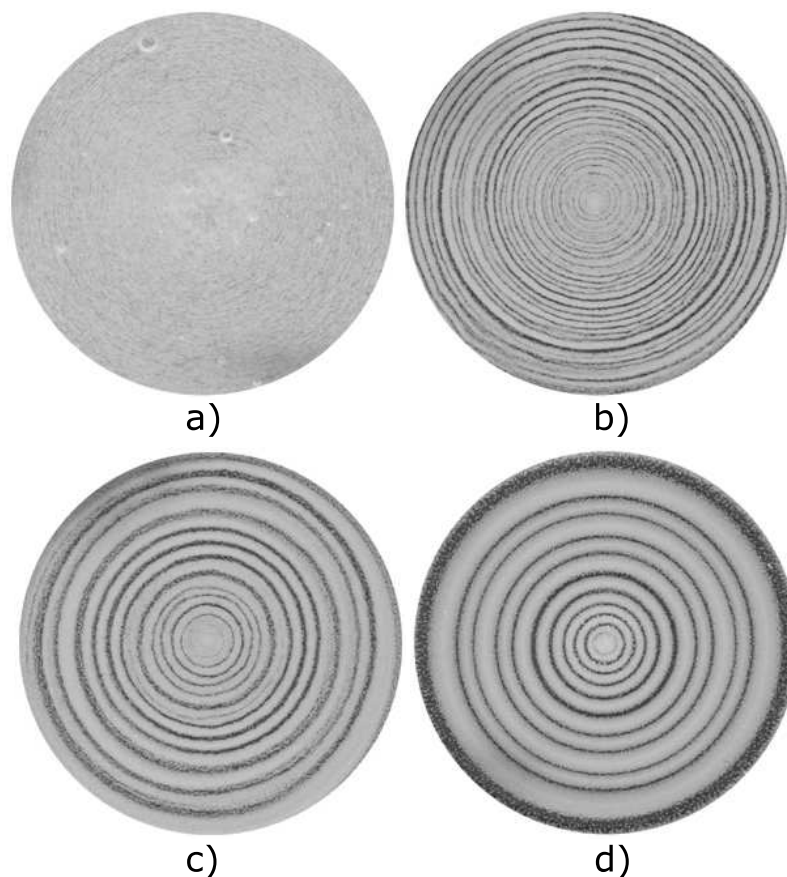
Při experimentu byl použit rotační viskozimetr typu kužel-deska značky Bohlin Gemini (Malvern Instruments, UK). Princip rotačního viskozimetru spočívá v určení viskozity ze síly potřebné pro otáčení předmětu ve zkoumané kapalině.

Jako elektoreologická suspenze byly použity 20% a 40% směsi skleněných kuliček o velikosti  $40 \mu m$  a tekuté silikonové pryže. Smyková rychlost byla pro každý experiment jiná, elektrické pole, ve kterém probíhal experiment, však bylo konstantní –  $1 kV/mm$ . Vygenerované obrazce byly zafixovány díky zesíťování silikonového pryžového materiálu. Tyto pak byly skenovány na profesionálním skeneru v odstínech šedi s rozlišením zhruba  $700 DPI$ . Výstupem pak byly obrázky o rozměrech asi  $1000 \times 1000$  pixelů (obrázek 1.2 a), b), c) a d)).

Kruhové struktury vytvořené pohybem elektoreologických kapalin v elektrickém poli evidentně závisí na smykové rychlosti, což je patrné z obrázku 1.2.

## 1.2 Obrazová analýza radiálně symetrických vzorků

Cílem obrazové analýzy radiálně symetrických vzorků je získat středovou závislost intenzit (jasů) v obrázku. Jedná se o závislost průměrného jasu na vzdálenosti od středu a její korelaci s vlastnostmi daného elektoreologického vzorku. Skládá se z několika částí. Prvním úkolem je najít střed symetrie dané struktury, aby mohly být spočítány průměrné hodnoty jasů bodů stejně vzdálených od středu. Pak je potřeba rozdělit celý obrazec na 4 nebo 8 sektorů a pro tyto spočítat průměrné



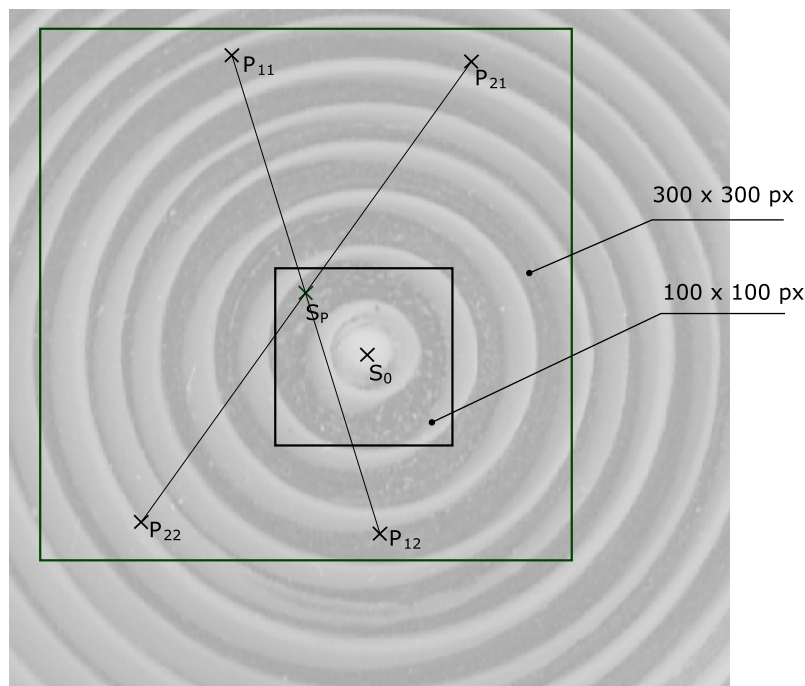
Obrázek 1.2: Elektroeologické tokové obrazce. 40% koncentrace skleněných kuliček (bílá barva) v elektrickém poli o velikosti  $1kV/mm$  při různých smykových rychlostech: a)  $0,1s^{-1}$ , b)  $1,0s^{-1}$ , c)  $3,0s^{-1}$  a d)  $30,0s^{-1}$ .

hodnoty jasů pro všechny body, které jsou stejně vzdáleny od skutečného středu. Tedy určit jas v závislosti na vzdálenosti od středu pro každý sektor zvlášť.

### 1.2.1 Hledání středu

Nejprve je potřeba rozpoznat střed symetrie. První přibližný odhad může být udělán manuálně, nebo automaticky tak, že bude zvolen bod, který je ve středu obrázku.

Na obrázku 1.3 je středová oblast vzorku spolu s popisky důležitých bodů.  $S_0$  je bod, který je uvažován jako přibližný střed. Okolo tohoto bodu je stanoveno okolí



Obrázek 1.3: Středová oblast elektreologického vzorku

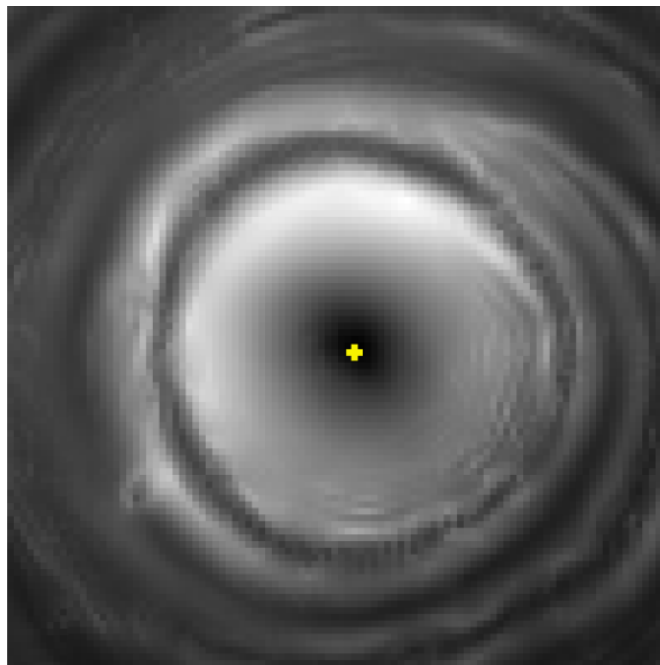
$100 \times 100 \text{ px}$ . Pro každý bod  $S_P$  (potenciální střed) v tomto okolí je určena oblast  $300 \times 300 \text{ px}$ , ve které jsou porovnávány intenzity dvojic bodů středově souměrných podle bodu  $S_P$ . Na obrázku 1.3 tedy dvojice  $P_{11}$  a  $P_{12}$ , případně  $P_{21}$  a  $P_{22}$ .

Uvažujeme, že  $p(x, y)$  je jas pixelu v  $x$ -tém sloupci a na  $y$ -tém řádku v obrázku. Prvním přibližným odhadem se určí bod  $S_0$  se souřadnicemi  $(x_{c0}, y_{c0})$ . Pro určité okolí tohoto bodu ( $100 \times 100 \text{ px}$ ) je pak počítána funkce symetrie (porovnávání dvojic bodů z předchozího odstavce) jako:

$$S_P(x_c, y_c) = \sum_{x,y=-300}^{300} \frac{(p(x_c + x, y_c + y) - p(x_c - x, y_c - y))^2}{\sqrt{x^2 + y^2}}, \quad (1)$$

kde čitatel je druhá mocnina rozdílů jasů dvou bodů stejně vzdálených od středu (tedy například dvojice  $P_{11}$  a  $P_{12}$  z obrázku 1.3) a jmenovatel je vzdálenost jednoho porovnávaného bodu (např.  $P_{11}$ ) od potenciálního středu  $S_P$ . Pro okolí bodu  $S_0$  tedy vznikne matice hodnot podle toho, nakolik jsou si symetricky souměrné body podobné. Čím je hodnota vyšší, tím jsou si podobnější, tím je pravděpodobnější, že je daný bod skutečným středem. Obrázek 1.4 graficky znázorňuje výsledek funkce

$S(x_c, y_c)$ . Světlejší pixely indikují vyšší hodnotu funkce  $S(x_c, y_c)$ , kříž blízko středu obrázku 1.4 určuje skutečnou pozici středu symetrie  $S$  – absolutní maximum funkce  $S(x_c, y_c)$  na pozici  $(x_{cm}, y_{cm})$ .



Obrázek 1.4: Grafická reprezentace funkce  $S(x_c, y_c)$ , kříž uprostřed označuje polohu skutečného středu

### 1.2.2 Transformace

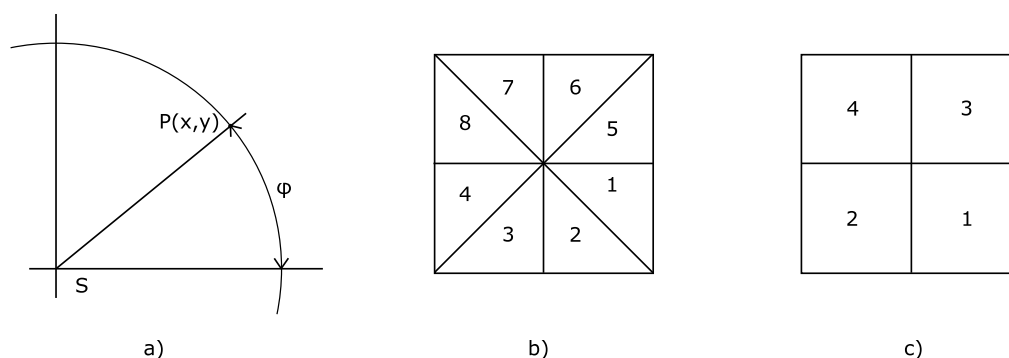
Vyskytuje se zde však jeden problém. Vzorky jsou relativně měkké a snadno deformovatelné. Obrázky proto mohou být nepatrně protáhlé v jednom či více směrech.

Proto je potřeba vzorek rozdělit na sektory a provést výpočet průměrných intenzit bodů stejně vzdálených od skutečného středu  $S$  pro každý sektor zvlášť. To je realizováno tak, že je celý vzorek procházen bod po bodu a pro každý z těchto bodů je spočítána vzdálenost  $l$  v  $px$  od skutečného středu  $S$  pomocí Pythagorovy věty a zaokrouhlena na celé číslo. Číslo sektoru, do kterého patří pixel na souřadnici

$p(x, y)$ , se počítá podle vzorce:

$$NS = \frac{\varphi}{\pi} \cdot \frac{N}{2}, \quad (2)$$

kde  $NS$  je číslo sektoru (je patrné na obrázku 1.5 b) a c)),  $\varphi$  je úhel podle schématu na obrázku 1.5 a) a  $N$  je počet sektorů, na který je vzorek dělen – 4 nebo 8. Je-li tedy známo, jaká je vzdálenost bodu  $p(x, y)$  od skutečného středu  $S$  a do jakého sektoru patří, tak je možné spočítat, jaké jsou průměrné intenzity bodů stejně vzdálených od středu  $S$  pro každý sektor zvlášť.



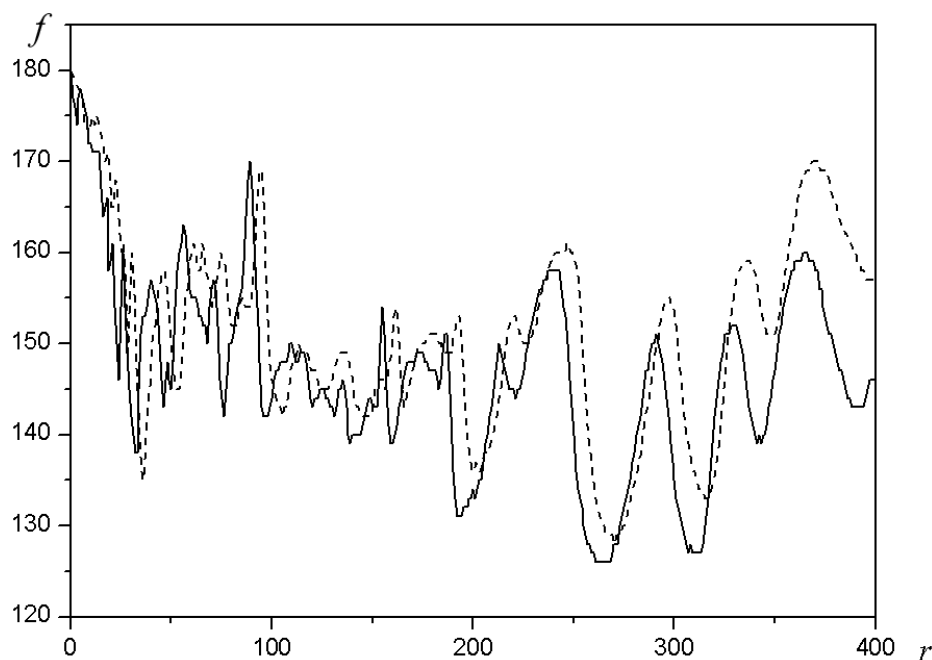
Obrázek 1.5: Dělení vzorku na sektory. a) Schéma určení čísla sektoru podle vzorce (2), b) čísla sektorů při rozdělení vzorku na 8 částí a c) čísla sektorů při rozdělení vzorku na 4 části

V okamžiku, kdy je vzorek rozdělen na sektory, tak uvažujeme jas jako funkci  $f_i(r)$  (index  $i$  určuje jeden ze sektorů) ve vzdálenosti  $r$  od skutečného středu. Na obrázku 1.6 je srovnání této funkce pro dva různé sektory v naskenovaném vzorku. Z tohoto obrázku je patrné, že je potřeba funkci  $f_j$  posunout a roztáhnout, aby byla eliminována deformace. Nástrojem pro zjištění podobnosti dvou funkcí je vzájemná korelační funkce [7], která je pro dvě reálné funkce  $f_i$  a  $f_j$  definována jako:

$$(f_i * f_j)(x) = \int f_i(r) f_j(r + x) dr \quad (3)$$

Uvažujme například dvě reálné funkce  $f_i$  a  $f_j$  které se liší pouze posunem v ose  $x$ . Je potřeba spočítat, jak musíme posunout funkci  $f_j$ , aby byla identická s funkcí  $f_i$ . Vzorec (3) posune funkci  $f_j$  ve směru osy  $x$  tak, že spočítá integrál pro každé možné





Obrázek 1.6: Srovnání dvou průběhů funkce  $f_i(r)$ . Jas  $f$  a vzdálenost od středu  $r$  jsou vyneseny na osách.

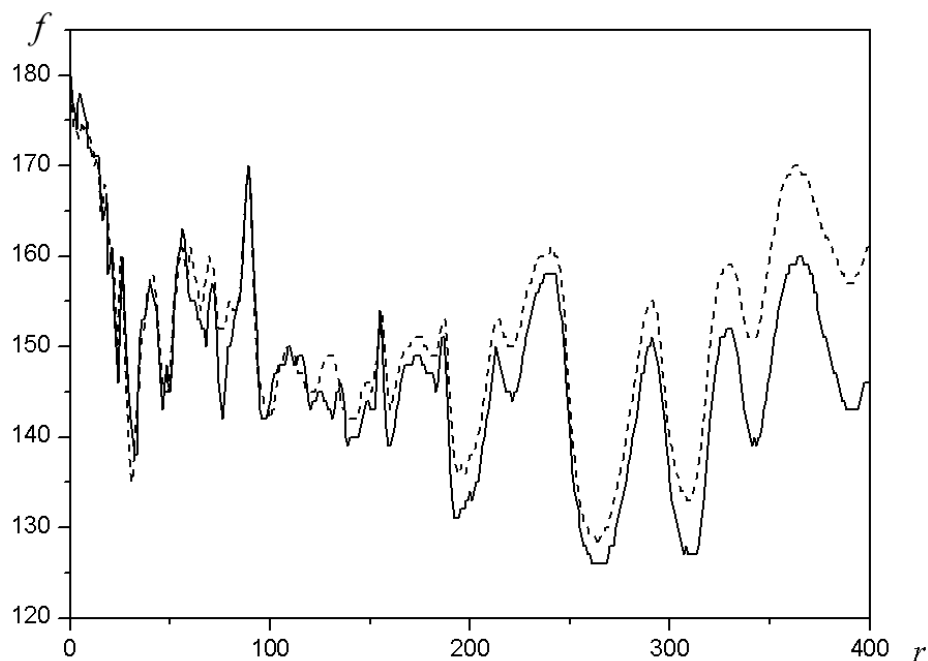
posunutí funkce  $f_j$ . Když jsou tyto dvě funkce shodné, je hodnota funkce  $(f_i * f_j)$  maximální. To je způsobeno tím, že překrytí vrcholů (lokálních maxim obou funkcí) způsobuje vyšší hodnotu integrálu. Stejně tak i zarovnání záporných oblastí (lokální minima) způsobuje vyšší hodnotu integrálu, protože součin dvou záporných čísel je kladný. Pro náš případ byla definována vzájemně korelační funkce obecněji takto:

$$(f_i * f_j)(h, x) = \int f_i(r) f_j(hr + x) dr, \quad (4)$$

kde  $h$  je měřítko zvětšení a  $x$  je posunutí ve směru osy  $x$ . Jestliže je hodnota funkce  $(f_i * f_j)$  maximální, takto znamená, že byla nalezena nejlepší kombinace  $h$  a  $x$ .

### 1.2.3 Výsledky

Když je nalezena nejvhodnější kombinace  $h$  a  $x$  pro všechny průběhy (vyjma prvního, který je referenční), tak jsou tyto průběhy přepočítány (interpolovány) podle těchto hodnot. Z těchto upravených průběhů (a prvního referenčního) je spočítána výsledná

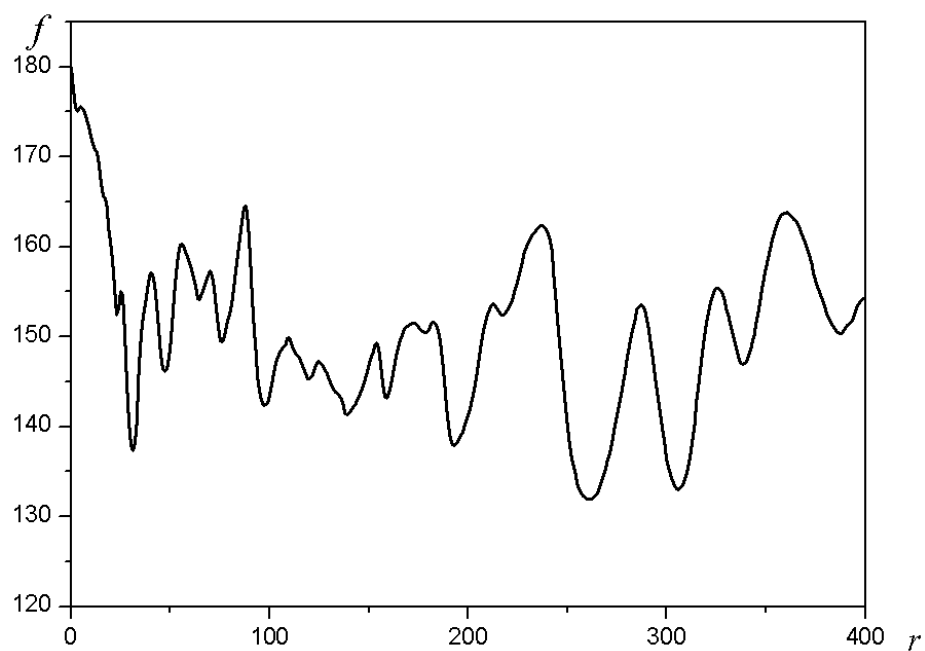


Obrázek 1.7: Původní funkce  $f_1(r)$  a transformovaná funkce  $f_2(hr + x)$ . Při hodnotách  $h = 0.972$  a  $x = -5.07$  je mezikorelační funkce maximální.

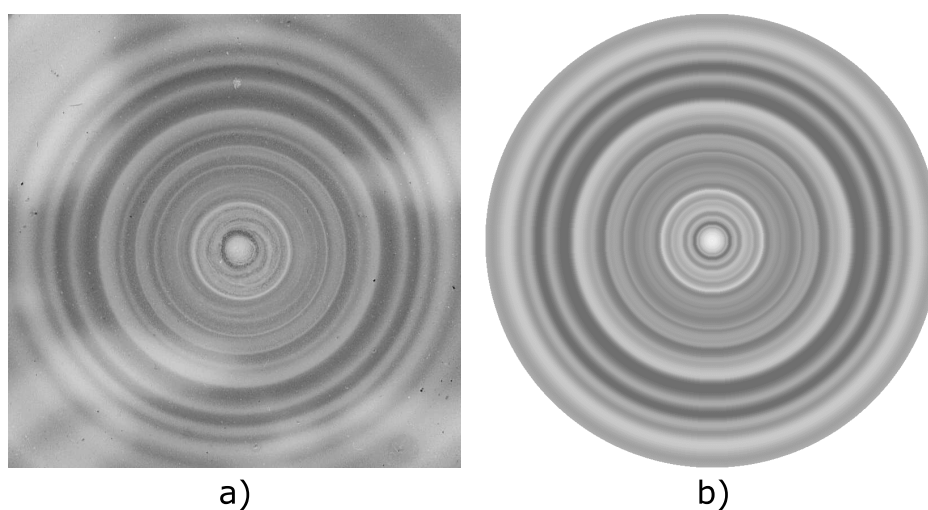
funkce  $f(r)$  (obrázek 1.8), která je jejich průměrem. Je to výsledná závislost průměrného jasů na vzdálenosti od středu vzorku.

Obrázek 1.9 zobrazuje srovnání centrálních částí originálního vzorku a obrázku vytvořeného na základě středové závislosti jasů, funkce  $f(r)$  z obrázku 1.8.

Nyní je radiálně symetrický vzorek charakterizován funkcí  $f(r)$ , zobrazenou na obrázku 1.8. Je tedy připravena závislost rozmístění lokálních maxim funkce  $f(r)$  na parametrech daného radiálního vzorku pro další zkoumání.



Obrázek 1.8: Průměrná  $f(r)$  funkce.



Obrázek 1.9: Centrální část původního obrázku (a) a odpovídající výsledek obrazové optimalizace (b)

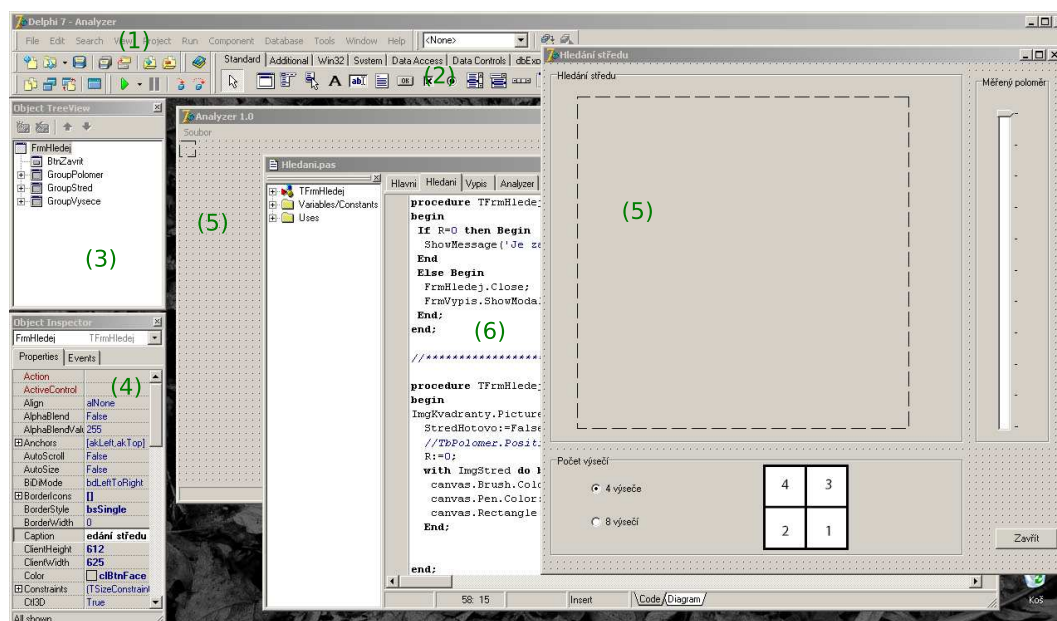
## 2 Použité programovací nástroje

V této kapitole budou představeny programovací nástroje, které byly použity při tvorbě programů Analyzer a Cross-correl. Jsou jimi IDE Delphi a Java.

### 2.1 Delphi 7

Delphi je IDE (Integrated debugging Enviroment) založené na jazyce Object Pascal. Lze v něm tvořit konzolové aplikace, jako například v prostředí Turbo Pascal, jeho hlavní výhodou je však velice jednoduché prostředí pro tvorbu GUI (Graphic User Interface). Což znamená, že vývojář jednoduše vybere z palety komponent komponenty, které potřebuje a myší je umístí na plochu kontejnerové komponenty (nejčastěji asi do okna – komponenta `TFrame`). Pak jen naprogramuje obsluhu událostí těchto komponent tak, jak potřebuje. Prostředí je plně objektové, takže programátor vlastně pracuje objektově, aniž by o tom věděl. To je důvodem, proč je Delphi tak rozšířeným nástrojem mezi programátory – začátečníky. Pokročilejší narazí na určité problémy, které jim pomohou vyřešit jiné programovací jazyky, jako například Java, o které se zmiňuji v části 2.2.

Jak je patrné na obrázku 2.1, tak se prostředí Delphi skládá z více oken. Hlavní okno (1) se skládá ze základní nabídky, hlavního menu a hlavně z palety komponent (2), které jsou programátorovi k dispozici. V okně Object TreeView (3) je možné podívat se na projekt a komponenty v něm umístěné ve stromové struktuře. V době návrhu aplikace je možné měnit některé vlastnosti komponent. K tomu slouží okno Object Inspector (4). Pro aktuální komponentu (tzn. která je zrovna vybraná – konkrétní instance) jsou zde všechny vlastnosti, které lze měnit v době návrhu (což jsou ty, které jsou v dané třídě v sekci `published`). Části (5) jsou instancemi komponenty `TFrame`, která je vlastně oknem ve Windows. Tyto dvě instance obsahují



Obrázek 2.1: Obrazovka IDE Delphi 7

instance dalších komponent, které ve výsledku vytvářejí funkční aplikaci. V okně (6) je zdrojový kód jednotlivých obsluh událostí.

V Delphi je možné vytvářet aplikace pro Windows, případně se dá aplikace přeložit i pro Linux. K tomu je však potřeba jiný překladač – Kylix, který je rovněž od firmy Borland, stejně jako celé IDE Delphi. Více o Delphi a programování konkrétní aplikace je v části 3.2.1

## 2.2 Java

Programovací jazyk Java je vyvíjen společností Sun Microsystems od roku 1991. Původně byl vyvíjen pro vestavěná zařízení<sup>1</sup>, v roce 1993 si však tvůrci uvědomili rostoucí vliv WWW a rozšířili tento projekt na tvorbu aplikací pro web. V roce 1995 byl představen. Od té doby jeho popularita roste a v současné době jej lze nalézt jak ve webových aplikacích, tak i v mobilních telefonech a dalších zařízeních.

<sup>1</sup>jedná se o běžná elektronická zařízení, jak například pračky, mikrovlnné trouby a podobné

Programy napsané v Javě nejsou závislé na platformě, na které budou spuštěny. Zdrojový kód, uložený v souboru s příponou `.java`, je překladačem (SDK – Standard Development Kit) přeložen do takzvaného byte-kódu. Ten je uložen v souboru s příponou `.class`. Tento byte-kód může být spuštěn na libovolném počítači na libovolné platformě. Jedinou podmínkou je, že na daném stroji je nainstalována JVM (Java Virtual Machine). Ta zajistí „dopřeložení“ zdrojového kódu pro daný operační systém a jeho spuštění. [8]

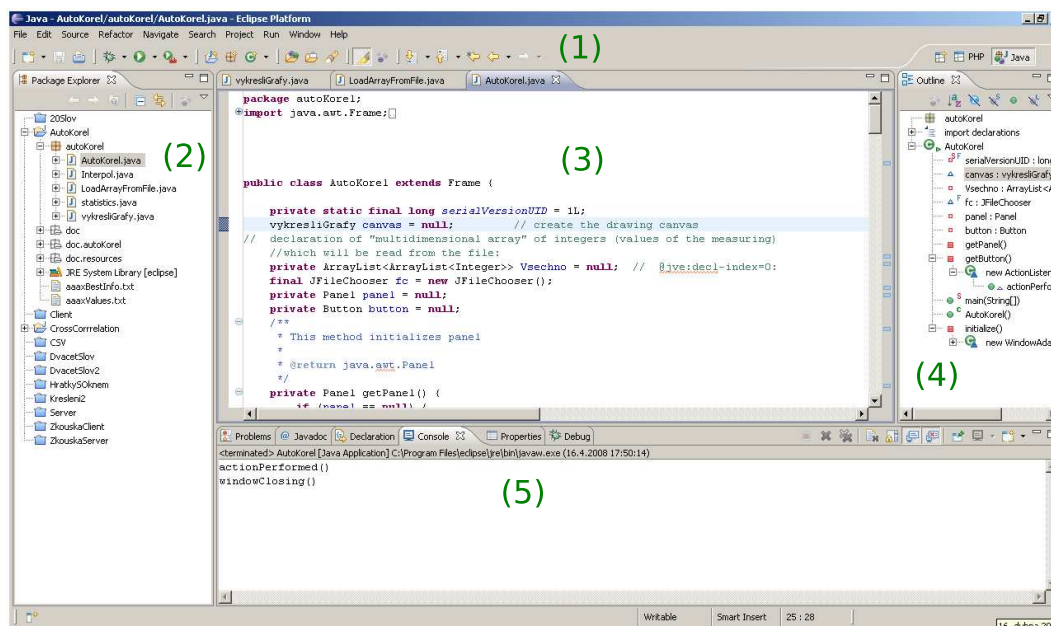
Java je striktně objektově orientovaný jazyk. Jinak, než objektově, v něm programovat nelze. Dokonce i datové typy jsou třídami, kromě typů základních (`integer`, `float`, `char`, ...). I tyto primitivní typy mají svoje objektové ekvivalenty (`Integer`, `Float`, `Char`, ...), jejichž názvy se liší jen ve velikosti prvního písmene. Například pole už je ale objektem. Tedy v případě, že se pole jmenuje `pole1`, tak velikost (délku) nezjistíme, že na toto pole zavoláme funkci `Length` v tomto tvaru: `Delka = Length(pole1)`, ale tak, že zavoláme metodu dané instance `pole1`: `Delka = pole1.Length`. Tato vlastnost má spoustu výhod, například jednoduchou znovupoužitelnost kódu, ale odrazuje od tohoto jazyka začátečníky a dává vytvářet fámám, že je nejsložitějším jazykem.

### 2.2.1 Vývojové prostředí pro práci s Javou

Programátor v Javě si vystačí jen s SDK. Zdrojový kód napíše v libovolném programu pro tvorbu textových souborů (Poznámkový blok, PSPad, ...) a přeloží a spustí pomocí příkazové řádky. Toto vybavení však neposkytuje kontrolu syntaktických chyb a práce s ním je nepraktická. Proto existují IDE pro práci s Javou, jako například Eclipse<sup>2</sup>. Toto vývojové prostředí jsem použil pro práci s Javou.

Jak je vidět na obrázku 2.2, tak se hlavní okno IDE Eclipse skládá taky z dílčích oken, podobně, jako tomu je u Delphi 7 (obrázek 2.1). Hlavní okno (1) obsahuje hlavní nabídku a tlačítka typu otevření projektu, jeho uložení a další. Okno Package

<sup>2</sup>Open source projekt vyvíjen společností Eclipse Foundation



Obrázek 2.2: Okno IDE Eclipse Europa

Explorer (2) obsahuje stromovou strukturu v aktuálním workspace, což je složka na disku, ve které se nachází projekty, se kterými lze pracovat. V oblasti (3) je editor, ve kterém jsou v jednotlivých záložkách otevřeny zdrojové kódy jednotlivých tříd. V části Outline (4) je stromová struktura aktuální třídy. Jsou zde zobrazeny všechny objekty, které se v ní nacházejí. V oblasti (5) je konzole, ve které běží Java aplikace vytvořené bez GUI (Graphical User Interface), tedy takové, které s uživatelem komunikují jen přes příkazovou řádku. Zde se také nachází okno s varováními a s chybami, nebo na záložce Properties ekvivalent k Object Inspectoru z Delphi 7 (obrázek 2.1, část (4)).

## II. Praktická část



## 3 Program Analyzér kruhových vzorků

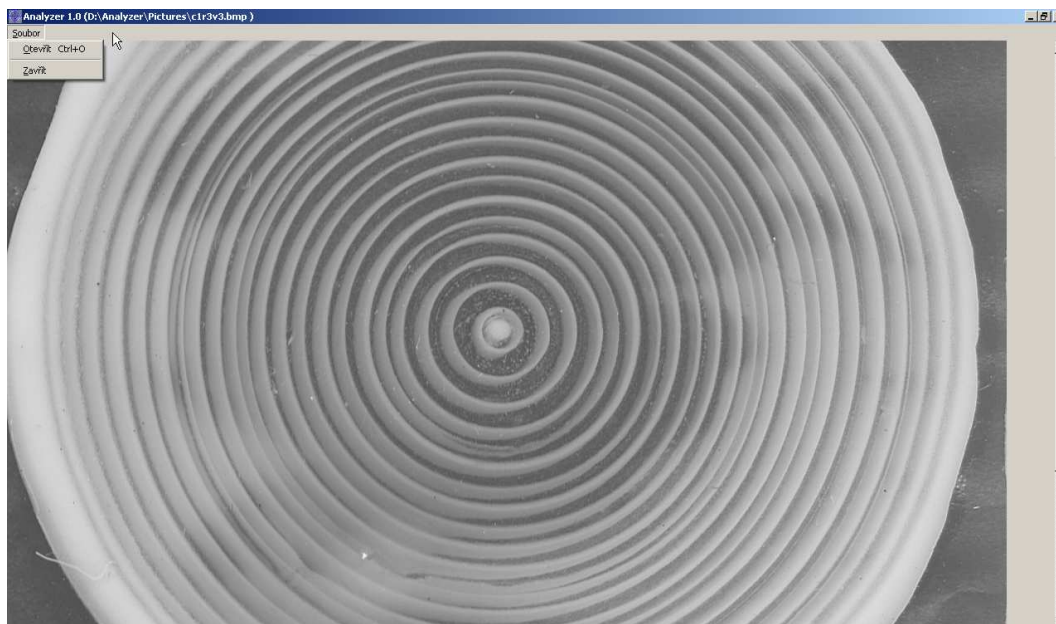
Program analyzér kruhových vzorků je vytvořen v IDE Delphi, o kterém je zmínka v části 2.1. V první části je tento program popsán z uživatelského hlediska. V další části pak je popsáno, jak byl naprogramován.

### 3.1 Uživatelský návod pro program Analyzér kruhových vzorků

Po spuštění se zobrazí prázdné okno, ve kterém je pouze nabídka Soubor v hlavním menu. Toto menu obsahuje dvě volby, z nichž jedna je pro ukončení celé aplikace a druhá je pro otevření obrázku – kruhového vzorku ve formátu BMP. Po kliknutí na volbu Otevřít se otevře dialogové okno pro výběr obrázku. Když uživatel vybere potřebný vzorek a volbu potvrdí tlačítkem Otevřít, tak se otevře obrázek – zobrazí se v okně Analyzáru. Tento stav může vypadat, jako například na obrázku 3.1.

#### 3.1.1 Hledání středu

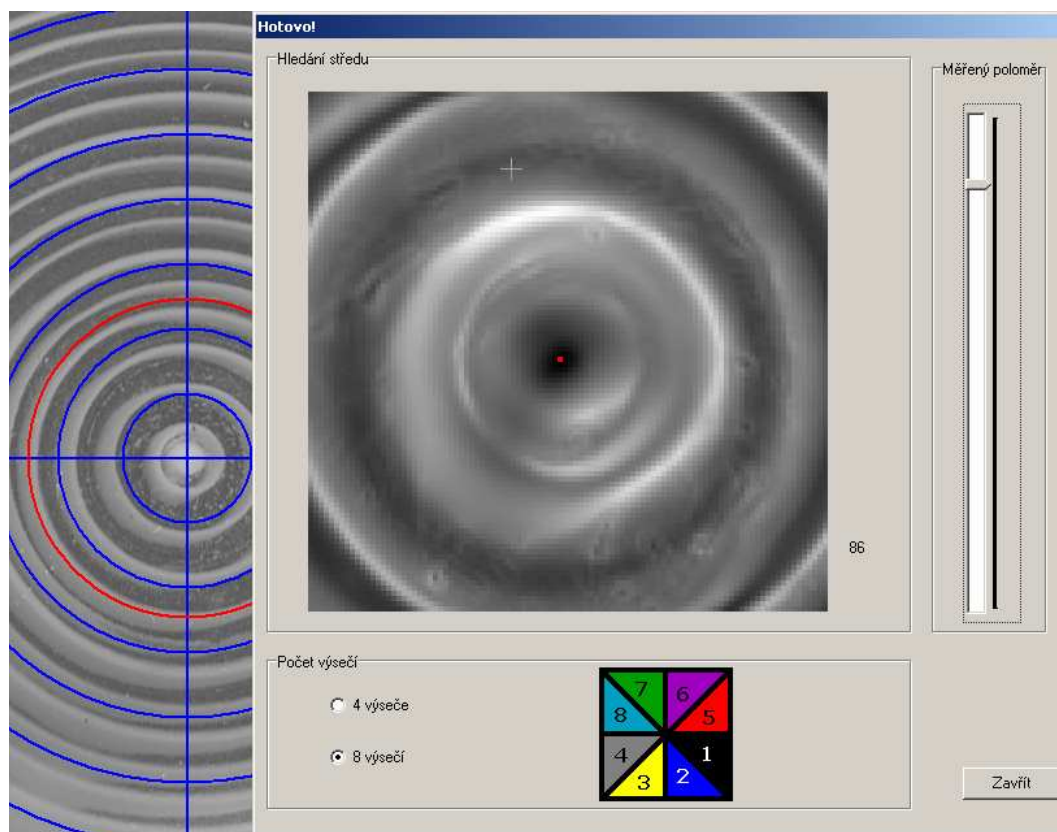
Jak bylo uvedeno v části 1.2.1, tak je nejprve potřeba určit bod, který je ve středu vzorku. V programu Analyzér se první přibližný odhad provádí manuálně, a to tak, že je potřeba kliknout myší do místa, kde se asi nachází střed. Po tomto kliku se automaticky spustí funkce, která pro okolí zadaného bodu zjišťuje, který z bodů v tomto okolí by nejvíce mohl být středem celého obrázku (část 1.2.1). Tato funkce pracuje poměrně dlouho (na PC s procesorem 1,8 GHz a RAM pamětí 1 GB přibližně 1/2 min). Proto se na začátku hledací funkce zobrazí okno, ve kterém pak budou zobrazeny výsledky, v jehož záhlaví se zobrazuje, kolik procent z výpočtu je už hotových.



Obrázek 3.1: Okno programu Analyzér kruhových vzorků s otevřeným kruhovým vzorkem

Když je výpočet dokončen, tak se zobrazí v okně Hledání středu (Obrázek 3.2) mapa bodů, jejichž intenzita určuje, nakolik by mohly být středem. Nejtmaší bod v tomto poli je tedy skutečným středem. Pro větší názornost je tento nejtmaší pixel zvýrazněn červeně, aby jej nebylo potřeba hledat mezi 10 000 body. Klikem na tento červený bod se přesune modrý kříž z pozice určené manuálně na pozici skutečného středu.

V tomto okně se rovněž nachází možnost nastavení měřeného poloměru a taky počet sektorů, na které bude vzorek rozdělen. Na obrázku 3.2 je v levé části středová oblast analyzovaného vzorku. Modrý kříž s modrými soudředy kružnicemi určuje polohu skutečného vypočítaného (a uživatelem upřesněného) středu, červená kružnice určuje měřený poloměr. Velikost této kružnice lze měnit v pravé části okna Hledání středu pomocí vertikálního posuvníku. Počet sektorů, na které bude obrázek rozdělen, může být buď 4 nebo 8. Jednu z těchto dvou možností lze zvolit v dolní části tohoto okna. Vedle těchto přepínacích tlačítek se zobrazuje schematický obrázek symbolizující, jak bude vzorek rozdělen. Sektory jsou zde očíslovány tak, jak budou



Obrázek 3.2: Okno s výsledky hledání středu a možnostmi nastavení

očíslovány naměřené hodnoty.

Po kliknutí na tlačítko Zavřít je zobrazeno okno výpisu, které se skládá ze dvou záložek, jak je patrné na obrázku 3.3. Na záložce s numerickými hodnotami je možné si naměřené hodnoty prohlédnout a případně uložit do textového souboru. K dispozici je také export dat do Microsoft Excelu prostřednictvím protokolu OLE. Export do textového souboru bude potřeba využít pro přenos těchto naměřených dat do programu Cross-Corel, o kterém je v části 4.1.

Na záložce Grafy (obrázek 3.3 b) ) je grafické znázornění naměřených hodnot. Jako výchozí jsou zobrazeny všechny závislosti jasů na vzdálenosti od středu. To proto, že jsou zaškrtnuté všechny zaškrtačací políčka kolem obrázku, znázorňujícího rozdělení vzorku na sektory, stejného jako na obrázku 3.2 v dolní části. Při vyškrtnutí libovolného (i více) z těchto políček se odpovídající průběh nevykreslí. Pak je ještě

Výpis

Výpis Grafy

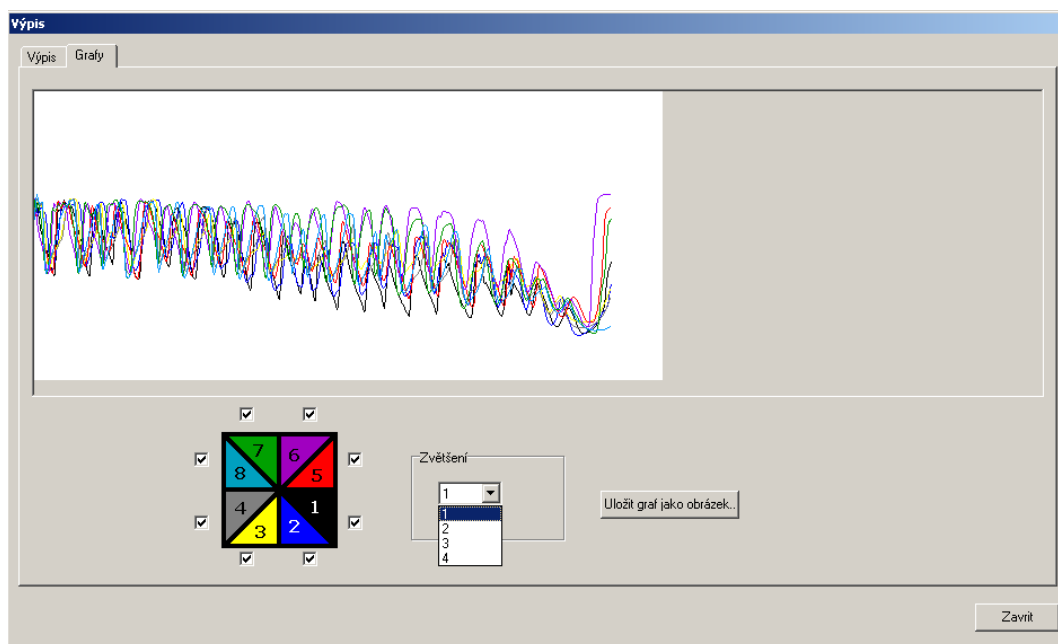
	1. vysec	2. vysec	3. vysec	4. vysec	5. vysec	6. vysec	7. vysec	8. vysec
9	183	179	177	178	189	190	188	182
10	183	178	173	173	187	188	187	182
11	181	175	166	170	185	186	186	179
12	179	167	157	165	184	183	182	177
13	177	150	153	162	181	179	177	172
14	170	132	150	158	179	172	169	166
15	157	128	148	153	172	162	160	161
16	144	131	146	147	162	152	156	155
17	131	129	147	141	150	146	165	142
18	129	116	149	135	147	150	162	147
19	133	111	144	129	148	148	144	136
20	143	111	139	140	150	145	135	138
21	145	116	141	164	148	144	146	168
22	147	130	158	171	148	146	164	172
23	151	148	168	170	149	150	169	169
24	154	160	171	166	151	154	169	165
25	158	165	169	160	153	154	169	158
26	159	166	163	155	154	155	165	154
27	161	165	159	149	156	159	161	152
28	163	160	153	144	158	161	156	147
29	162	147	147	139	159	156	142	141

Export dat do MS Excelu

Export do textového souboru

Zavřít

a)



b)

Obrázek 3.3: Okno s naměřenými hodnotami a) numerickými a b) grafickými

k dispozici možnost zvětšení vykreslených průběhů. Jedná se však jen o zvětšení ve směru osy  $y$ , tedy změna velikosti amplitudy. Po kliknutí na tlačítko Uložit graf jako obrázek.. se zobrazí ukládací dialogové okno, kterým může uživatel zadat umístění pro uložení grafu jako obrázku ve formátu BMP.

## 3.2 Analyzátor kruhových vzorků z programátorského hlediska

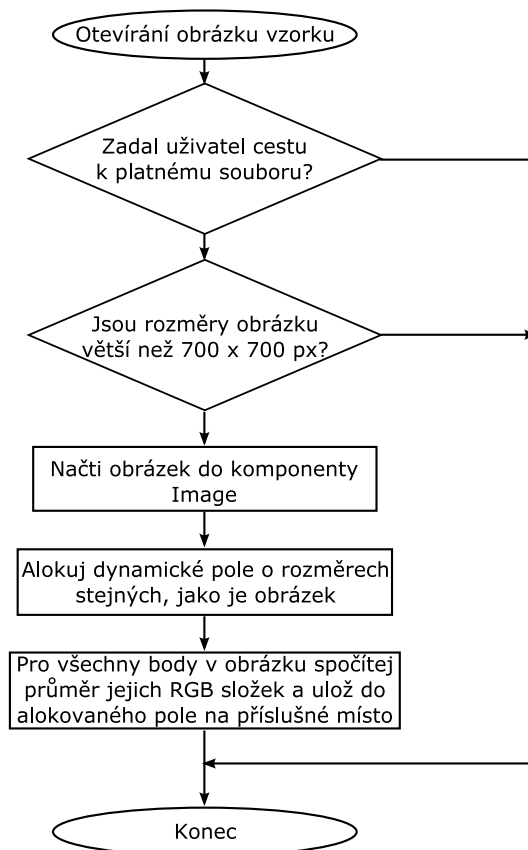
Algoritmus programu Analyzátor lze rozdělit do několika dílčích částí, které probíhají v určitém smyslu nezávisle na sobě. Jejich pořadí je ale nezaměnitelné. Jsou to otevření obrázku, hledání středu a jako poslední je rozdělení obrázku na sektory a výpis hodnot.

### 3.2.1 Otevření obrázku

Hlavní okno Analyzátoru se skládá z několika komponent, které nejsou hned po spuštění vidět. Jsou to hlavní menu (instance třídy `TMainMenu`), ve kterém je právě hlavní nabídka, o které se píše v části 3.1. Pak je to komponenta `Image` (třída `TImage`), ve které bude po otevření zobrazen obrázek vzorku, a dialogové okno pro otevření obrázku vzorku, které je instancí třídy `TOpenPictureDialog`.

Kliknutím na volbu Otevřít v hlavní nabídce se otevře dialog pro výběr souboru pro otevření. To zajišťuje metoda `Execute` instance dialogového okna. Tato metoda otevře dialog, a pokud uživatel vybral existující soubor, tak vrátí hodnotu `True`. Pokud uživatel vybral soubor ve formátu BMP, jehož rozměry jsou větší než  $700 \times 700$  px, tak se obrázek začne otevírat. Jednak se obrázek zobrazí v komponentě `Image`, ale hlavně je načten do dvourozměrného pole, které má stejné rozměry, jako daný obrázek. Mělo by se jednat o obrázek v odstínech šedi. Tedy o matici bodů, z nichž každý má hodnotu od 0 do 255. V komponentě `TImage` se však pracuje

s 24 bitovou grafikou. Aby byla práce s jednotlivými hodnotami rychlejší, tak jsou tyto hodnoty zprůměrovány a uloženy právě do tohoto pole, ve kterém jsou data připravena pro hledání středu a následnou analýzu. Vývojový diagram otevírání obrázku vzorku je na obrázku 3.4.



Obrázek 3.4: Vývojový diagram otevírání obrázku vzorku v programu Analyzér

### 3.2.2 Hledání středu

Program Analyzér postupuje při hledání středu podle postupu popsaného v části 1.2.1. To znamená, že v okamžiku, kdy je otevřený obrázek, program čeká na to, až uživatel zadá polohu přibližného středu. Z programátorského hlediska to znamená, že

v případě, že uživatel klikne do plochy obrázku v instanci komponenty `TImage`, tak se spustí obsluha události `OnMouseDown`. Tato obsluha zajistí vykreslení modrého kříže do plochy obrázku a má k dispozici souřadnice bodu, nad kterým uživatel kliknul. Od tohoto okamžiku je viditelný obrázek vzorku jenom na orientační zobrazování a program pracuje jen s polem, do kterého uložil upravené hodnoty při otevírání obrázku v části 3.2.1.

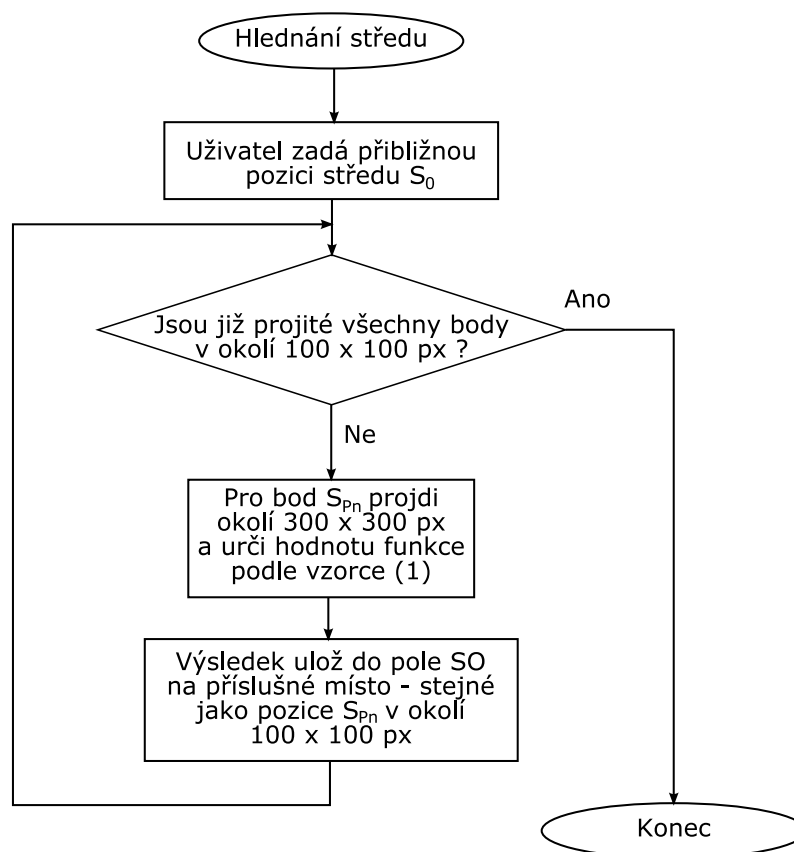
Samotný algoritmus hledání středu se skládá ze čtyř do sebe vložených `For` cyklů. To proto, že pro procházení dvourozměrného pole je zapotřebí dvou těchto cyklů. A jak je uvedeno v části 1.2.1 na obrázku 1.3, tak je potřeba procházet vnitřní oblast  $100 \times 100 \text{ px}$  (dva cykly `For`) a pro každý pixel z této oblasti ještě oblast  $300 \times 300 \text{ px}$ . To je hlavní důvod toho, že tento výpočet trvá dlouho.

Po kliknutí do plochy je známo, kde uživatel odhaduje polohu středu – bod  $S_0$  o souřadnicích  $(x_0, y_0)$ . Vnější dvojice cyklů `For` tedy probíhá od  $x_0 - 50$  do  $x_0 + 50$  a druhý od  $y_0 - 50$  do  $y_0 + 50$ , což je právě okolí  $100 \times 100 \text{ px}$  z obrázku 1.3. Uvažujme, že každý z bodů v tomto okolí je  $S_{P_n}$ , kde  $n$  je v rozmezí od 1 do 10 000.

Pro všechny body  $S_{P_n}$  je počítána funkce podle vzorce (1) pro okolí  $300 \times 300 \text{ px}$ . Výsledky této funkce jsou ukládány do pole se jménem `S0`, které pak je převedeno na mapu pixelů pro grafickou interpretaci středové funkce, jako například na obrázku 1.4.

Výsledek hledání středu je uložen v poli `S0`. V okamžiku, kdy jsou tam všechny hodnoty, tak je ukončeno automatické hledání středu (ve vývojovém diagramu na obrázku 3.5 oblast označená jako konec). Výsledek je z pole `S0` je graficky vykreslen do okna hledání středu, což je instance třídy `TFrame`, ve které je i instance třídy `TImage` pojmenovaná `ImgStred`, do které je právě výsledek vykreslen (obrázek 3.2). Oblast v `ImgStred` představuje okolí bodu  $S_0$ , které bylo prohledáváno. Je vyplněna hodnotami z pole `S0`, kde pro větší názornost jedné hodnotě z tohoto pole odpovídá plocha  $4 \times 4 \text{ px}$ . Nejtmavší bod pak není vykreslen černou, ale červenou, aby jej bylo možné snadno najít.

Když uživatel klikne do prostoru komponenty `ImgStred`, tak přemístí modrý kříž



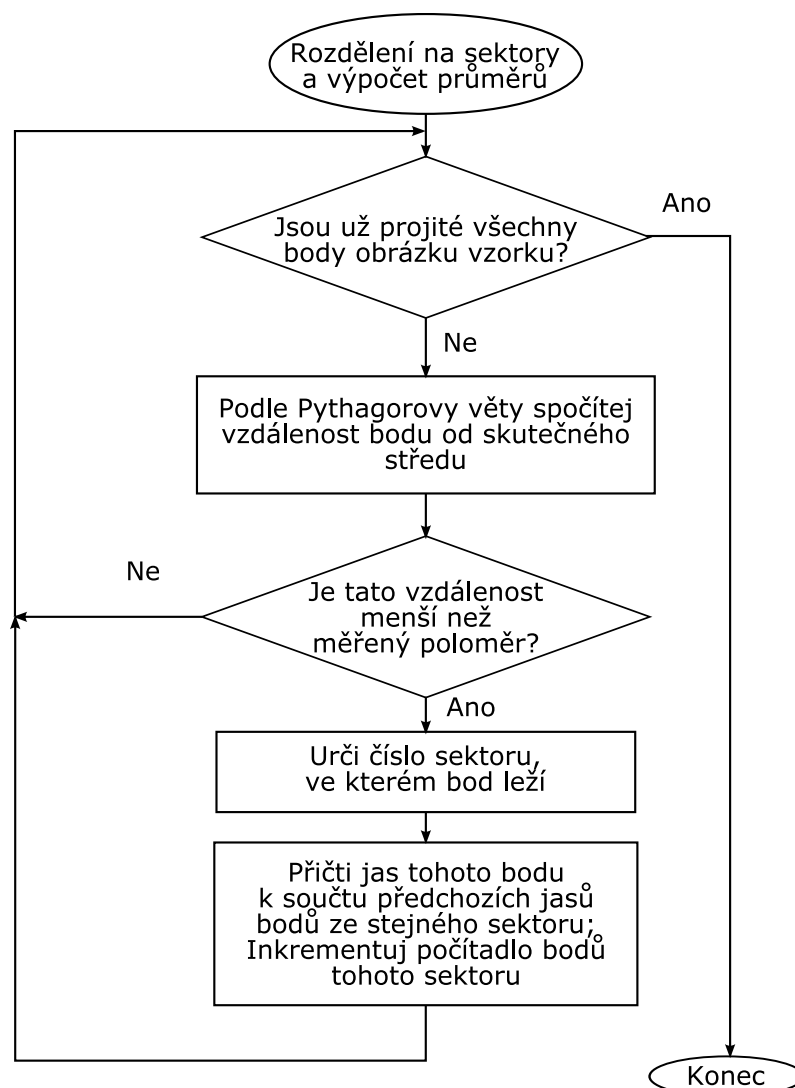
Obrázek 3.5: Vývojový diagram hledání středu vzorku v programu Analyzér

se soustřednými kružnicemi v obrázku vzorku. To znamená, že klikem do plochy `ImgStred` upřesňuje volbu polohy středu. Kliknutím na červený bod tedy zajistí, že kříž je umístěn na pozici skutečného středu vzorku.

### 3.2.3 Rozdělení na sektory a výpočet průměrů

Jak je vidět na obrázku 3.2, tak je v okně Hledání středu možné nastavit měřený poloměr a zvolit počet sektorů, na které bude obrázek vzorku rozdělen. S těmito zvolenými hodnotami se pracuje v okamžiku, kdy uživatel klikne na tlačítko Zavřít. Procedura, která probíhá mezi tím, než se zavře Okno hledání a otevře Okno výpis, je schematicky zobrazena na obrázku 3.6.





Obrázek 3.6: Vývojový diagram rozdělení obrázku na sektory a výpočtu průměrných hodnot

Výsledkem této procedury jsou součty jasů ve stejné vzdálenosti od středu pro každý sektor zvlášť a taky počty bodů pro každý sektor ve stejné vzdálenosti. Hodnoty, které jsou zobrazeny v okně výpis (obrázek 3.3 a)) jsou tedy podílem součtu jasů bodů v určitém sektoru a vzdálenosti ku počtu těchto bodů. Jsou to tedy průměrné hodnoty.

Tyto hodnoty jsou v okně 3.3 a) vypsány do komponenty `StringGrid` (`TStringGrid`). Kromě toho jsou však ještě vypsány do instance komponenty `TMemo`, což je vpod-

statě pole pro víceřádkový text. Tato instance je však neviditelná (vlastnost `Visible` je nastavena na hodnotu `False`). To jen proto, že má metodu `SaveToFile`, která uloží textový obsah dané instance do textového souboru. Dialogové okno pro výběr souboru, do kterého budou hodnoty uloženy, je vyvoláno po kliknutí na tlačítko `Export do textového souboru` v okně `Výpis` (obrázek 3.3 a)). Takto vyexportovaná data budou potřeba v programu `Cross-Corel` v kapitole 4.

## 4 Program Cross-Corel

Program Cross-Corel byl vytvořen v programovacím jazyce Java v prostředí Eclipse Europe, o kterých je zmínka v části 2.2. Jeho funkce spočívá v načtení naměřených hodnot z programu Analyzér (část 3.1), jejich upravení ve smyslu případného posunutí a roztáhnutí. Výstupem je pak finální závislost  $f(r)$ , která je průměrem upravených průběhů.

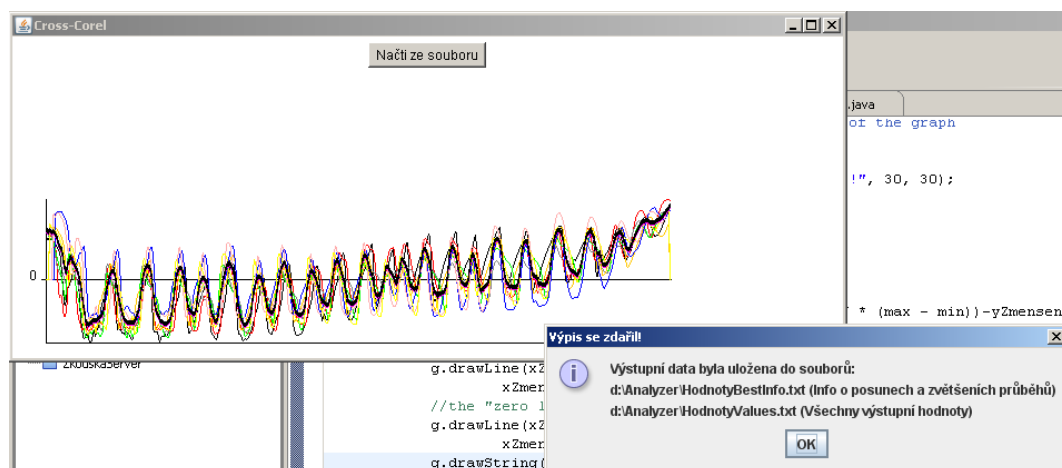
### 4.1 Uživatelský návod pro ovládání programu Cross-Corel

Ovládání programu Cross-Corel je jednodušší, než v případě Analyzéro kruhových obrázků z části 3.1. Program Cross-Corel je poměrně jednoduchý. Jedinou uživatelskou akcí je v podstatě zadání cesty k souboru, ve kterém jsou uloženy hodnoty z programu Analyzér. Vše ostatní už program udělá automaticky.

Po spuštění programu je zobrazeno prázdné okno s jedním tlačítkem a plochou pro kreslení grafiky, ve které budou zobrazeny grafy závislostí  $f_i(r)$ . Přes tuto plochu jsou vykresleny uhlopříčně dvě černé úsečky, které v tuto chvíli (po otevření programu) symbolizují, že není načten žádný soubor.

Po kliknutí na tlačítko s nápisem *Načti ze souboru* se otevře dialogové okno pro výběr souboru. Zde uživatel najde soubor, který chce otevřít. Mělo by se jednat o soubor vytvořený programem Analyzér – tedy textový soubor, kde jsou hodnoty jednotlivých průběhů uloženy ve sloupcích oddělených mezerami. Těchto sloupců by nemělo být víc než osm.

Když je vybrán soubor pomocí otevíracího dialogu, tak program načte hodnoty ze zadaného souboru a zpracovává je. Konec zpracování je patrný například na obrázku 4.1.

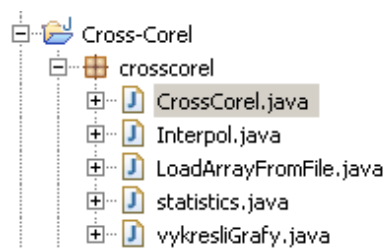


Obrázek 4.1: Okno programu Cross-Corel v okamžiku dokončení vzájemně korelační funkce

Dialogové okno na tomto obrázku s nadpisem *Výpis se zdařil!* informuje o tom, kde byly uloženy soubory s výstupními hodnotami. V našem případě se soubor se vstupními hodnotami jmenoval *Hodnoty.txt*, umístěný v `D:\Analyzer\`. Výstupem programu Cross-Corel jsou dva textové soubory. Vždy to jsou soubory, jejichž název začíná jménem vstupního souboru bez přípony – v případě vstupního souboru *Hodnoty.txt* tedy *Hodnoty* a pro soubor, ve kterém jsou uloženy informace o tom, jak byly jednotlivé průběhy posunuty nebo změněna jejich velikost je k tomuto názvu přidán řetězec *BestInfo* a je uložen s příponou *.txt*. V tomto souboru se nachází informace o tom, jak byly jednotlivé průběhy posunuty a roztáhnuty oproti prvnímu referenčnímu průběhu. Jméno druhého souboru také začíná jménem vstupního souboru bez přípony a je k němu přidán řetězec *Values*. V tomto souboru jsou uloženy všechny výstupní hodnoty, tedy všechny průběhy  $f_i(r)$  už posunuté a se změněnou velikostí. Jako poslední (5. v případě 4 průběhů, 9. v případě 8 průběhů) je k nim vypsán průměrný průběh  $f(r)$ . Oba dva tyto soubory jsou zapsány tam, kde se nacházel zdrojový soubor. V našem případě tedy `D:\Analyzer\`.

## 4.2 Cross-Corel z programátorského hlediska

Program Cross-Corel je vytvořen v programovacím jazyce Java (2.2) v IDE Eclipse (2.2.1). Skládá se z pěti unikátních tříd, které se nachází v jednom package. Hierarchická struktura je patrná na obrázku 4.2.



Obrázek 4.2: Hierarchická struktura programu Cross-Corel

Funkce jednotlivých tříd v programu Cross-Corel:

- **CrossCorel.java**: Hlavní vizuální třída, obsahuje metodu `Main`, jejíž obsah se provádí po spuštění celé aplikace. Její instance představuje hlavní okno, jak je vidět na obrázku 4.1 bez informačního okna. Tato třída obsahuje instanci tlačítka, jehož obsluha události kliknutí provádí celou vzájemně korelační funkci.
- **LoadArrayFromFile.java**: Instanci této třídy je potřeba zadat do konstruktoru jako parametr jméno souboru. Tato implementuje vstupní proud ze souboru (třída `DataInputStream`) a načte z tohoto souboru číselná data. Její další funkce je, že tato data uloží do pole polí (analogie dvojrozměrného pole), ve kterém první index znamená o který průběh se jedná a druhý už určuje pozici v příslušném průběhu.
- **VykresliGrafy.java**: Tato třída je potomkem třídy `Canvas`, která představuje plátno pro kreslení grafických objektů, jako jsou úsečky, elipsy, obdélníky a

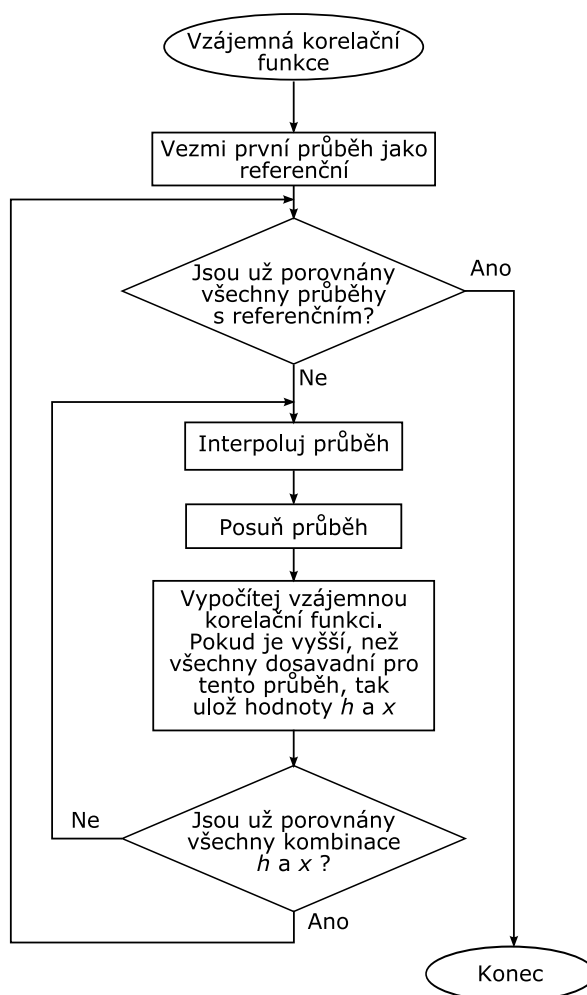
podobné. Její instance je právě plocha na kterou jsou vykresleny grafy a osy grafu. Tato třída musí obsahovat metodu `Paint`, ve které je kód vykreslování. Tato metoda je automaticky volána v okamžiku, když se změní hodnoty grafu (pole polí), který je vykreslen, nebo když je změněna velikost plochy `Canvasu` (případně i celého okna aplikace).

- `statistics.java`: Tato třída reprezentuje dvě jednorozměrná pole a operace s nimi. Jedná se o normalizaci hodnot v poli (odečtení průměru daného průběhu od každé jeho hodnoty) a porovnání těchto dvou průběhů vzájemně korelační funkcí. Metoda této třídy `GetCorelValue` vrací právě hodnotu vzájemně korelační funkce podle vzorce (4).
- `Interpol.java`: V této třídě se nacházejí metody, které přepočítávají hodnoty pole podle zadaných parametrů. Jedná se o posun a roztažení, případně zmenšení ve směru osy  $x$ . K těmto výpočtům jsou zapotřebí parametry  $x$  (posunutí) a  $h$  jako zvětšení, které udává, kolikrát bude interpolovaný průběh delší, než původní.

Celá vzájemně korelační funkce se provede po stisku tlačítka *Načti ze souboru*. Nejprve musí uživatel zadat cestu k souboru prostřednictvím dialogového okna, které je instancí třídy `JFileChooser`. V okamžiku, kdy je známa cesta k souboru s hodnotami, tak se s touto cestou instanciuje třída `LoadArrayFromFile`, která načte obsah souboru vytvořeného programem Analyzér z naměřených hodnot do pole polí, reprezentovaného Java kolekcí `ArrayList<ArrayList<Integer>>`, která se v programu jmenuje `Vsechno`. Z tohoto pole se pak porovnávají všechny průběhy s prvním. To tím způsobem, že se vždy uvažuje první průběh `Vsechno.Get(0)` jako referenční. Pak se mění vlastnosti druhého průběhu (posunutí a zvětšení) a vždy se spočítá hodnota vzájemné korelační funkce. Pro určité zvětšení (v rozmezí od 0,9 do 1,1 s krokem 0,005) se průběh posunuje od pozice -20 do pozice +20 s krokem 1 bod. Pro každou pozici při každém zvětšení se počítá hodnota vzájemné korelační funkce a program hledá maximální hodnotu. Toto maximum pak ukládá do souboru, jak je

zmíněno v části 4.1.

Takto program porovnává všechny průběhy s referencím a pro každý získá optimální dvojici hodnot posunutí  $x$  a zvětšení  $h$ . Zobrazené grafy, jak je vidět na obrázku 4.1 jsou již interpolované průběhy podle těchto nejlepších nalezených hodnot. Průběh vykreslený silnější černou čarou je průměrný průběh ze všech čtyř nebo osmi interpolovaných. Do tohoto průměru se nezapočítávají nuly, které vznikly posunem dílčích průběhů na libovolnou stranu ve směru osy  $x$ . Schematický vývojový diagram je uveden na obrázku 4.3



Obrázek 4.3: Vývojový diagram vzájemné korelační funkce v programu Cross-Corel

## Závěr

Byla představena metoda obrazové analýzy radiálně symetrických vzorků. Tato metoda si klade za cíl stanovit závislost průměrných jasů stejně vzdálených od středu daného symetrického vzorku. Toho se podařilo dosáhnout i přes vyskytlé problémy, jako je deformace daného vzorku v určitém směru.

Pro realizaci obrazové analýzy byly vytvořeny dva programy. Jednak v programovacím prostředí IDE Delphi program Analyzér, který stanoví funkce  $f_i(r)$  pro daný vzorek pro potřebný počet sektorů. Pak program Cross-Corel, který byl napsán v programovacím jazyce Java, který průběhy získané programem Analyzér upraví formou posunu a zvětšení právě proto, že jsou vzorky deformované.

Výsledná funkce  $f(r)$ , což je průměrná funkce, která už eliminovala defomaci vzorku, je vyexportována programem Cross-Corel do textového souboru, se kterým lze dále pracovat.



## Závěr v angličtině

Image analysis of radial symmetrical samples was presented. The goal of this method is to obtain the dependence of average brightness on distance from the real centre in the radial symmetrical sample. This was succed through the problems, like deformations of the sample in some direction.

For the image analysis realization was two programs created. The Analyzer in IDE Delphi which defines the  $f_i(r)$  functions for the sample for needed number of radial sectors. Next the Cross-Corel in the programming language Java which the mesasured values from the Analyzer modify (scale and zoom in  $x$  axis) because the samples can be deformed.

The final average function  $f(r)$ , which elminates the deformation is exported by the Cross-Corel program to the text file for the further analysing and working.

## Literatura

- [1] Winslow W.M., Induced fibrillation of suspension, *J. Appl. Phys.* 20 (1949) pp. 1137–1140.
- [2] Parthasarathy M. and Klingberg D.J., Electrorheology: mechanisms and models, *Mater. Sci. Eng. R* 17 (1996) pp. 57–103.
- [3] Hao T., Electrorheological fluids, *Adv. Mater.* 13 (2001) pp. 1847–1857.
- [4] Hao T., Electrorheological suspensions, *Adv. Colloid Interface Sci.* 97 (2002) pp. 1–35.
- [5] Henley S. and Filisko F.E., Flow profiles of electrorheological suspensions: An alternative model for ER activity, *J. Rheol.* 43 (1999) pp.1323–1336.
- [6] Filisko F. E., Henley S. and Quist G., Recent developments in the properties and composition of electrorheological fluids, *J. Intelligent Mater. Syst. Struct.* 10 (1999) pp. 476–480.
- [7] Wolfram, S., *The Mathematica book*, 3<sup>rd</sup> ed., Cambridge, New York: Cambridge University Press (1996)
- [8] Herout, P., *Učebnice jazyka Java*. Praha: Kopp, 2000, ISBN 80-7232-115-3
- [9] Kadlec, V., *Učíme se programovat v Delphi a jazyce Object Pascal*. Praha: Computer Press, 2001, ISBN 80-7226-245-9

## Seznam použitých symbolů a zkratek

**Rheologie** je věda, která se zabývá studiem deformací hmoty.

**Px** Pixel (zkrácení anglických slov picture element, obrazový prvek) je nejmenší jednotka digitální rastrové (bitmapové) grafiky.

**Intenzita elektrického pole** (elektrická intenzita)  $E [V/m]$  je fyzikální veličina, vyjadřující velikost a směr elektrického pole. Je definována jako elektrická síla působící na těleso s kladným jednotkovým elektrickým nábojem.

**Viskozita** je fyzikální veličina, udávající poměr mezi tečným napětím a změnou rychlosti v závislosti na vzdálenosti mezi sousedními vrstvami při proudění skutečné kapaliny.

**IDE** Integrated development environment (Integrované vývojové prostředí) je software usnadňující práci programátorů, většinou zaměřený na jeden konkrétní programovací jazyk.

**SDK** software development kit (sada pro vývoj software) je to souprava nástrojů pro vývoj aplikací, uživatelského rozhraní či frameworku pro určitou platformu či hardware.

**GUI** Graphical user interface (grafické uživatelské rozhraní) je druh komunikace s počítačem mající podobu interaktivních grafických prvků.

## Seznam obrázků

1.1	Řetězcové struktury . . . . .	11
1.2	Elektroreologické vzorky v závislosti na smykové rychlosti . . . . .	13
1.3	Středová oblast elektroreologického vzorku . . . . .	14
1.4	Výsledek funkce $S(x_c, y_c)$ . . . . .	15
1.5	Dělení vzorku na sektory . . . . .	16
1.6	Srovnání dvou průběhů funkce $f_i(r)$ . . . . .	17
1.7	Srovnání funkcí $f_1(r)$ a $f_2(hr + x)$ . . . . .	18
1.8	Průměrná $f(r)$ funkce. . . . .	19
1.9	Centrální část původního obrázku (a) a odpovídající výsledek obrazové optimalizace (b) . . . . .	19
2.1	Obrazovka IDE Delphi 7 . . . . .	21
2.2	Okno IDE Eclipse Europa . . . . .	23
3.1	Okno programu Analyzér kruhových vzorků s otevřeným kruhovým vzorkem . . . . .	26
3.2	Okno s výsledky hledání středu a možnostmi nastavení . . . . .	27
3.3	Okno s naměřenými hodnotami a) numerickými a b) grafickými . . . . .	28
3.4	Vývojový diagram otevírání obrázku vzorku v programu Analyzér . . . . .	30
3.5	Vývojový diagram hledání středu vzorku v programu Analyzér . . . . .	32
3.6	Vývojový diagram rozdělení obrázku na sektory a výpočtu průměrných hodnot . . . . .	33

---

4.1	Okno programu Cross-Corel v okamžiku dokončení vzájemně korelační funkce . . . . .	36
4.2	Hierarchická struktura programu Cross-Corel . . . . .	37
4.3	Vývojový diagram vzájemné korelační funkce v programu Cross-Corel	39

## Seznam příloh

- P I Zdrojový kód hledání středu
- P II Zdrojový kód vzájemně korelační funkce pro dva průběhy funkce  $f_i(r)$
- P III Obsah přiloženého CD
- P IV Výsledky analýzy 4 vzorků
- P V Ponížil P., Pavlínek V., Kitano T., Dřímál T.: Image analysis of radial symmetrical samples. In: Revetria R. et al. (eds.) System science and simulation in engineering, proceedings of 6th WSEAS international conference on System science and simulation in engineering (ICOSSE '07), Venice, Italy, November 21-23, 2006, p. 330-333. ISBN: 978-960-6766-14-5
- P VI Ponížil P., Pavlínek V., Kitano T., Dřímál T.: Image analysis of electrorheological flow patterns. International Journal of Mathematical Models and Methods in Applied Sciences, 1, 4 (2007) 239-242 ISSN: 1998-0140

## Příloha P I: Zdrojový kód hledání středu

```
Begin
Suma:=0;
Vaha:=0;
SY:=-1;
SX:=-1;
Procent:=-1;
  If kriz Then begin

    For Y1:=StredY-PulPole to StredY+PulPole Do Begin
      Inc(SY);
      inc (Procent);
      FrmHledej.Caption:='Working. Finished'+IntToStr(Procent)+'%';
      Application.ProcessMessages;
      for X1:=StredX-PulPole to (StredX+PulPole-1) do Begin
        Inc(SX);

        for B:=Y1-PulTestPole to Y1+PulTestPole do Begin
          Hodnota:=B+(2*(Y1-B));
          for A:=X1-PulTestPole to X1+PulTestPole do Begin
            V:=sqr(X1-A)+sqr(Y1-B)+1;
            Suma:=(Suma+sqr((Obrazek[A,B]-Obrazek[2*X1-A,
              Hodnota]))/V);

            Vaha:=Vaha+(1/V);
          End;
        End;
      End;
      SO[SX,SY]:=Sqrt(Suma/Vaha);
      Suma:=0;
      Vaha:=0;
    End;
    SX:=-1;
  End;

  End;
FrmHledej.Caption:='Finished!';
End;
```

## Příloha P II: Zdrojový kód vzájemně korelační funkce pro dva průběhy funkce $f_i(r)$

```
public class statistics {
private ArrayList<Integer> pole1 = null;
private ArrayList<Integer> pole2 = null;
private float[] prumery = new float[2];
private float sigma1 = 0;
private float sigma2 = 0;

public statistics (final ArrayList<Integer> arg0,
                  final ArrayList<Integer> arg1){

    pole1 = arg0;
    pole2 = arg1;
    float sum1 = 0;
    float sum2 = 0;
    //calculate the averages of all values
    //from the array for every array extra
    for (int i = 0;i<pole1.size();i++){
        sum1 +=pole1.get(i);
    }
    for (int i = 0;i<pole2.size();i++){
        sum2 += pole2.get(i);
    }

    prumery[0] = sum1/pole1.size();
    prumery[1] = sum2/pole2.size();

    for (int i = 0;i<pole1.size();i++){
        if (pole1.get(i)!=0){
            pole1.set(i, Math.round(pole1.get(i) - prumery[0]));
            sigma1 += (pole1.get(i)*pole1.get(i));
        }
    }
    for (int i = 0;i<pole2.size();i++){
        if (pole2.get(i)!=0){
            pole2.set(i, Math.round(pole2.get(i) - prumery[1]));
            sigma2 += (pole2.get(i)*pole2.get(i));
        }
    }
    sigma1 = (float) Math.sqrt(sigma1) ;
    sigma2 = (float) Math.sqrt(sigma2);
    /**
     * in every element in array pole1 and pole2
```



```

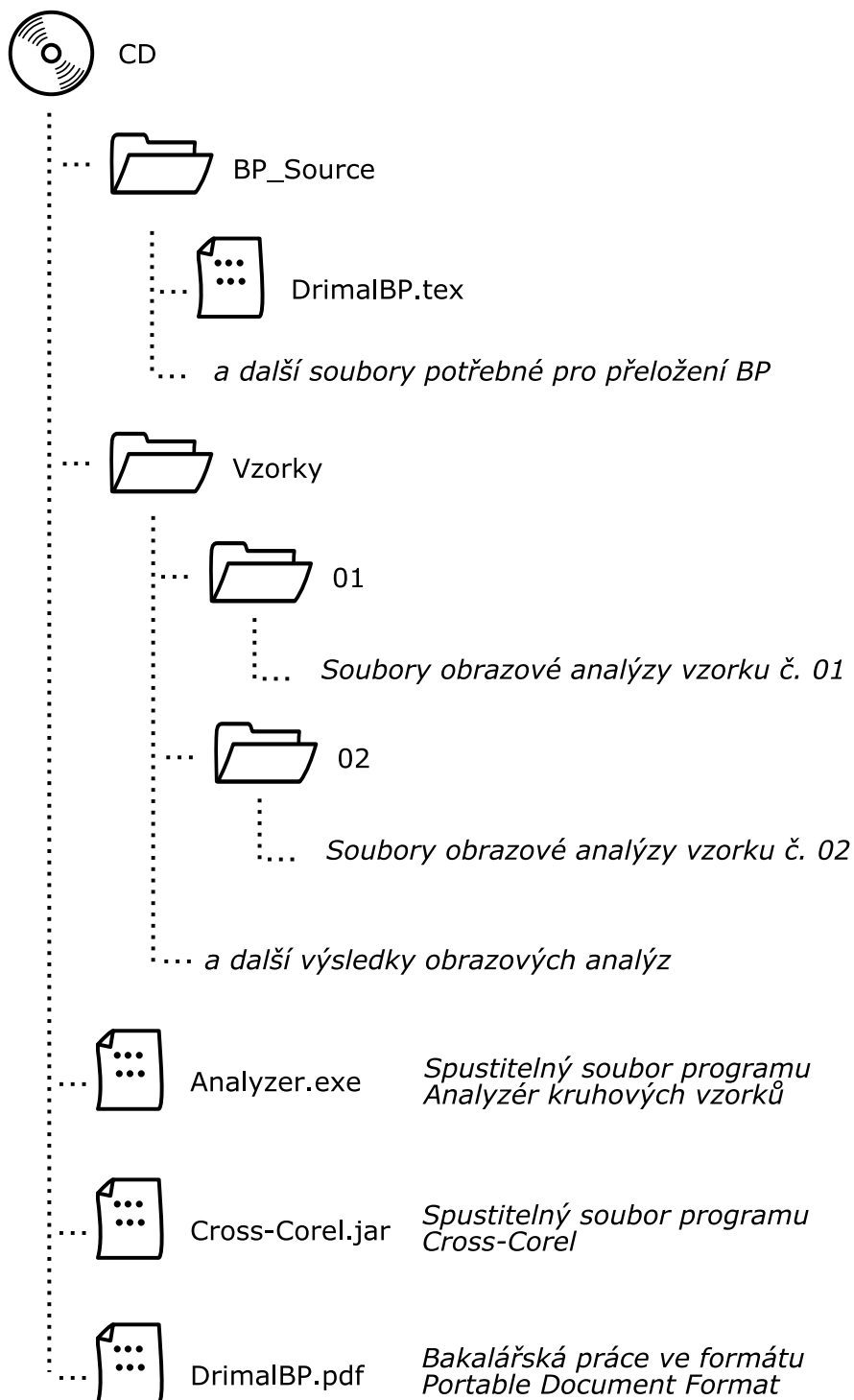
        * is subtracted average from original value
        */
    }
    /**
     *
     * @returns the value of correlation two arrayLists
     */
    public float getCorelValue(){
        int delka = 0;
        float hodnota = 0;
        float vysl = 0;
        if (pole1.size() > pole2.size()) delka = pole1.size();
        else delka = pole2.size();

        for (int i = 0; i < delka; i++){
            hodnota = (float) (pole1.get(i) * pole2.get(i));
            vysl = vysl + hodnota;
        }

        return vysl / (sigma1 * sigma2);
    }
    public ArrayList<ArrayList<Integer>> getArrays(){
        ArrayList<ArrayList<Integer>> pole =
            new ArrayList<ArrayList<Integer>>();
        pole.add(pole1);
        pole.add(pole2);
        return pole;
    }
}

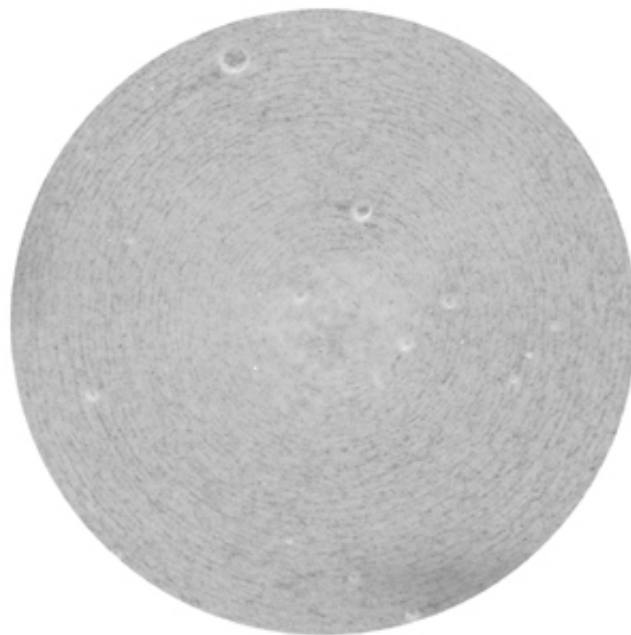
```

## Příloha P III: Obsah přiloženého CD

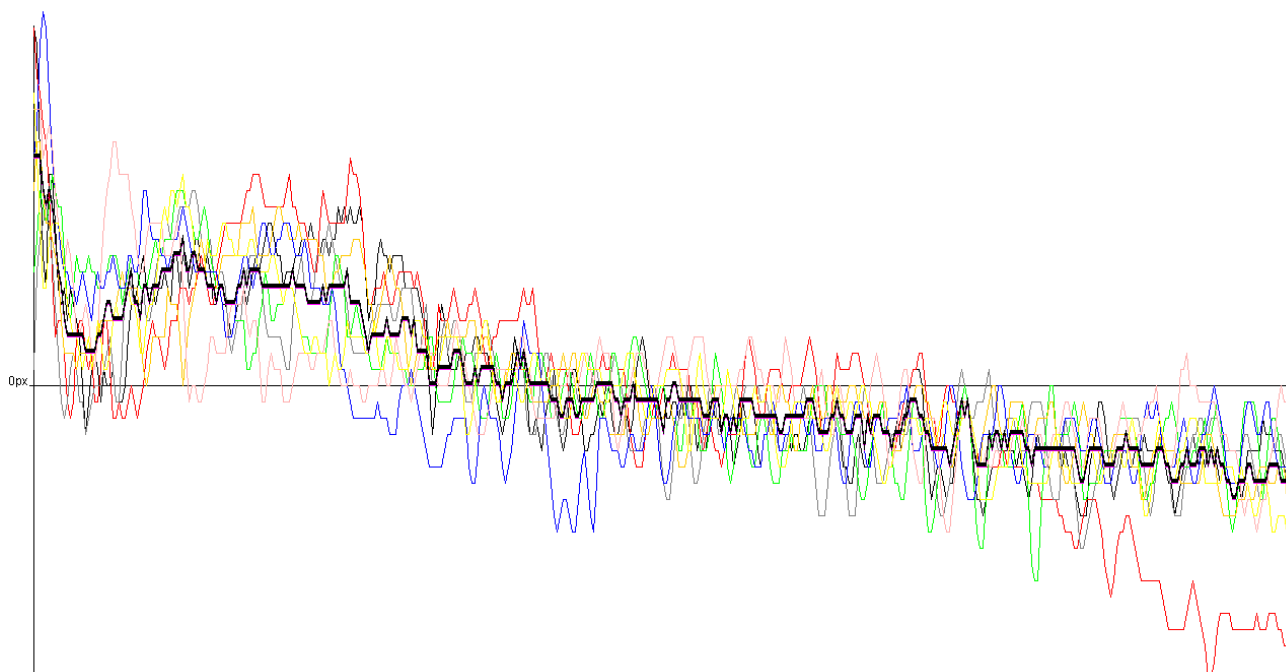


## Příloha P IV: Výsledky analýzy 4 vzorků

## Vzorek 01

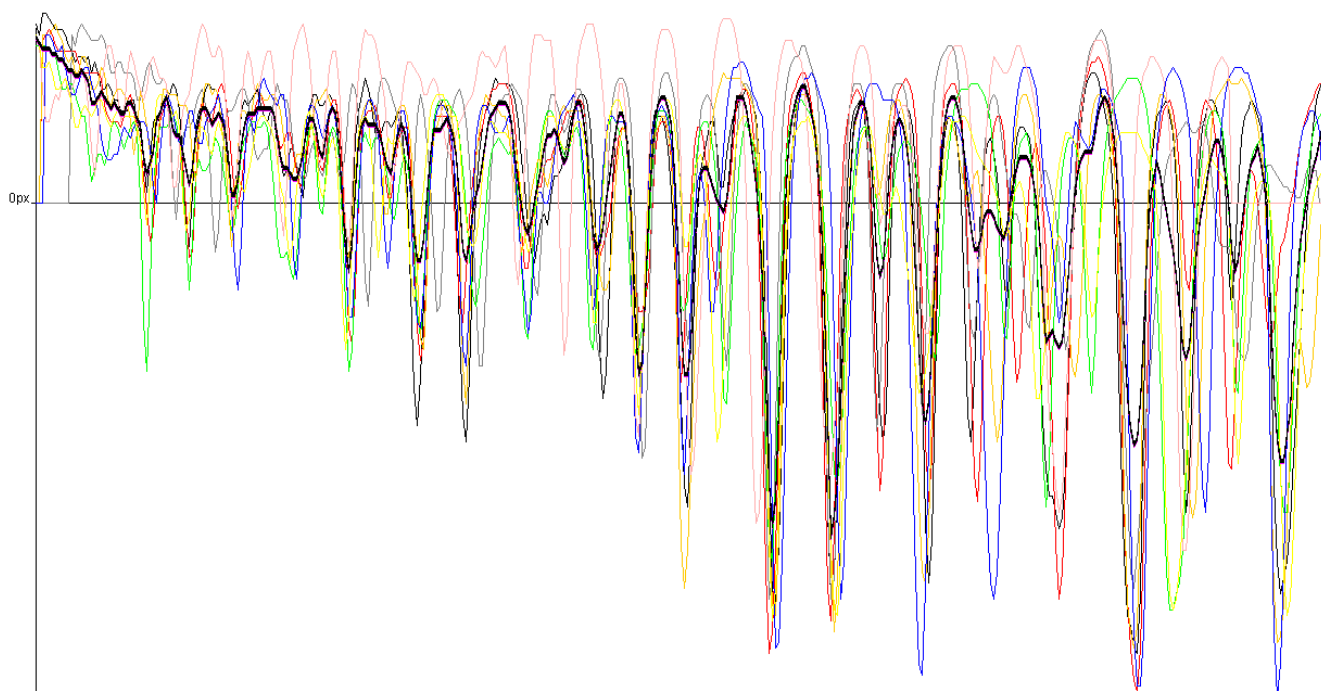


číslo vzorku: 01  
Název vzorku: 40-001-10-1  
Koncentrace skleněných kuliček: 40 %  
Smyková rychlost:  $0,1 \text{ s}^{-1}$   
Intenzita elektrického pole: 1 kV/mm



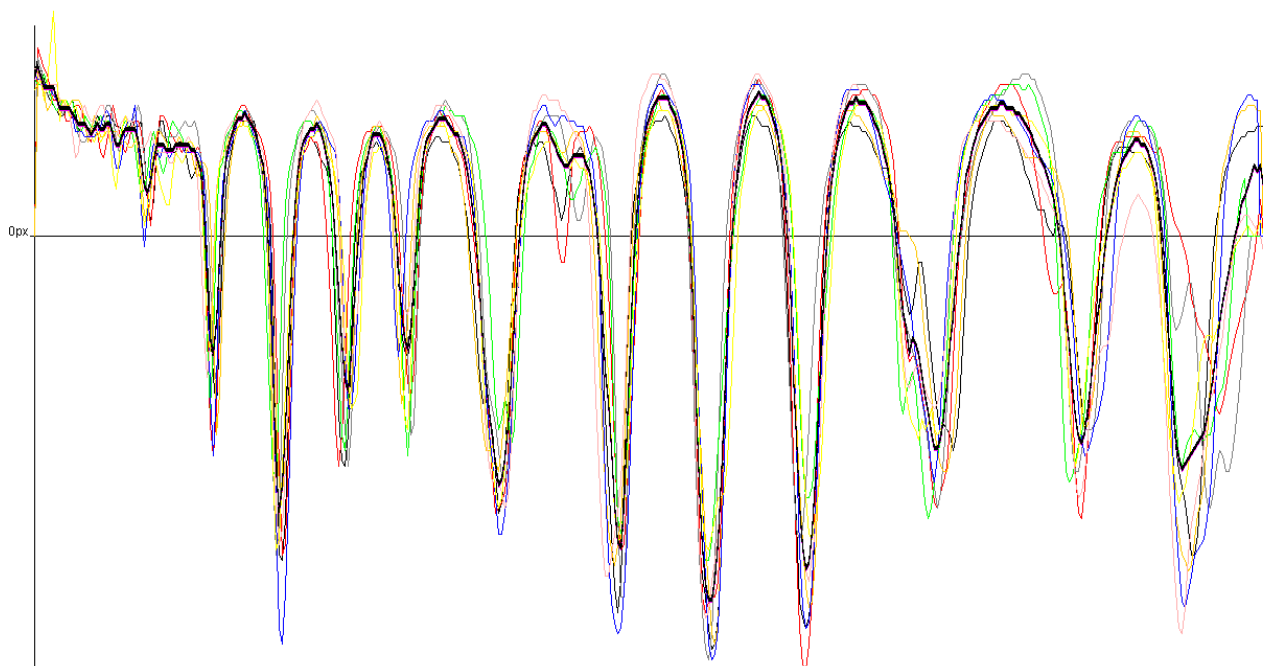
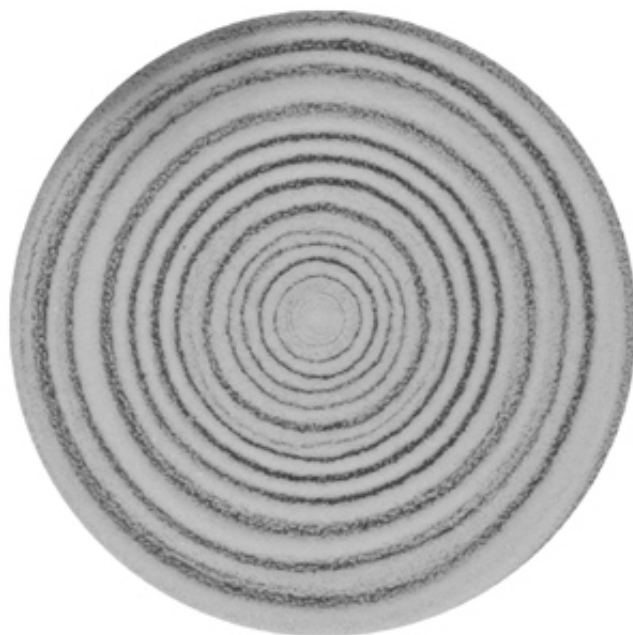
## Vzorek 02

číslo vzorku: 02  
Název vzorku: 40-010-10-2  
Koncentrace skleněných kuliček: 40 %  
Smyková rychlost:  $1,0 \text{ s}^{-1}$   
Intenzita elektrického pole:  $1 \text{ kV/mm}$

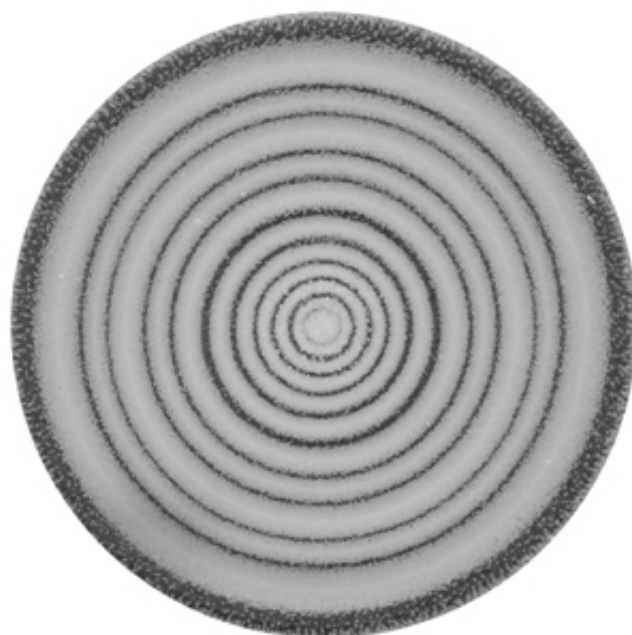


## Vzorek 03

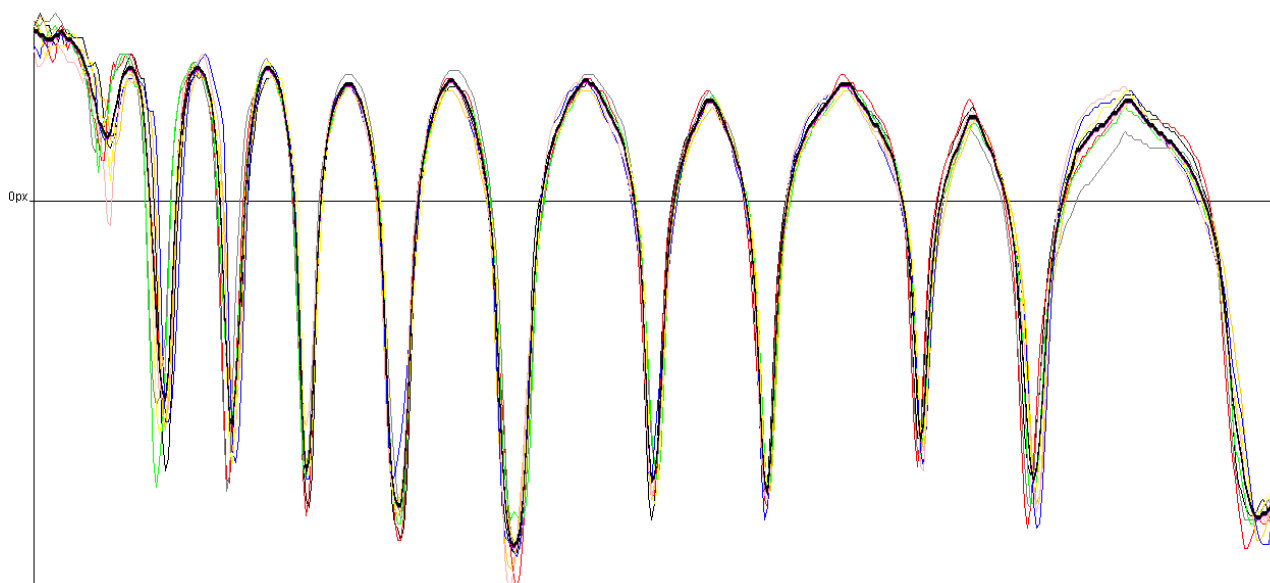
číslo vzorku: 03  
Název vzorku: 40-030-10-1  
Koncentrace skleněných kuliček: 40 %  
Smyková rychlost:  $3,0 \text{ s}^{-1}$   
Intenzita elektrického pole:  $1 \text{ kV/mm}$



## Vzorek 04



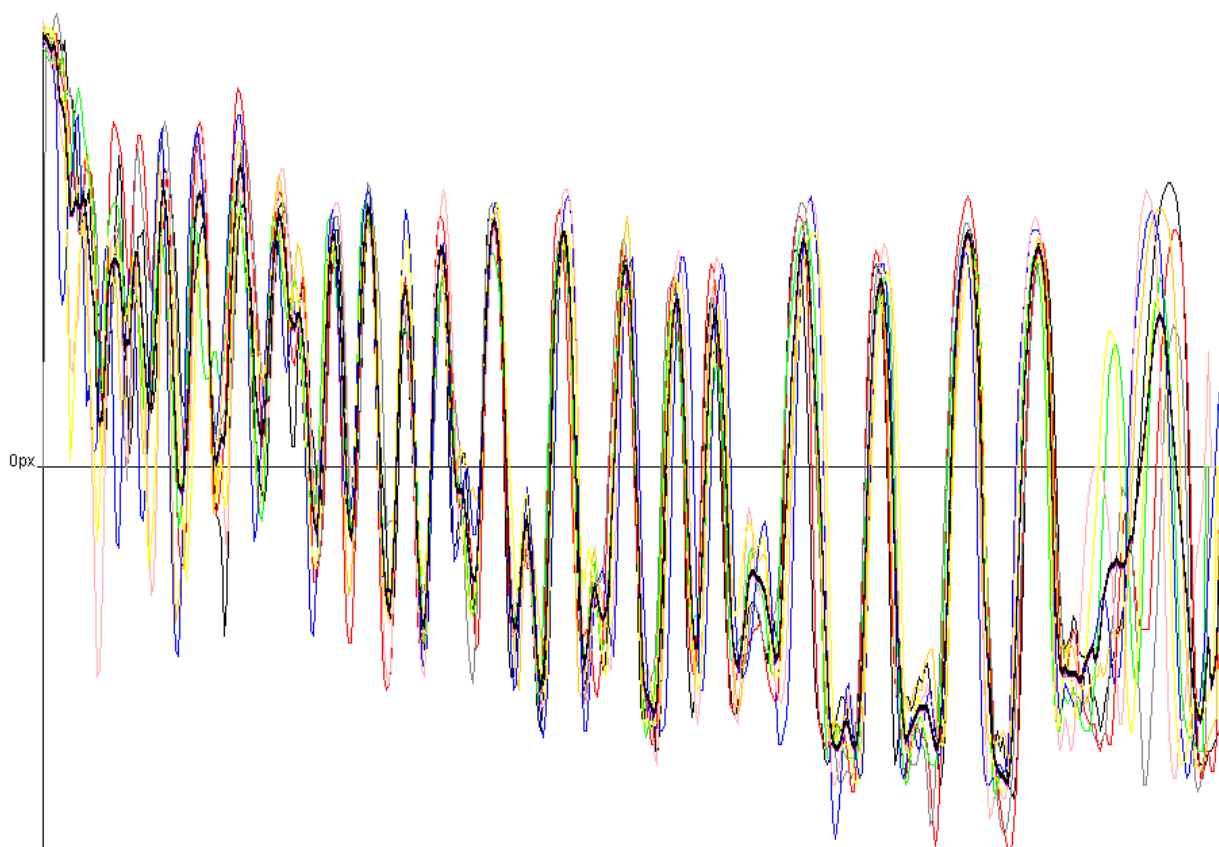
číslo vzorku: 04  
Název vzorku: 40-030-10-1  
Koncentrace skleněných kuliček: 40 %  
Smyková rychlost:  $3,0 \text{ s}^{-1}$   
Intenzita elektrického pole:  $1 \text{ kV/mm}$



## Vzorek 05



číslo vzorku: 05  
Název vzorku: 20-030-10-1  
Koncentrace skleněných kuliček: 20 %  
Smyková rychlost:  $3,0 \text{ s}^{-1}$   
Intenzita elektrického pole: 1 kV/mm

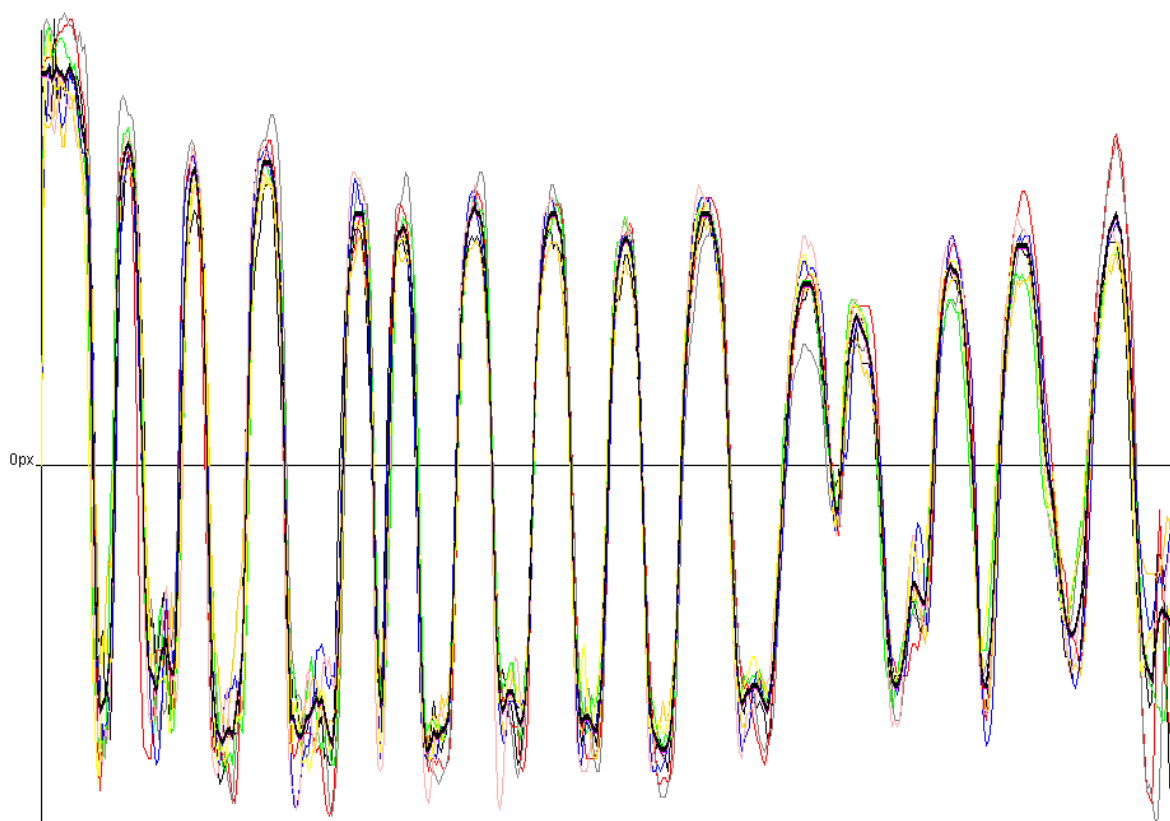




## Vzorek 06



číslo vzorku: 06  
Název vzorku: 20-300-10-1  
Koncentrace skleněných kuliček: 20 %  
Smyková rychlost:  $30 \text{ s}^{-1}$   
Intenzita elektrického pole:  $1 \text{ kV/mm}$



Příloha P V: Ponížil P., Pavlínek V., Kitano T., Dřímál T.: Image analysis of radial symmetrical samples. In: Revetria R. et al. (eds.) System science and simulation in engineering, proceedings of 6th WSEAS international conference on System science and simulation in engineering (ICOSSE '07), Venice, Italy, November 21-23, 2006, p. 330-333. ISBN: 978-960-6766-14-5

# Image Analysis of Radial Symmetrical Samples

PETR PONÍŽIL<sup>1</sup>, VLADIMÍR PAVLÍNEK<sup>2</sup>, TAKESHI KITANO<sup>2</sup>, TOMÁŠ DŘÍMAL<sup>3</sup>

<sup>1</sup>Faculty of Technology,  
<sup>2</sup>Polymer Centre,  
<sup>3</sup>Faculty of Applied Informatics,  
Tomas Bata University in Zlin,  
TGM 275, 762 72 Zlin,  
CZECH REPUBLIC

ponizil@centrum.cz <http://fyzika.ft.utb.cz/personal/petr/>

*Abstract:* A method of image analysis of images, which are developed from electrorheological samples, is presented. Due to the process of preparation, electrorheological samples show a radial symmetry. Numerical transformations are necessary to remove sample deformation and obtain correct radial dependence of image intensity as the function characterizing the sample image.

*Key-Words:* Image analysis, radial symmetry

## 1 Introduction

Electrorheological (ER) fluids are systems whose rheological properties (viscosity, yield stress, shear modulus) can be controlled by external electric field. For the fluids researched so far, ER effect usually causes a continuous increase in viscosity, or immediate (in milliseconds) solidification of the material. This is called a positive electrorheological effect. However, also the opposite effect has been described for several systems – a decrease of viscosity in electric field (a negative ER effect). These facts indicate a wide range of possible practical applications, therefore understanding of the phenomena is highly desirable.

The positive ER effect, which was described by Winslow (1) more than 50 years ago, has been in focus of many papers concentrating on the elucidation of the mechanism, preparation of ER fluids with optimum efficiency and their possible use. The research findings can be found in a number of comprehensive reviews (2–4).

Majority of electrorheological fluids are suspensions of solid particles in electrically non-conducting fluid (usually mineral or vegetable oils). First dispersed phase was represented by inorganic or organic materials, water-free or with some portion of water or other activators. Recent generations of ER fluids are based on suspensions of electrically conducting polymers and nanoparticles of various nature, because of their suitable polarizability.

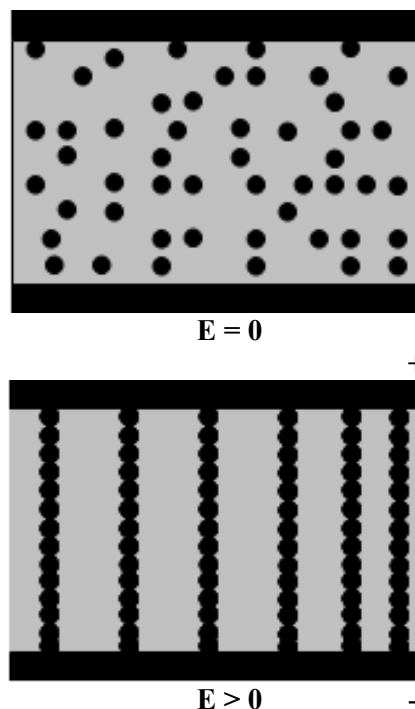


Fig. 1: Scheme of chain structure of ER suspension without flow field.

ER effect is considered to be caused by interfacial polarization of dispersed particles (3,4) produced by external DC or AC electric fields of high intensity ( $E$  in order of kV/mm). Polarized particles are oriented in the direction of electric field and create structures (chains), which increase the rigidity of the originally liquid system (Fig. 1). On the application of flow field, at low shear rates viscosity is high (after exceeding possible yield stress). It is supposed that shear forces cause degradation of the structures created; therefore

viscosity falls with a rise of shear rate and finally, at high shear rates, it gets to the level of zero-field viscosity.

The arrangement of the particle chains in electric field has been proved via optical microscopy. In order to explain the mechanism of reorganization of these structures in electric field during flow, majority of studies so far have followed the dependence of rheological and viscoelastic behaviour (gradient dependence of viscosity, shear stress, viscoelastic moduli) or yield stress of ER fluids in relation to the chemical nature, and physical properties of dispersed particles which affect their polarizability (particle size and shape, DC and AC electric conductivity, permittivity and dielectric loss). However, these results do not provide any evidence of the changes in particle arrangement during flow. On the other hand, direct optical display of ER suspensions in rotational viscometer presented in papers (5,6) indicated that particles organize into lamellar or ring structures, which are optimal for minimum energy dissipation, and this structure depends on the flow field (Fig. 2). These experiments, which have only been carried out for several systems, showed the way to the elucidation of general factors controlling flow mechanism of ER structures.

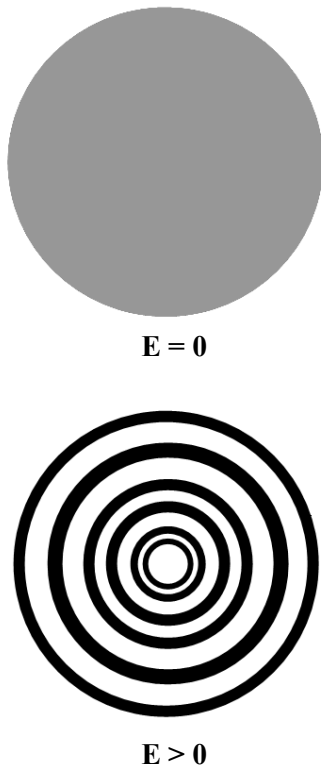


Fig. 2: Example of a structure of electrorheological suspension formed between two rotating parallel plates

## 2 Experiment

### 2.1 Centring

Samples were scanned on commercial scanner and greyscale images as Fig. 3 with typical size 1000x1000 pixels were obtained. The goal is to gain radial dependence of darkness of the image. Suppose that  $p(x,y)$  is brightness of pixel in  $x$ -column and  $y$ -row of the image.

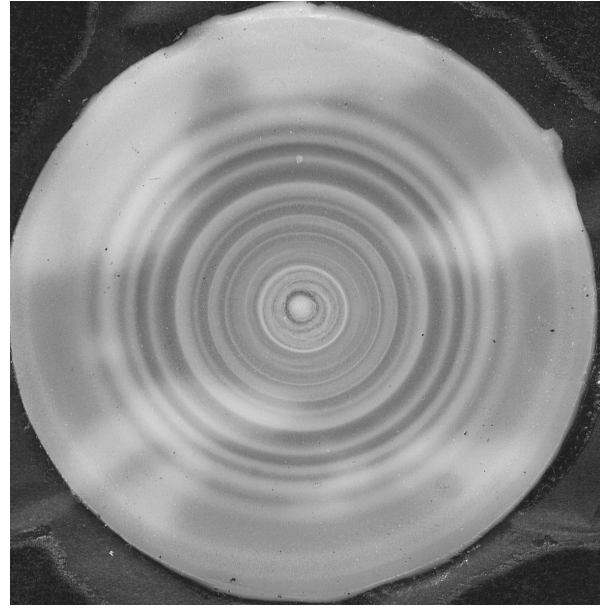


Fig. 3: The scanned image of electrorheological sample.

There are several problems. Samples are relatively soft and easy deformable. The images are a little prolonged in some directions.

The first task is to recognize position of centre of symmetry. The first estimation of the centre position  $(x_{c0}, y_{c0})$  can be made manually or automatically (as the centre of the image). For some neighbourhood a symmetry function  $S(x_c, y_c)$  is computed as

$$S(x_c, y_c) = \sum_{x,y=-200}^{200} \frac{(p(x_c + x, y_c + y) - p(x_c - x, y_c - y))^2}{\sqrt{x^2 + y^2}}$$

where numerator is square of brightness difference of symmetrical pixels and denominator is weight (number of pixels in some distance is proportional to this distance). Fig. 4 shows graphical representation of  $S(x_c, y_c)$  function. Brighter pixels indicate higher value of  $S(x_c, y_c)$  function, cross near image centre indicates position of real symmetry centre – the total maximum of  $S(x_c, y_c)$  function with position  $(x_{cm}, y_{cm})$ .

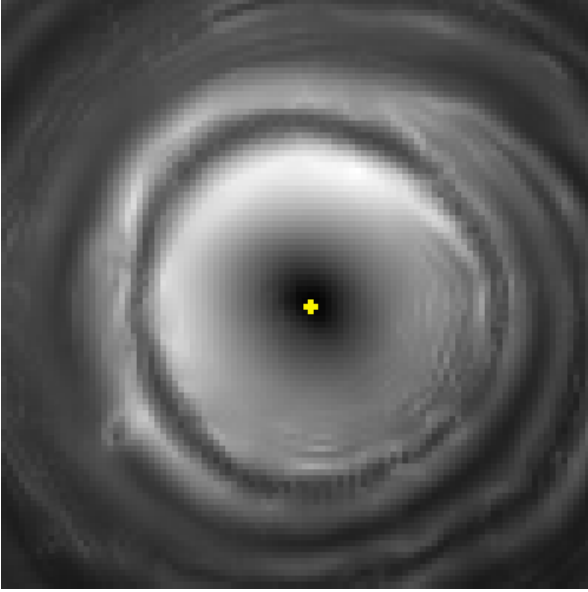


Fig. 4: The graphical representation of  $S(x_c, y_c)$  function.

## 2.2 Transformation

If the centre is found, image is divided into 8 radial sectors and mean brightness as a function  $f_i(r)$  (index  $i$  denotes the sector) of the real centre distance  $r$  is computed. On the fig. 5 is the comparison of such functions for two different sectors of image. It is obvious that a little shift and scale change is necessary to eliminate deformation.

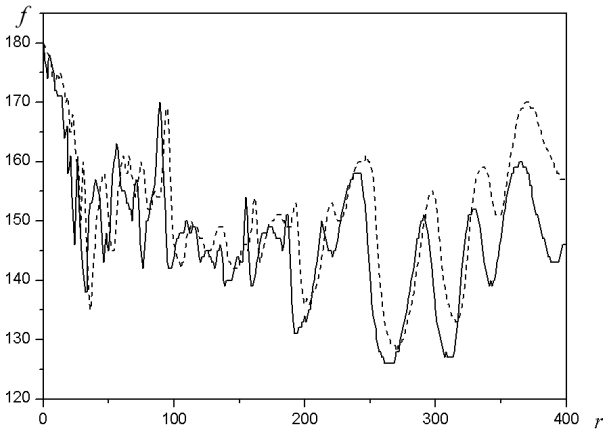


Fig. 5: The comparison of  $f_i(r)$  functions for two sectors. Intensity  $f$  and radial distance  $r$  are both in arbitrary units.

A tool for the examination of the similarity of two functions is cross-correlation function (7). This function is defined for real functions as

$$(f_i * f_j)(x) = \int f_i(r) f_j(r + x) dr .$$

For example, consider two real valued functions  $f_i$  and  $f_j$  that differ only by a shift along the  $x$ -axis. One can calculate the cross-correlation to figure out how much  $f_j$  must be shifted along the  $x$ -axis to make it identical to  $f_i$ . The formula essentially slides the  $f_j$  function along the  $x$ -axis, calculating the integral for each possible amount of sliding. When the functions match, the value of  $(f_i * f_j)$  is maximized. The reason for this is that when lumps (positives areas) are aligned, they contribute to making the integral larger. Also, when the troughs (negative areas) align, they also make a positive contribution to the integral because the product of two negative numbers is positive. For our occasion we define cross-correlation function more generally as

$$(f_i * f_j)(h, x) = \int f_i(r) f_j(hr + x) dr ,$$

where  $h$  is a scale change and  $x$  is a shift. If the value of  $(f_i * f_j)$  is maximized, the best combination of  $h$  and  $x$  is found.

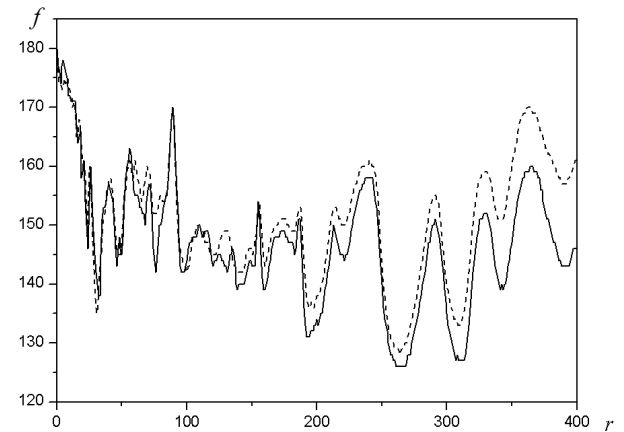


Fig. 6: The original  $f_i(r)$  function and transformed  $f_2(hr+x)$  function. Values  $h = 0.972$ ,  $x = -5.07$  maximize the cross-correlation function.

## 2.3 Results

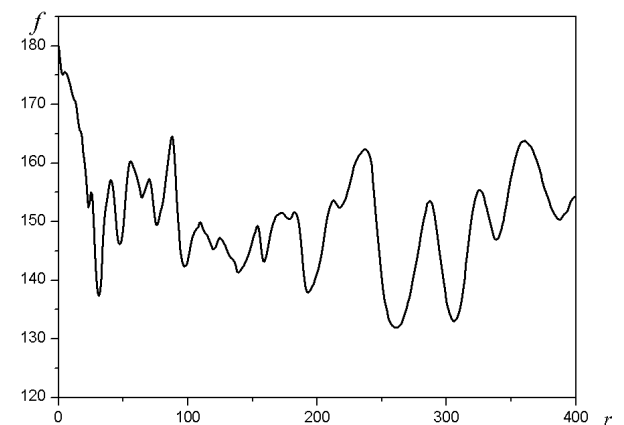
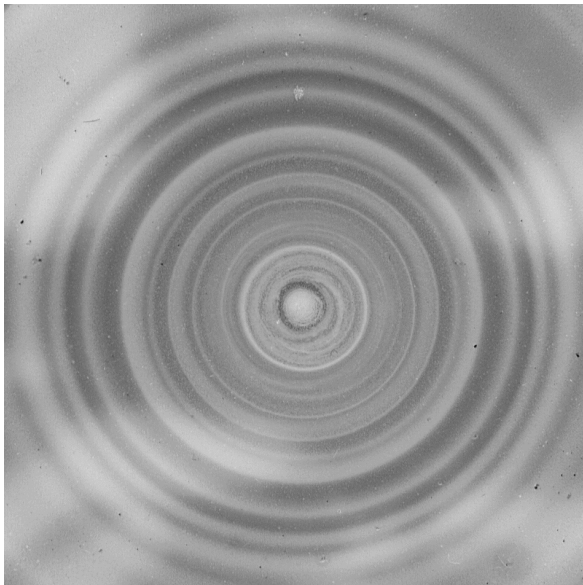


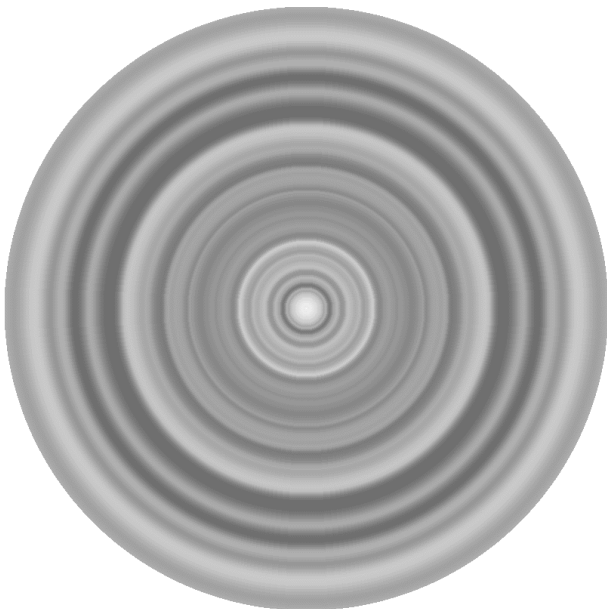
Fig. 7: The averaged  $f(r)$  function.

After rescaling of  $f_j$  functions for each sector, the mean value of one original and all transformed  $f_j$  functions is computed. This way was obtained the average  $f(r)$  function as the dependence of average intensity on radial distance from sample centre and is presented on the Fig. 7.

Fig. 8 shows comparison of the central part of the original sample image and corresponding image based on radial dependence of intensity  $f(r)$  from Fig. 7.



a)



b)

Fig. 8: The central part of the original image (a) and corresponding result of the image optimization (b).

Now the sample is characterized by the  $f(r)$  function as showed on Fig. 7. A dependence of distribution of peaks of the  $f(r)$  function on parameters of samples preparation can be studied.

### 3 Conclusion

A method of the optimization of radial symmetrical images was suggested.

### Acknowledgements

The authors kindly acknowledge the support by the Ministry of Education, Youth and Sport of the CR (project MSM 7088352101), the Czech Science Foundation (projects 106/05/0550 and 202/06/0419).

### References

1. Winslow W.M., Induced fibrillation of suspension, *J. Appl. Phys.* **20** (1949) pp. 1137–1140.
2. Parthasarathy M. and Klingberg D.J., Electrorheology: mechanisms and models, *Mater. Sci. Eng. R* **17** (1996) pp. 57–103.
3. Hao T., Electrorheological fluids, *Adv. Mater.* **13** (2001) pp. 1847–1857.
4. Hao T., Electrorheological suspensions, *Adv. Colloid Interface Sci.* **97** (2002) pp. 1–35.
5. Henley S. and Filisko F.E., Flow profiles of electrorheological suspensions: An alternative model for ER activity, *J. Rheol.* **43** (1999) pp.1323–1336.
6. Filisko F. E., Henley S. and Quist G., Recent developments in the properties and composition of electrorheological fluids, *J. Intelligent Mater. Syst. Struct.* **10** (1999) pp. 476–480.
7. Wolfram, S., *The Mathematica book*, 3<sup>rd</sup> ed., Cambridge, New York: Cambridge University Press (1996)

Příloha P VI: Ponížil P., Pavlínek V., Kitano T.,  
Dřímál T.: Image analysis of electrorheological flow  
patterns. International Journal of Mathematical Mo-  
dels and Methods in Applied Sciences, 1, 4 (2007)  
239-242 ISSN: 1998-0140

# Electrorheological flow patterns analysis

P. Ponížil<sup>1</sup>, V. Pavlínek<sup>2</sup>, T. Kitano<sup>2</sup>, T. Dřímál<sup>3</sup>

<sup>1</sup> Faculty of Technology, Tomas Bata University in Zlín, TGM square 275, 762 72 Zlín, Czech Republic

<sup>2</sup> Polymer Centre, Tomas Bata University in Zlín, TGM square 275, 762 72 Zlín, Czech Republic

<sup>3</sup> Faculty of Applied Informatics, Tomas Bata University in Zlín, Nad Stráněmi 4511, 762 72 Zlín, Czech Republic

## Introduction

Electrorheological (ER) fluids are systems whose rheological properties (viscosity, yield stress, shear modulus) can be controlled by external electric field. Polarized particles are oriented in the direction of electric field and create structures (chains), which increase the rigidity of the originally liquid system.

The arrangement of the particle chains in electric field has been proved via optical microscopy. A direct optical display of ER suspensions in rotational viscometer presented in papers [1] indicated that particles organize into lamellar or ring structures, which are optimal for minimum energy dissipation, and this structure depends on the flow field. The aim of this paper is to describe ring electrorheological structures, using image analysis, with respect to conditions of their development.

## Experimental

Rotational viscometer Bohlin Gemini (Malvern Instruments, UK) in parallel plates geometry (diameter 40 mm, gap 1 mm) was used for experiments. 20 and 40 wt. % suspensions of glass beads (40  $\mu\text{m}$ ) in liquid silicone rubber were sheared at various shear rates in the presence of electric field (1 kV/mm) and generated circular flow patterns were fixed via curing of silicone rubber matrix.

Later, samples were scanned on commercial scanner and greyscale images with typical size 1000x1000 pixels were obtained.

## Results and discussion

As can be seen in figure 1, circular structures developed during the flow of electrorheological fluids upon application of electric field significantly depend on a

value of shear rate used. At low shear rates higher number of thin rings is formed. With increasing shear rate, number of rings decreases while their width rises.

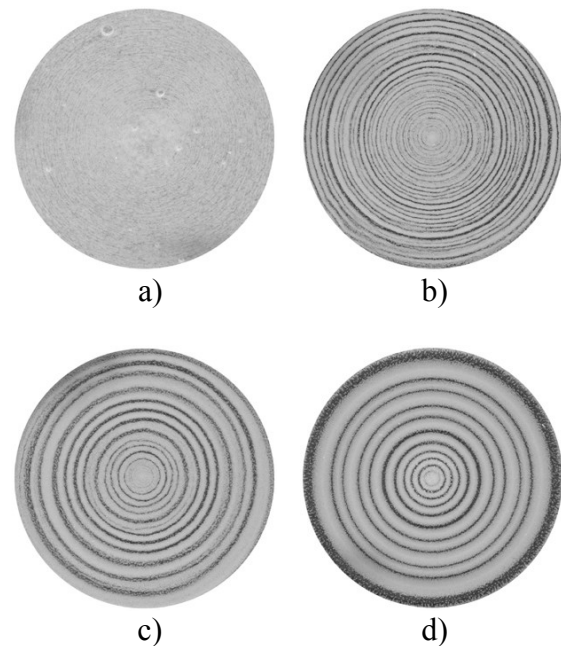


Fig. 1: ER flow patterns. 40% of glass beads (white colour) at electric field strength of 1 kV/mm and various shear rates a) 0.1  $\text{s}^{-1}$ , b) 1.0  $\text{s}^{-1}$ , c) 3.0  $\text{s}^{-1}$  and d) 30.0  $\text{s}^{-1}$ .

The goal of presented mathematical procedure is to describe electrorheological ring patterns by some simple numbers defined by radial dependence of darkness of the image and to correlate them with conditions of their preparation.

Suppose that  $p(x,y)$  is brightness of pixel in  $x$ -column and  $y$ -row of the image. There are several problems to be solved prior analysis. Samples are relatively soft and easily deformable therefore images are slightly prolonged in some directions because of manipulation.



In the first step the centre of sample symmetry was detected as a point  $(x_c, y_c)$  minimizing the function  $S(x_c, y_c)$ :

$$S(x_c, y_c) = \sum_{x,y} \frac{(p(x_c + x, y_c + y) - p(x_c - x, y_c - y))^2}{\sqrt{x^2 + y^2}}$$

where  $p(x, y)$  is brightness of  $(x, y)$  pixel. If the centre is found, image is divided into 8 radial sectors and mean brightness as a function  $f_i(r)$  (index  $i$  denotes the sector) of the real centre distance  $r$  is computed for each sector. The functions  $f_i(r)$  and  $f_j(r)$  can be a little shifted (due to inaccuracy of centre detection) and elongated due to the sample deformation.

The generalized cross-correlation function [2]

$$(f_i * f_j)(h, x) = \int f_i(r) f_j(hr + x) dr$$

is a tool for finding the best shift  $x$ , and elongation  $h$  of function  $f_j$ , maximizing the function  $(f_i * f_j)$  and obtain the maximal similarity of  $f_i$  and  $f_j$ . Then all functions for all sectors are averaged to obtain final  $f(r)$  function representing radial dependence of image brightness.

After the image preprocessing two parameters for pattern characterization were chosen; contrast and period. Contrast (amplitude of  $f(r)$  function) is defined as standard deviation

$$\sigma = \sqrt{\frac{1}{R} \int_0^R \left( f(r) - \frac{1}{R} \int_0^R f(r) dr \right)^2 dr}.$$

Contrast in our interpretation represents the relative difference between brighter and darker parts of image.

Period is the mean distance between neighbouring  $f(r)$  maxima, *ie.* mean gap width of neighbouring bright pattern circles

It seems that pattern contrast in figure 2 increases with shear rate, however only small differences with concentration of glass beads appeared. On the other hand, pattern period significantly changes with both, shear rate and concentration of particles (fig. 3).

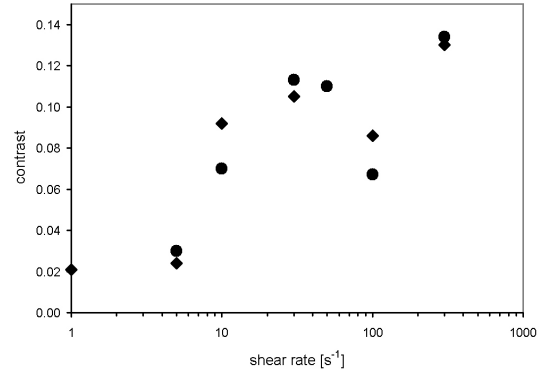


Fig. 2: Pattern contrast as a function of shear rate for various glass beads concentrations (• - 20%, ♦ - 40%).

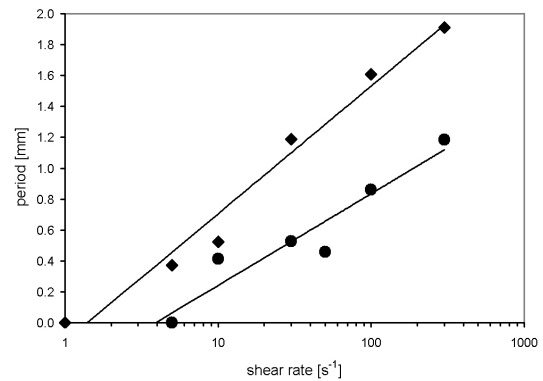


Fig. 3: Pattern period as a function of shear rate for various glass beads concentrations (• - 20%, ♦ - 40%).

## Conclusions

It was proven that our procedure of image analysis enables correlation of flow patterns in electrorheological fluids with conditions of their development.

## Acknowledgments

The authors kindly acknowledge the support by the Ministry of Education, Youth and Sport of the Czech Republic (project MSM 7088352101) and the Grant Agency of the Czech Republic (project 202/06/0419).

## References

- [1] S. Henley and F. E. Filisko, "Flow profiles of electrorheological suspensions: An alternative model for ER activity", *J. Rheol.* 43, 1999, pp. 1323–1336.
- [2] S. Wolfram, *The Mathematica book*, 3rd ed., Cambridge, New York, Cambridge University Press, 1996.