

Akademia Górniczo-Hutnicza im. Stanisława Staszica
Wydział Elektrotechniki, Automatyki, Informatyki i Elektroniki
Katedra Metrologii

Rozprawa doktorska

**ANALIZA METOD DETEKCJI
DYFRAKCYJNYCH LINII KIKUCHIEGO**

Autor: mgr inż. Rafał Frączek

Promotor: prof. dr hab. inż. Tomasz Zieliński

Kraków 2006

PODZIĘKOWANIA

Chciałbym bardzo serdecznie podziękować

*Panu dr. hab. Adamowi Morawcowi
pracownikowi Instytutu Metalurgii i Inżynierii Materiałowej PAN w Krakowie
za udostępnienie mi bazy obrazów mikroskopowych
oraz życzliwe wprowadzenie w tematykę niniejszej pracy,*

*Panu dr. inż. Zbigniewowi Mikrutowi
pracownikowi Katedry Automatyki AGH
– za pomocne uwagi dotyczące sieci neuronowych oraz transformacji Hougha,*

*Prof. dr. hab. inż. Tomaszowi Zielińskiemu
– za możliwość podpatrywania prawdziwego Wzoru uczoneści.*

Rafał Frączek

Spis treści

WPROWADZENIE – MOTYWACJA, TEZA I ZAKRES PRACY.....	5
MOTYWACJA	5
CEL I TEZA PRACY	6
ZAKRES PRACY	6
STRUKTURA PRACY	7
1. OBRAZY DYFRAKCYJNE TYPU KIKUCHIEGO W MIKROSKOPII ELEKTRONOWEJ	9
1.1. WPROWADZENIE	9
1.2. MECHANIZM POWSTAWANIA OBRAZÓW DYFRAKCYJNYCH W MIKROSKOPIE SKANINGOWYM.....	10
1.3. MECHANIZM POWSTAWANIA OBRAZÓW W MIKROSKOPIE TRANSMISYJNYM	12
2. METODY ANALIZY I PRZETWARZANIA OBRAZÓW STOSOWANE W PRACY	17
2.1. TRANSFORMACJA FOURIERA.....	17
2.2. TRANSFORMACJA RADONA.....	17
2.3. TRANSFORMACJA HOUGHA	19
2.3.1. Podstawy teoretyczne.....	19
2.3.2. Metody obliczeniowe.....	24
2.3.3. Zastosowania	24
2.4. TRANSFORMACJA FALKOWA.....	24
2.4.1. Ciągła transformacja falkowa.....	24
2.4.2. Dyskretna transformacja falkowa.....	26
2.4.3. Transformacja ridgelet.....	30
2.4.4. Transformacja curvelet	32
2.5. ODSZUMIANIE OBRAZÓW	34
2.5.1. Liniowa filtracja w dziedzinie przestrzeni.....	36
2.5.2. Nieliniowa filtracja w dziedzinie przestrzeni	36
2.5.3. Metoda pochodnych cząstkowych	37
2.5.4. Filtracja przestrzenno-częstotliwościowa.....	38
2.5.5. Filtracja w dziedzinie współczynników transformacji falkowej.....	38
2.6. FILTRACJA ZA POMOCĄ PRZESTRAJANYCH FILTRÓW KIERUNKOWYCH.....	40
2.7. PODSUMOWANIE.....	42
3. METODY KLASYFIKACJI STOSOWANE W PRACY	43
3.1. ALGORYTMY NIENADZOROWANE	43
3.1.1. Algorytm <i>k</i> -średnich	43
3.1.2. Algorytm rozmytych <i>k</i> -średnich	44
3.1.3. Algorytm Gustafsona-Kessla.....	45
3.2. ALGORYTMY NADZOROWANE.....	47
3.2.1. Algorytm <i>k</i> -najbliższych sąsiadów	47
3.2.2. Algorytm rozmytych <i>k</i> -najbliższych sąsiadów.....	47
3.2.3. Algorytm MMD	48
3.3. METODA WEKTORÓW NOŚNYCH.....	49
3.3.1. Przypadek liniowo separowalny	50
3.3.2. Przypadek nieseparowalny liniowo.....	51
3.3.3. Klasyfikacja nieliniowa	52
3.4. SIECI NEURONOWE	53
3.4.1. Matematyczny model neuronu	53
3.4.2. Warstwy sieci neuronowej.....	54
3.4.3. Wstępne i końcowe przetwarzanie danych.....	55
3.4.4. Klasyczny algorytm wstecznej propagacji błędów.....	57
3.4.5. Momentowa metoda wstecznej propagacji błędów.....	58
3.5. PODSUMOWANIE.....	59
4. ALGORYTM 1. – SKANOWANIE OBRAZU ZA POMOCĄ SPECJALNEJ MASKI.....	60
4.1. WSTĘP.....	60
4.2. STRUKTURA ALGORYTMU	60

4.3.	WSTĘPNE PRZETWARZANIE OBRAZU ŹRÓDŁOWEGO.....	60
4.3.1.	<i>Odszumianie obrazu</i>	61
4.3.2.	<i>Wyrównanie nierównomiernego tła obrazu</i>	61
4.3.3.	<i>Detekcja i usunięcie artefaktów</i>	64
4.4.	PĘTLA GŁÓWNA ALGORYTMU.....	67
4.4.1.	<i>Filtracja za pomocą maski Sobela</i>	67
4.4.2.	<i>Skanowanie obrazu za pomocą maski złożonej z trzech linii prostych</i>	67
4.4.3.	<i>Detekcja linii na podstawie analizy zmienności parametru p</i>	71
4.4.4.	<i>Wyznaczenie najbardziej prawdopodobnego położenia linii</i>	72
4.5.	WYNIKI BADAŃ.....	73
4.5.1.	<i>Wpływ wartości progu θ_p na skuteczność algorytmu</i>	73
4.5.2.	<i>Wpływ zastosowanej maski filtrującej na skuteczność algorytmu</i>	73
4.5.3.	<i>Wpływ nierównomiernego oświetlenia obrazu na skuteczność algorytmu</i>	74
4.5.4.	<i>Porównanie skuteczności opracowanego algorytmu z algorytmem referencyjnym</i>	75
4.6.	PODSUMOWANIE.....	75
5.	ALGORYTM 2. – MODYFIKACJA TRANSFORMACJI HOUGH.....	76
5.1.	WSTĘP.....	76
5.2.	STRUKTURA ALGORYTMU.....	76
5.3.	WSTĘPNE PRZETWARZANIE OBRAZU ŹRÓDŁOWEGO.....	76
5.4.	FILTRACJA KIERUNKOWA.....	77
5.5.	BINARYZACJA.....	78
5.6.	PRZETWARZANIE OBRAZU BINARNEGO.....	78
5.7.	ZMODYFIKOWANA TRANSFORMACJA HOUGH.....	79
5.8.	EKSTRAKCYJA MAKSYMÓW W PRZESTRZENI TRANSFORMACJI HOUGH.....	81
5.9.	WERYFIKACJA.....	82
5.10.	WYODREBNIENIE PAR LINII RÓWNOLEGŁYCH.....	83
5.11.	WYNIKI BADAŃ.....	83
5.12.	PODSUMOWANIE.....	84
6.	ALGORYTM 3. – ZASTOSOWANIE ZAAWANSOWANYCH METOD PRZETWARZANIA OBRAZÓW I KLASYFIKACJI.....	86
6.1.	WSTĘP.....	86
6.2.	STRUKTURA ALGORYTMU.....	86
6.3.	ESTYMACJA POZIOMU SZUMU.....	86
6.4.	ODSZUMIANIE.....	88
6.5.	ZWIĘKSZENIE KONTRASTU.....	89
6.5.1.	<i>Zwiększenie kontrastu za pomocą modyfikacji krzywej tonalnej</i>	89
6.5.2.	<i>Zwiększenie kontrastu za pomocą transformacji curvelet</i>	90
6.6.	FILTRACJA KIERUNKOWA.....	92
6.7.	BINARYZACJA.....	92
6.8.	PRZETWARZANIE OBRAZU BINARNEGO.....	92
6.9.	TRANSFORMACJA HOUGH.....	92
6.10.	DETEKCJA MAKSYMÓW W PRZESTRZENI TRANSFORMACJI HOUGH.....	93
6.10.1.	<i>Proste progowanie (ST)</i>	93
6.10.2.	<i>Analiza lokalnego kontrastu (LCA)</i>	94
6.10.3.	<i>Klasyfikacja cech (FC)</i>	96
6.10.4.	<i>Analiza profilu (PPA)</i>	96
6.11.	WERYFIKACJA.....	98
6.12.	POSZUKIWANIE BRAKUJĄCYCH LINII.....	99
6.13.	WYNIKI.....	99
6.14.	OPTIMALIZACJA METODY – DOBÓR OPCJI I WARTOŚCI PARAMETRÓW.....	101
6.14.1.	<i>Czas wykonania poszczególnych etapów algorytmu</i>	101
6.14.2.	<i>Ocena wpływu poszczególnych etapów na wyniki końcowe</i>	101
6.15.	PODSUMOWANIE.....	102
7.	WNIOSKI KOŃCOWE.....	103
8.	LITERATURA.....	105

Wprowadzenie – motywacja, teza i zakres pracy

Motywacja

Ważnym problemem inżynierii materiałowej jest możliwość powtarzalnej produkcji materiałów krystalicznych o określonych właściwościach fizycznych. Właściwości te są w znacznej mierze zdeterminowane wielkością i orientacją domen krystalicznych, z których zbudowany jest dany materiał. Szczególną rolę w badaniu tych właściwości odgrywa tzw. tekstura krystalograficzna będąca topografią przestrzennego rozkładu orientacji poszczególnych ziaren. Pełne wyznaczenie tekstury wymaga określenia orientacji wielu tysięcy ziaren, dlatego ważnym problemem jest możliwość jej automatycznego wyznaczenia. Stosując technikę OM (*Orientation Mapping*), wnioskuje się o wielkości i orientacji domen na podstawie obrazów dyfrakcyjnych uzyskanych z mikroskopii elektronowej, np. typu Kikuchiego. Obrazy te zawierają tzw. linie Kikuchiego. Ich układ pozwala na określenie dokładnej orientacji każdego z analizowanych mikro- lub nano-obszarów. Na podstawie tych wyników można wnioskować w inżynierii materiałowej o właściwościach badanych materiałów krystalicznych i opracowuje technologie ich produkcji. Wiarygodność oraz precyzja uzyskanych map orientacji mikro- lub nano-obszarów jest ściśle powiązana z liczbą poprawnie znalezionych par linii na obrazie. Im większa jest liczba tych par, tym wiarygodność ta jest większa [Lass98]. Rozwój techniki OM został zapoczątkowany w skaningowej mikroskopii elektronowej (OM/SEM), w której uzyskuje się obrazy za pomocą dyfrakcji elektronów wstecznie rozproszonych. Metoda OM jest także rozwijana w transmisyjnej mikroskopii elektronowej (OM/TEM). Zastosowanie mikroskopii transmisyjnej pozwala na znaczne zwiększenie przestrzennej rozdzielczości metody, dzięki czemu możliwe jest badanie materiałów o ultradrobny ziarne, gdzie konieczne jest uzyskiwanie informacji z obszarów o wielkości rzędu nanometrów. Jednym z najczęściej stosowanych narzędzi obliczeniowych służących do detekcji linii prostych na obrazach jest transformacja Hougha [Mali02]. W przeszłości opracowano już odpowiednie algorytmy adaptujące ją do wyznaczenia linii Kikuchiego [Lass98, Mora02] na obrazach uzyskiwanych metodą OM/SEM. Do tej pory jednak nie opracowano komercyjnych rozwiązań odpowiednich dla metody OM/TEM. Stosowane w tym przypadku niekomercyjne aplikacje nie przynoszą satysfakcjonujących rezultatów [Sztw06a]. Przyczyną jest to, iż bezpośrednio stosowanie transformacji Hougha do automatycznej detekcji linii Kikuchiego nie przynosi zadowalających wyników z uwagi na występujące trudności z przetwarzaniem tego typu obrazów.

Cel i teza pracy

Celem pracy jest opracowanie nowych, lepszych niż obecnie stosowane, metod detekcji dyfrakcyjnych linii Kikuchiego na obrazach cyfrowych pochodzących z transmisyjnego mikroskopu elektronowego. Aby taka detekcja mogła być skuteczna i wiarygodna, algorytm musi uwzględniać występujące na obrazach znaczne zróżnicowanie takich właściwości linii Kikuchiego jak szerokość, długość, kontrast z otoczeniem oraz jaskrawość. Z uwagi na konieczność przetwarzania wielu tysięcy obrazów, opracowany algorytm musi być również zoptymalizowany pod kątem szybkości działania. W celu umożliwienia przetwarzania wielu obrazów charakteryzujących się znacznym zróżnicowaniem globalnych właściwości, takich jak np. stopień zaszumienia czy nierównomierne oświetlenie tła, konieczne jest opracowanie oraz dobór odpowiednich metod wstępnego przetwarzania obrazu źródłowego. Ponieważ zastosowanie niektórych transformacji wymaga uprzedniej binaryzacji obrazu monochromatycznego, konieczny jest dobór odpowiednich algorytmów i procedur progowania obrazu monochromatycznego. Opracowywane metody powinny wykorzystywać nowoczesne narzędzia przetwarzania i analizy obrazów, takie jak zmodyfikowana transformacja Hougha, transformacja falkowa oraz sieci neuronowe.

W niniejszej rozprawie doktorskiej postawiono i udowodniono następującą tezę:

Wykorzystując nowoczesne techniki przetwarzania obrazów cyfrowych możliwe jest opracowanie skutecznej i wiarygodnej metody detekcji dyfrakcyjnych linii Kikuchiego w dyfrakcyjnych obrazach.

W rozprawie zaproponowano odpowiednie algorytmy oraz dokładnie je przebadano. Uzyskane bardzo dobre wyniki detekcji pozwalają stwierdzić, że powyższa teza jest prawdziwa.

Zakres pracy

Zakres pracy obejmuje realizację następujących zadań:

1. dobór oraz optymalizacja metod i parametrów wstępnego przetwarzania (uzdatniania) obrazu mikroskopowego, w tym: odszumienia, korekcji nierównomiernego oświetlenia obrazu, progowania i binaryzacji;
2. dobór oraz optymalizacja metod i parametrów ekstrakcji cech linii Kikuchiego za pomocą różnych odmian transformacji Hougha;
3. dobór i optymalizacja metod klasyfikacji (detekcji linii), np. metody k -najbliższych sąsiadów i sieci neuronowych;
4. optymalizacja końcowego algorytmu pod względem szybkości działania.

Struktura pracy

W pracy przedstawiono kolejno opracowane przez autora algorytmy służące do detekcji linii Kikuchiego na obrazach mikroskopowych. To podejście rzutu na strukturę pracy, która składa się z trzech rozdziałów przedstawiających poruszaną tematykę oraz trzech prezentujących opracowane metody.

W rozdziale pierwszym omówiono stosowaną w mikroskopii elektronowej technikę badania struktury materiałów krystalicznych przy wykorzystaniu obrazów dyfrakcyjnych. Szczególny akcent położono na mechanizm powstawania dyfrakcyjnych linii Kikuchiego w mikroskopii elektronowej. Przedstawione zostały również najczęściej spotykane w nauce i technice zastosowania metody elektronów wstecznie rozproszonych.

W rozdziale drugim przedstawiono zastosowane w pracy metody analizy i przetwarzania obrazów. W kolejności zostały omówione następujące transformacje obrazu: Radona, Hougha oraz falkowa. W dalszej kolejności przedstawiono zagadnienie odszumiania obrazów oraz szczegółowo omówiono zastosowane w pracy różne metody wykorzystywane w tym celu. Rozdział kończy się prezentacją zasady działania filtrów kierunkowych.

Rozdział trzeci zawiera opis zastosowanych w pracy metod klasyfikacji danych. Omówiono w kolejności takie metody klasyfikacji jak: algorytm k -średnich, algorytm rozmytych k -średnich, algorytm Gustafsona-Kessla, algorytm MMD (*Mean Minimum Distance*). Oprócz tych metod, opisano nowoczesną metodę klasyfikacji za pomocą tzw. wektorów nośnych SVM (*Support Vector Machines*). Jako ostatnia została zaprezentowana metoda klasyfikacji za pomocą sztucznych sieci neuronowych. Przedstawiono model sztucznego neuronu, strukturę typowej sieci neuronowej, jak również klasyczny algorytm uczenia sieci za pomocą wstecznej propagacji błędów.

W rozdziale czwartym został przedstawiony pierwszy z opracowanych algorytmów. Szczegółowo omówiona została istota jego działania. Przedstawiono wyniki badań skuteczności działania algorytmu oraz pokazano wpływ jego poszczególnych elementów składowych na wyniki końcowe.

Rozdział piąty zawiera opis algorytmu bazującego na zaproponowanej modyfikacji standardowej transformacji Hougha. Przedstawiono kolejne etapy pracy tego algorytmu oraz istotę wprowadzonych zmian. Rozdział kończy się prezentacją wyników badań dotyczących skuteczności jego działania.

Rozdział szósty zawiera opis ulepszanego algorytmu detekcji linii Kikuchiego. Przedstawiono zastosowane ulepszenia, obejmujące, po pierwsze, etap wstępnego przetwarzania obrazu oraz, po drugie, etap końcowego przetwarzania i weryfikacji otrzymanych wyników. Dodatkowo przedstawiono wyniki analizy złożoności obliczeniowej opracowanej metody oraz wyniki łącznego porównania wszystkich opracowanych algorytmów.

Pracę zamykają wnioski końcowe i wykaz cytowanej literatury.

Niniejsza rozprawa jest wynikiem końcowym realizacji grantu promotorskiego nr 3 T10C 037 28 (dyscyplina: miernictwo interdyscyplinarne) pod tytułem: „**Metody detekcji dyfrakcyjnych linii Kikuchiego na obrazach z mikroskopii elektronowej**”, który był finansowany w okresie 10.05.2005 – 9.02.2007 przez Komitet Badań Naukowych oraz realizowany przez doktoranta w Katedrze Metrologii Wydziału Elektrotechniki, Automatyki, Informatyki i Elektroniki Akademii Górniczo-Hutniczej w Krakowie.

Praca powstała w wyniku współpracy KM WEAlE-AGH oraz Instytutu Metalurgii i Inżynierii Materiałowej Polskiej Akademii Nauk w Krakowie. Doktorant otrzymał z IMIM-PAN testowe obrazy mikroskopowe, które zostały zarejestrowane przez dr Emmanuel Bouzy

(LETAM, Universite Paul Verlaine, Metz), oraz program referencyjny służący do detekcji linii Kikuchiego. Rejestracji obrazów dokonano przy użyciu mikroskopu transmisyjnego Philips CM 200 pracującego przy napięciu 200kV. Problem detekcji linii Kikuchiego na obrazach oraz warunki eksperymentu zostały doktorantowi przedstawione przez Pana doktora habilitowanego Adama Morawca, pracownika IMIM-PAN w Krakowie.

1. Obrazy dyfrakcyjne typu Kikuchiego w mikroskopii elektronowej

1.1. Wprowadzenie

Badając właściwości ciał stałych, stwierdzamy, że tylko nieliczne z nich posiadają strukturę monokrystaliczną. W otaczającym nas świecie z trudem znajdujemy monokryształy. Jednak, przy odrobinie szczęścia, możemy je spotkać w przyrodzie. Najczęściej będą to (mono) kryształy soli kuchennej, np. pochodzące z kopalni soli w Wieliczce. Znacznie rzadziej spotkamy monokryształ w postaci diamentu w biżuterii. Możemy także spotkać monokryształy w laboratoriach badawczych. Jednak jest faktem, że monokryształy są naprawdę rzadkością w naszym otoczeniu.

Z kolei, zdecydowana większość ciał metalicznych posiada strukturę polikrystaliczną. Praktycznie każda metaliczna część lodówki, roweru czy pralki jest polikryształem. Również wiele innych materiałów, takich np. jak skały, ceramiki czy też część polimerów, posiada budowę polikrystaliczną. Polikryształ jest konglomeratem wielkiej ilości małych monokryształów zwanych krystalitami lub ziarnami (patrz rys. 1.1).



Rys. 1.1. Struktura polikryształu [Joun1]

Jeśli przyjmiemy, że typowy rozmiar ziarna jest rzędu 10 nm, a próbka ma rozmiar rzędu 1 cm, to polikryształ będzie zawierał miliard ziaren. Obszar przejściowy pomiędzy ziarnami to granica międzyziarnowa, zwana również granicą ziaren. Atomy znajdujące się w obszarze granicy międzyziarnowej charakteryzują się znacznie mniejszym stopniem uporządkowania przestrzennego. Z kolei atomy w obszarze krystalitu charakteryzują się wysokim stopniem uporządkowania.

Sieć krystalograficzna (przestrenny układ atomów w obrębie krystalitu) każdego ziarna ma określoną orientację względem przyjętego układu odniesienia. W przestrzennej orientacji sieci krystalograficznych występuje na ogół znaczne zróżnicowanie pomiędzy kolejnymi krystalitami. Gdyby jednak rozkład orientacji ziaren był zupełnie przypadkowy, wtedy powiedzielibyśmy, że materiał nie posiada tekstury krystalograficznej. W praktyce orientacje przestrzenne ziaren nie są zupełnie przypadkowe, ale grupują się wokół pewnych charakterystycznych orientacji krystalograficznych, zwanych składowymi tekstury. Innymi słowy, nieprzypadkowy rozkład orientacji ziaren to tekstura krystalograficzna [Rand92]. Ma ona ogromne znaczenie dla oceny właściwości materiału polikrystalicznego. Informuje nas o tym, jakie orientacje sieci krystalograficznych dominują w próbce. Informacja ta ma kluczowe znaczenie dla badania oraz przewidywania makroskopowych właściwości próbki materiału [Schw00, Hump99].

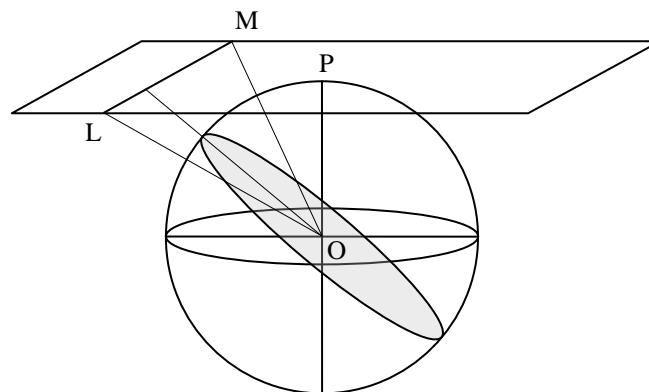
Współcześnie teksturę wyznacza się między innymi za pomocą metody OM opartej na dyfrakcji elektronów. Istnieją zasadniczo dwie odmiany tej metody. Pierwszą z nich jest mikroskopowe obrazowanie orientacji ziaren za pomocą dyfrakcji elektronów wstecznie rozproszonych EBSD (*Electron Back Scatter Diffraction*) w mikroskopii skaningowej [Schw00]. Druga wykorzystuje dyfrakcję elektronów w mikroskopii transmisyjnej TEM (*Transmission Electron Microscopy*) [Schw98]. Dzięki metodzie OM możliwe jest między innymi bezpośrednio wyznaczenie pełnej orientacji (trzy kąty) poszczególnych ziaren, ilościowe badanie mikrotekstury.

1.2. Mechanizm powstawania obrazów dyfrakcyjnych w mikroskopie skaningowym

Metoda ta została pierwotnie opracowana w 1954 roku [Alam54, Oie1], jednak dopiero w latach siedemdziesiątych XX wieku została po raz pierwszy zastosowana w praktyce do badania tekstury [Vena73]. Istota tej metody polega na określaniu przestrzennej orientacji poszczególnych ziaren na podstawie analizy położenia na obrazie dyfrakcyjnych linii powstających w wyniku rozpraszania wiązki elektronów na sieci krystalicznej ziarna [Wrig91].

1.2.1. Powstawanie obrazów dyfrakcyjnych [Oie1]

Na początku rozważmy sferę o środku O i punkcie P na jej powierzchni (rys. 1.4). Załóżmy, że płaszczyzna rzutowania jest styczna do sfery w punkcie N . Odcinek OP przecina płaszczyznę rzutowania w punkcie p , który jest obrazem punktu P w rzucie ortogonalnym. Z kolei odcinek LM jest obrazem płaszczyzny prostopadłej do odcinka OP .



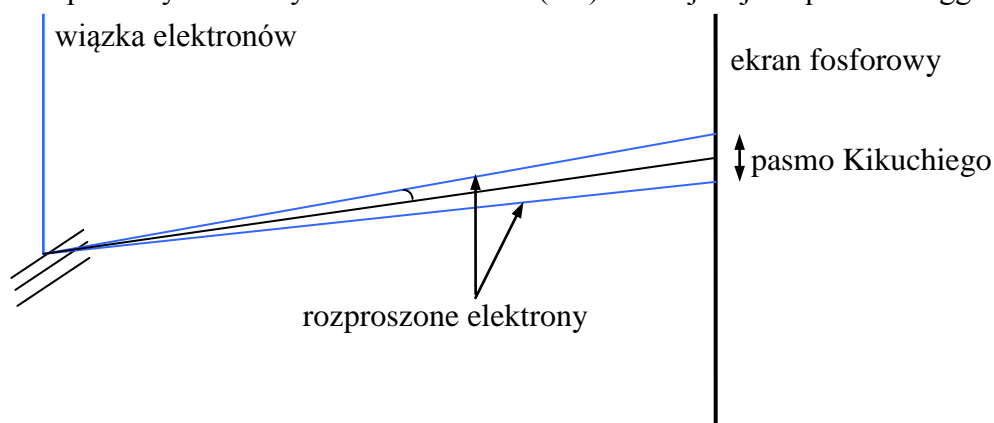
Rys. 1.2. Rzut ortogonalny [Oie1]

Założmy dalej, że wypolerowana próbka badanego materiału jest umieszczona wewnątrz mikroskopu elektronowego pod stosunkowo dużym kątem (zwykle 70°) względem wiązki elektronów (patrz rys. 1.3). Następnie wiązka elektronów jest kierowana na badany punkt próbki. Atomy materiału rozpraszają padające na próbkę elektrony. Rozproszona wiązka jest rzutowana na płaszczyzny krystalitu we wszystkich kierunkach. Zgodnie z dualną, falowo-korpuskularną naturą materii, rozproszone elektrony mogą być traktowane jako fala o określonej długości, która ulega zjawisku dyfrakcji oraz interferencji. Długość fali odpowiadającej elektronowi zależy od jego energii, która z kolei zależy od wartości napięcia przyspieszającego elektron. Przy typowej wartości napięcia rzędu 20 kV elektronowi odpowiada fala o długości rzędu 0,0062 nm ($\lambda = h/\sqrt{2m_0eE}$). Rozproszone elektrony padają na płaszczyzny krystalitu (patrz rys. 1.4).

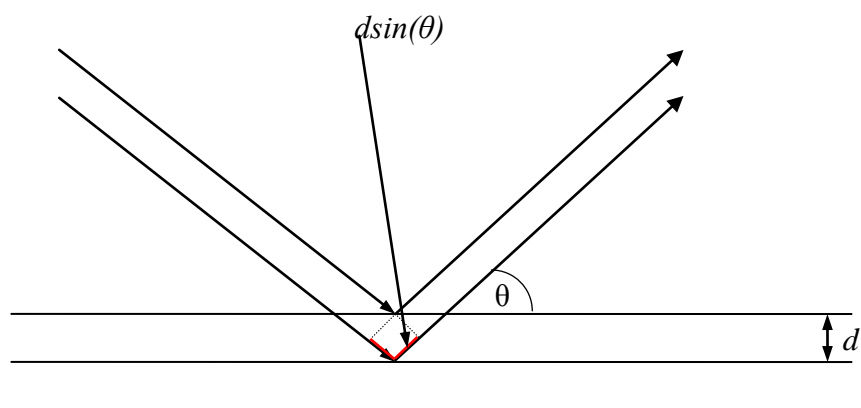
Różnica długości dróg dla elektronów padających na sąsiednie płaszczyzny wynosi $2d\sin\theta$. Jeśli różnica tych długości jest równa całkowitej wielokrotności długości fali elektronu, to następuje wzmocnienie fal tych elektronów. Matematycznie opisuje to następująca zależność:

$$n\lambda = 2d \sin(\theta) \quad (1.1)$$

gdzie n jest liczbą całkowitą, λ jest długością fali elektronu, zaś d jest odległością między sąsiednimi płaszczyznami krystalitu. Równanie (1.1) znane jest jako prawo Bragga.

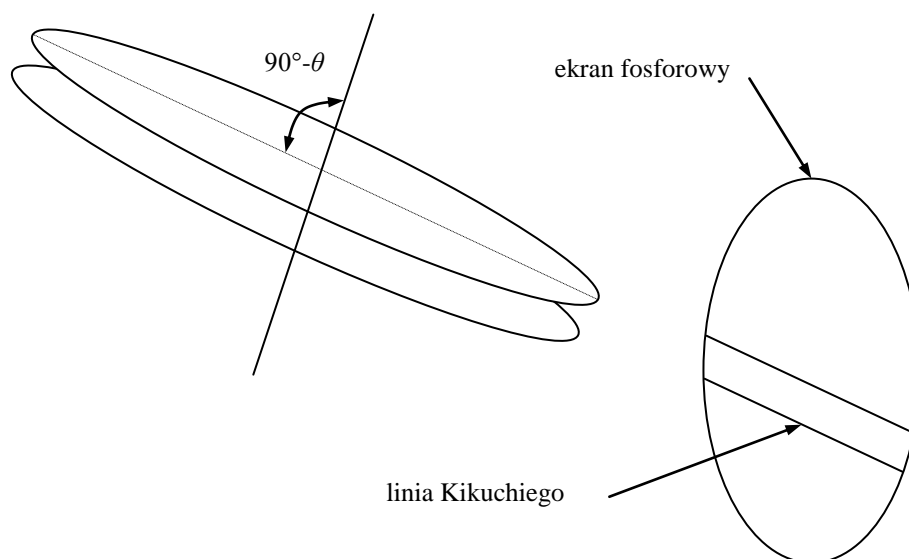


Rys. 1.3. Powstawanie obrazów dyfrakcyjnych [Oie1]



Rys. 1.4. Rozproszenie wiązki elektronów na płaszczyznach krystalitu [Oie1]

W przypadku, gdy różnica dróg nie jest całkowitą wielokrotnością długości fali elektronu, następuje mniejsze lub większe tłumienie fal tych elektronów. W rezultacie rozproszona fala elektronów przyjmuje w przestrzeni kształt dwóch stożków złączonych wierzchołkami.



Rys. 1.5. Stożki dyfrakcyjne [Oie1]

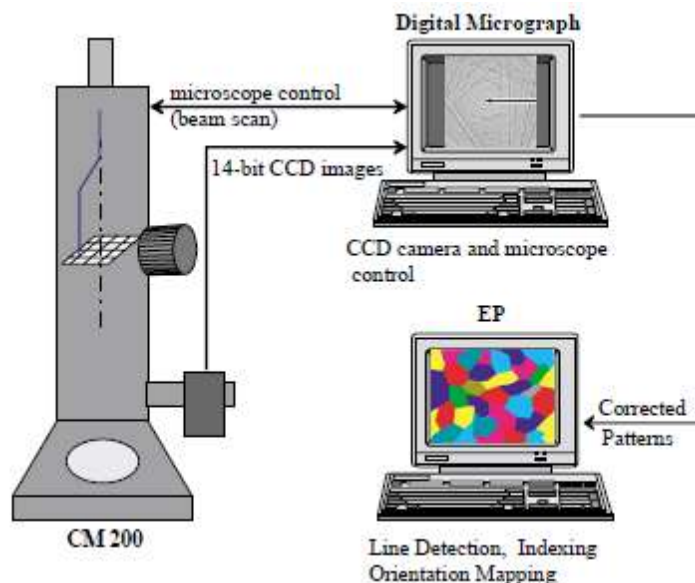
Kąt rozwarcia tych stożków jest stosunkowo duży – dla typowej długości fali elektronu rzędu 0.0062 nm i odległości między kolejnymi płaszczyznami rzędu 0.233 nm (aluminium) wynosi on 89°. Ślady przecięcia podstawy stożka z płaszczyzną ekranu mogą być obserwowane za pomocą fosforowego ekranu przymocowanego do czułej kamery. Jest ona zwykle skierowana poziomo, dzięki czemu ekran może znajdować się blisko próbki. Umożliwia to obserwację stożków w szerokim zakresie kątów rozproszenia. Geometria zachodzącego zjawiska może być interpretowana za pomocą rzutu ortogonalnego. W tym przypadku środek sfery O jest punktem, w którym następuje dyfrakcja elektronów, fosforowy ekran jest płaszczyzną rzutowania, a punkt styczności N jest środkiem uzyskiwanego obrazu z kamery. Na obrazach uzyskiwanych z kamery stożki elektronów jawią się jako proste linie. Od nazwiska swego odkrywcy linie te zwane są liniami Kikuchiego. Obszar pomiędzy dwoma równoległymi liniami zwany jest pasmem Kikuchiego. Trzeba zaznaczyć, że ściśle rzecz biorąc, obrazem stożka w rzucie ortogonalnym na płaszczyznę ekranu jest linia krzywa, jednak na obrazie dyfrakcyjnym na ogół uzyskuje się linie proste, bowiem kąt rozwarcia stożka jest bliski 90°, a ekran znajduje się daleko od próbki. Im dalej jednak od środka obrazu tym bardziej hiperboliczny kształt może być zauważony.

W celu wyznaczenia tekstury danej próbki przesuwana jest wiązka elektronów wzdłuż jej powierzchni. Ponieważ dla każdego kolejnego położenia wiązki elektronów względem powierzchni próbki następuje rejestracja obrazu, dlatego w celu wyznaczenia tekstury próbki (rys. 1.9) konieczna jest analiza wielu takich obrazów.

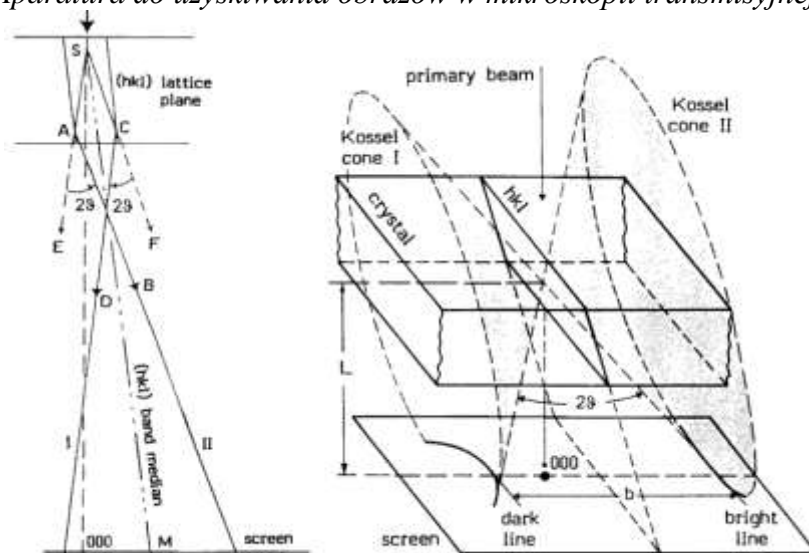
Metoda EBSD pozwala osiągnąć rozdzielczość przestrzenną rzędu 30 nm, zaś kątową rzędu 0,5°.

1.3. Mechanizm powstawania obrazów w mikroskopie transmisyjnym

W przypadku mikroskopii transmisyjnej TEM uzyskuje się obrazy dyfrakcyjne Kikuchiego TKP (*Transmission Kikuchi Diffraction Pattern*) lub obrazy dyfrakcyjne powstałe w wyniku zastosowania zbieżnej wiązki elektronów CBED (*Convergent Beam Elektron Diffraction Pattern*) [Sztw06a].



Rys. 1.6. Aparatura do uzyskiwania obrazów w mikroskopii transmisyjnej [Fund03]



Rys. 1.7. Proces powstawania obrazów TKP [Schw98]

Wiązka elektronów pada na cienką warstwę badanego materiału. Wiązka ta ma kształt stożka o regulowanym kącie rozwarcia 2α . Jeśli kąt α jest mniejszy od kąta θ_b (θ_b jest kątem Bragga), uzyskuje się obrazy TKP, zaś obrazy CBED powstają w przypadku, gdy kąt α jest większy od kąta θ_b [Fund03]. Przenikające przez cienką warstwę próbki elektrony ulegają rozproszeniu a następnie dyfrakcji. W efekcie fala elektronów przyjmuje w przestrzeni kształt dwóch stożków złączonych wierzchołkami (patrz rys. 1.7). Gdy fala elektronów osiąga ekran, krawędzie podstawy stożków tworzą na ekranie charakterystyczne linie: linie Kikuchiego (patrz rys 1.8).

W mikroskopii transmisyjnej metoda OM pozwala na uzyskanie rozdzielczości przestrzennej map orientacji tekstury rzędu 10 nm oraz kątowej rzędu $0,5^\circ$. Jest to jedyny sposób pozwalający na badanie materiałów nanokrystalicznych oraz silnie odkształconych.

Ponieważ geometria linii CBED jest taka sama jak linii TKP, dlatego w dalszej części pracy w odniesieniu do dyfrakcyjnych linii CBED także będzie stosowane określenie: linie Kikuchiego.

1.3.1. Trudności związane z przetwarzaniem obrazów mikroskopowych

Powolny rozwój technik w mikroskopii transmisyjnej wynika między innymi z dużych trudności związanych ze skuteczną i wiarygodną detekcją linii Kikuchiego w obrazach TKP i CBED. Najważniejsze z nich to:

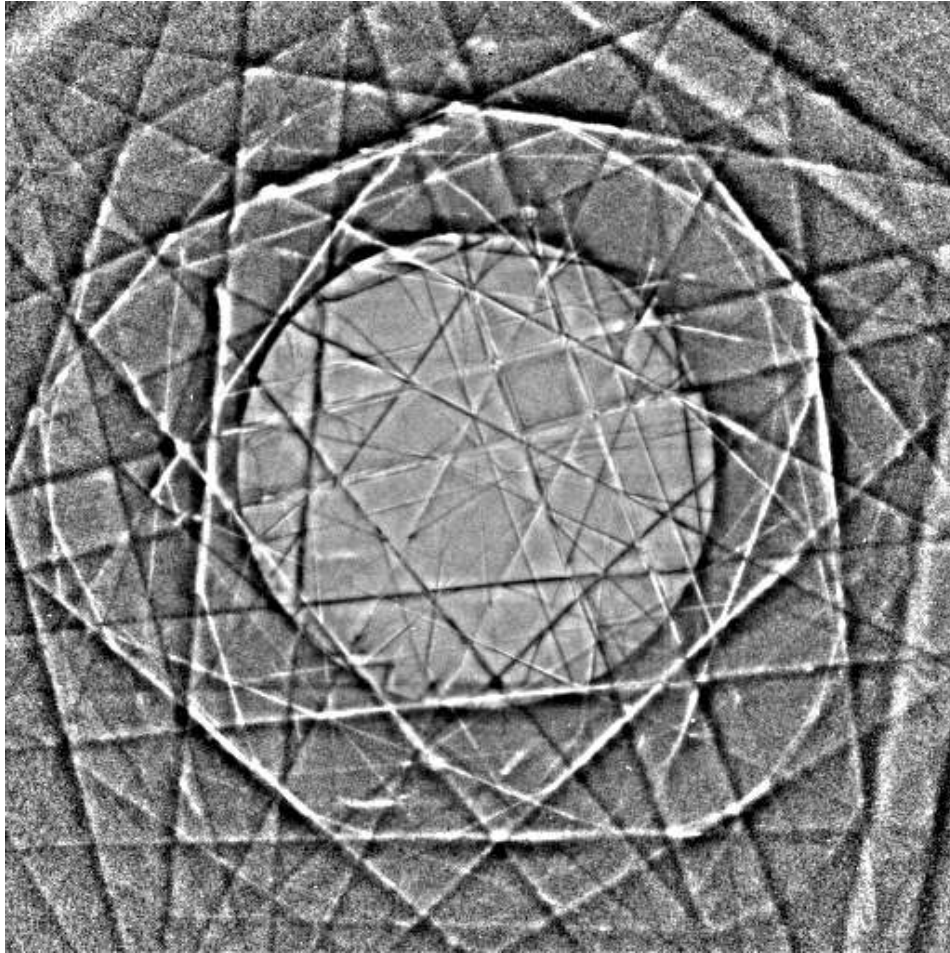
- a) znacząca nierównomierność tła niektórych obrazu;
- b) występowanie artefaktów, np. w postaci przesłony;
- c) niektóre linie charakteryzują się niewielkim kontrastem w stosunku do otoczenia;
- d) niektóre linie są w różny sposób widoczne na obrazie – linia może być jasna lub ciemna, linia może być w jednym miejscu jasna, a w drugim ciemna. Podobnie, może być jasna po jednej stronie, a ciemna po drugiej;
- e) obrazy mogą być znacząco zaszumione;
- d) niektóre linie charakteryzują się znaczącym rozmyciem (linia jest stosunkowo szeroka).

Opracowane w dalszej części pracy algorytmy detekcji linii Kikuchiego starają się minimalizować wpływ powyższych cech na jakość uzyskiwanych wyników.

1.3.2. Zastosowania metody OM

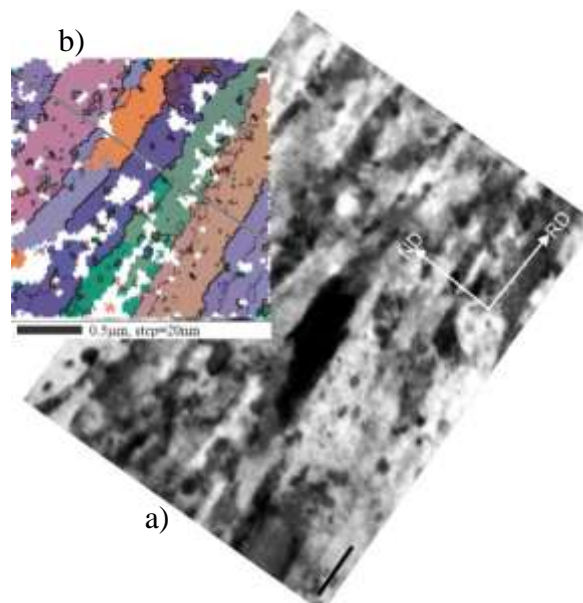
Metoda OM jest stosowana w następujących obszarach nauki i techniki:

- analiza tekstur w przemyśle stalowym i aluminiowym w związku z kontrolą jakości [Dixo06, Driv96, Hurl03];
- badanie tekstur w przemyśle stalowym w celu ulepszenia obróbki powierzchni [Archi04, Hunt02];
- badanie tekstur w związku z właściwościami elektrycznymi i magnetycznymi [Vand02];
- badanie wpływu granic międzyziarnowych na korozję oraz badanie defektów sieci krystalicznej w metalurgii i przemyśle jądrowym [Schu04];
- jako metoda uzupełniająca dla tradycyjnych technik, takich jak dyfrakcja promieni rentgenowskich [Bung00];
- pomiar rozkładu naprężeń w materiałach [Otho02, Brew02];
- badanie wpływu granic międzyziarnowych na rodzaj i charakter pęknięć materiału [Chen94];
- badanie wpływu tekstury na właściwości wysokotemperaturowych nadprzewodników [Drie05];
- badanie struktury materiałów w geologii [Prio99, Piaz05, Prio99];
- mikroskopowe badania tekstury, a w szczególności związku między mikroteksturą a mikrostrukturą [Piaz04];
- jako metoda uzupełniająca dyfrakcję promieni rentgenowskich w badaniu tekstury w skali makroskopowej [Bung00];
- badanie rekrytalizacji w metalach, stopach oraz skałach [Trim00, Heid00, Piaz04];
- badanie cienkich warstw, a w szczególności: tranzystorów cienkowarstwowych, pamięci nieulotnych oraz diod laserowych [Trag02, Naka97];
- identyfikacja faz substancji [Cabu04];
- analiza i przewidywanie właściwości metali, stopów, materiałów ceramicznych, półprzewodników, nadprzewodników [Gey02];



Rys. 1.8. Przykładowy obraz z mikroskopu transmisyjnego

Na rysunku (1.9) przedstawiono, uzyskany za pomocą TEM, przykładową strukturę stopu aluminium b0013 po walcowaniu na zimno do 71 %, obszar osnowy, przekrój podłużny.



Rys. 1.9. Przykładowa struktura materiału i jego mikrotekstura, a) obszar w jasnym polu, b) topografia orientacji (obraz zamieszczony dzięki uprzejmości dr. hab. Krzysztofa Sztwiertni z IMIPN)

- pomiar i analiza takich właściwości jak wielkość ziaren i lokalna tekstura [Rand00, Dark03, Hump01];
- pomiar naprężeń i analiza pęknięć [Floe02];
- badanie wpływu wysokiej temperatury na cienkie warstwy substancji [Mirp04];
- badanie wpływu błędnych orientacji krystalograficznych na proces korozji [Yuan03];
- badanie wpływu mikrostruktury na właściwości zmęczeniowe stopów [Gopi97];
- badanie deformacji plastycznych skał [Basc02].
- Wpływ granicy międzyfazowej na właściwości materiałów polikrystalicznych [Sztw06b]

2. Metody analizy i przetwarzania obrazów stosowane w pracy

W niniejszym rozdziale przedstawiono podstawy matematyczne metod przetwarzania obrazów, stosowanych w części eksperymentalnej pracy. W szczególności skrótowo opisano:

- transformacje obrazów: Fouriera, Radona, Hougha, falkową (*wavelet*), *ridgelet* oraz *curvelet*,
- przestrajane filtry kierunkowe,
- różne metody odszumiania obrazów.

2.1. Transformacja Fouriera

Dwuwymiarowa transformacja Fouriera jest zdefiniowana następującym równaniem [Ziel05]:

$$X_{DFT}(k, l) = \sum_{m=0}^{M-1} \left(\sum_{n=0}^{N-1} x(m, n) \cdot e^{-j \frac{2\pi}{N} nl} \right) e^{-j \frac{2\pi}{M} mk} \quad (2.1)$$

zaś transformacja odwrotna ma postać:

$$x(m, n) = \frac{1}{N} \sum_{l=0}^{N-1} \left(\frac{1}{M} \cdot \sum_{k=0}^{M-1} X_{DFT}(k, l) e^{j \frac{2\pi}{M} mk} \right) e^{j \frac{2\pi}{N} nl} \quad (2.2)$$

gdzie m, n są indeksami dyskretnego położenia przestrzennego, natomiast k, l są indeksami związanych z nimi dyskretnych częstotliwości ($0 \leq m, k \leq M-1$; $0 \leq n, l \leq N-1$). Transformacje te mogą być interpretowane jako sekwencja jednowymiarowych transformacji Fouriera (prostej lub odwrotnej) wierszy macierzy, po której następuje sekwencja transformacji kolumn nowej macierzy, będącej wynikiem pierwszej operacji. Dla jednowymiarowej DFT opracowano szybkie algorytmy obliczeniowe [Szab90].

2.2. Transformacja Radona

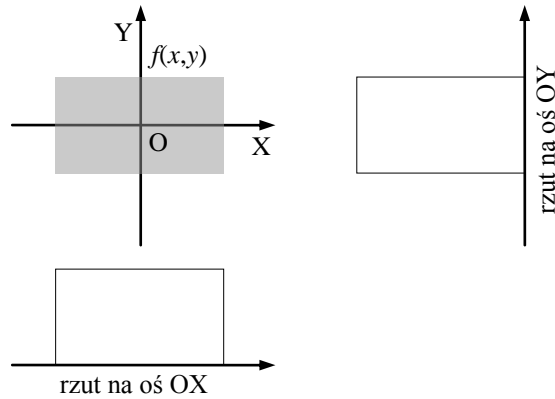
Transformacja Radona obrazu $f(x, y)$ jest zdefiniowana następująco [Wiki1]:

$$\mathbf{R}f(\theta, t) = \int f(x, y) \delta(x \cdot \cos(\theta) + y \cdot \sin(\theta) - t) dx dy \quad (2.3)$$

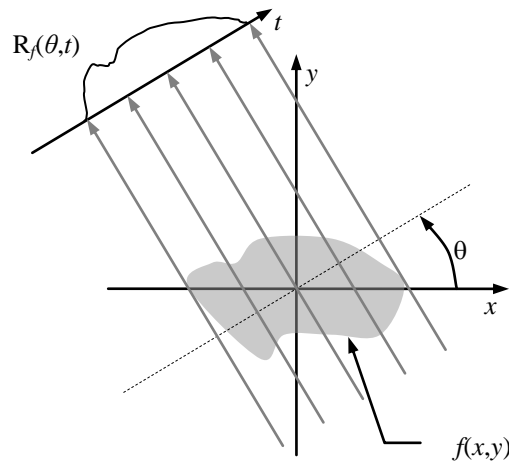
gdzie δ jest dystrybucją Diraca. Powyższe równanie mówi, że transformacja ta jest całką obrazu wzdłuż linii prostej danej równaniem:

$$x \cdot \cos(\theta) + y \cdot \sin(\theta) - t = 0 \quad (2.4)$$

Prosta ta jest nachylona pod kątem θ do osi OY, a jej odległość od początku układu współrzędnych wynosi t . Jeśli wyznaczymy transformację Radona przy ustalonym kącie θ dla $t \in (-t_{min}; t_{max})$, gdzie t_{min} oraz t_{max} są tak dobrane, by pokryć cały obszar obrazu, to otrzymamy projekcję obrazu w kierunku θ .



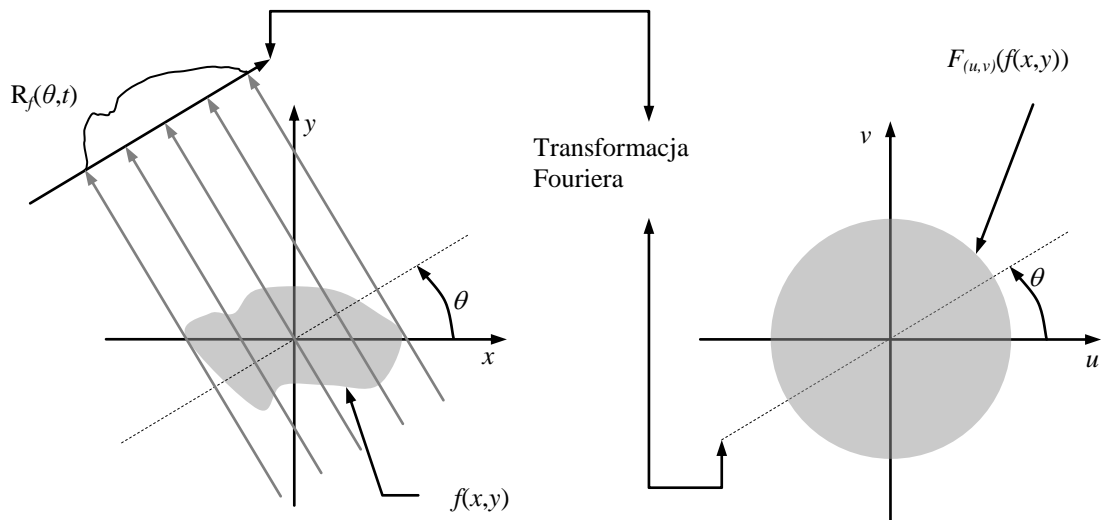
Rys. 2.1. Projekcja obrazu na osie OX oraz OY w transformacji Radona [Math1]



Rys. 2.2. Transformacja Radona obiektu o nieregularnym kształcie

Transformacja Radona jest ściśle powiązana z transformacją Fouriera [Toft96]. Związek ten wynika z właściwości transformacji Fouriera znanej jako *Fourier Slice Theorem*, a mówiącej, że transformacja Fouriera rzutu obrazu na oś pionową jest równa poziomemu profilowi dwuwymiarowej transformacji Fouriera tego obrazu. Własność ta może być uogólniona na dowolny kierunek rzutowania, co przekłada się na inną cechę tej transformacji, wedle której jeżeli obraz $f(x, y)$ zostanie obrócony o pewien kąt θ względem osi OX to transformacja Fouriera tego obrazu zostanie odpowiednio obrócona o ten sam kąt. Innymi słowy, transformacja Fouriera rzutu obrazu na prostą nachyloną pod kątem $\theta+90^\circ$ do osi OX przedstawia transformację Fouriera obrazu wzdłuż osi nachylonej pod kątem θ do osi radialnej (rys. 2. 2). Własność ta pozwala na szybkie wyznaczenie transformacji Radona za pomocą transformacji Fouriera. W tym celu dla ustalonego kąta θ wyznacza się odwrotną

transformację Fouriera z odpowiedniego profilu 2D DFT obrazu (patrz rys. 2.3). Pozwala to na zmniejszenie złożoności obliczeniowej transformacji Radona z $O(N^2)$ do $O(N\log(N))$.



Rys. 2.3. *Fourier Slice Theorem*

2.3. Transformacja Hougha

2.3.1. Podstawy teoretyczne

Transformacja Hougha jest techniką, która umożliwia wyodrębnienie z obrazu obiektów o określonym kształcie [Leav93]. Została ona opracowana przez Paula Hougha w 1962 roku [Houg62, Gonz93]. Ponieważ standardowa transformacja Hougha wymaga opisanie poszukiwanych obiektów za pomocą zbioru parametrów, to jest używana do detekcji takich obiektów jak: linie proste [Riss89], okręgi [Kimm75, Yip92] czy elipsy [Ball81, Yuen89]. Transformacja ta działa na obrazach binarnych, tj. takich, w których piksel posiada tylko dwie wartości: 1 albo 0. Chociaż istnieje uogólniona transformacja Hougha pozwalająca na detekcję obiektów o dowolnym kształcie, to z uwagi na fakt, że jest ona bardzo złożona obliczeniowo, jak również na fakt iż w niniejszej pracy poszukiwane są linie proste a nie inne obiekty, nie jest ona stosowna. Najważniejszą zaletą tej transformacji jest to, iż jest ona stosunkowo odporna na ewentualne zakłócenia, które mogą wystąpić na obrazie – szum oraz przerwy w poszukiwanych obiektach. Ponieważ transformacja Hougha stanowi kluczowy element praktycznie każdej metody do detekcji linii prostych, poniżej zostanie przedstawiony jej szczegółowy opis.

Istota transformacji Hougha polega na tym, iż każda kolejna wartość wejściowa (punkt na płaszczyźnie) dodaje swój wkład do globalnego rozwiązania (linia na obrazie, której częścią jest dany punkt). Rozważmy pojedynczy punkt (x, y) na płaszczyźnie. Przez ten punkt przechodzi nieskończenie wiele linii prostych. Każda z tych linii może być opisana równaniem:

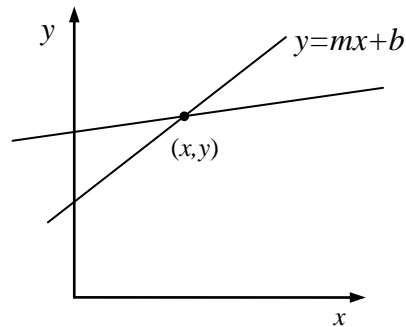
$$y = mx + b \quad (2.5)$$

gdzie m jest tangensem kąta, jaki linia tworzy z dodatnią półosią OX , zaś b jest punktem przecięcia linii z osią OY . Każdą linię przechodzącą przez punkt (x, y) można przedstawić jako punkt w przestrzeni parametrów m oraz b . W istocie, dla wszystkich linii

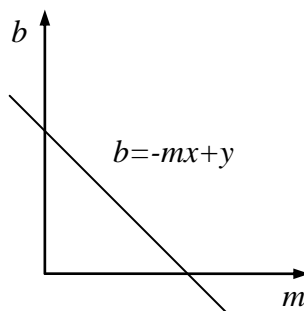
przechodzących przez ustalony punkt (x, y) istnieje dokładnie jedna wartość b dla danej wartości m .

$$b = y - mx \quad (2.6)$$

Każdemu punktowi przestrzeni (x, y) odpowiada linia w przestrzeni (m, b) , i odwrotnie – każdemu punktowi w przestrzeni (m, b) odpowiada linia w przestrzeni (x, y) .

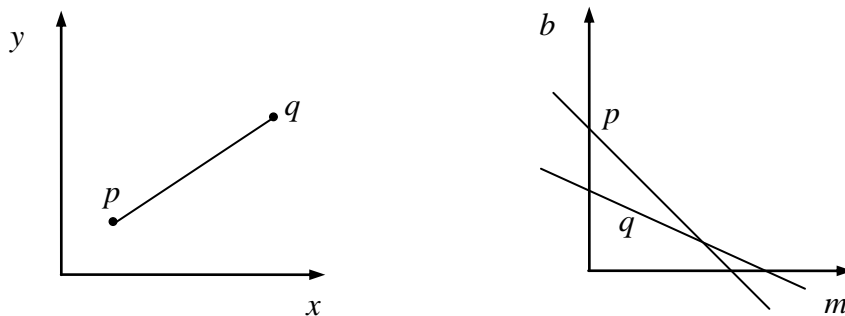


Rys. 2.4. Punkt na płaszczyźnie



Rys. 2.5. Prosta odpowiadająca pojedynczemu punktowi

Rozważmy teraz dwa punkty p oraz q w przestrzeni (x, y) , leżące na jednej prostej. W tym przypadku, każdemu punktowi odpowiada jedna linia w przestrzeni (m, b) . Linie te przecinają się w jednym punkcie, który wyznacza parametry prostej, na jakiej leżą piksele p oraz q . Uogólniając powyższe spostrzeżenia na większą ilość współliniowych punktów w przestrzeni (x, y) , można stwierdzić, że każdemu z tych punktów odpowiada linia prosta w przestrzeni (m, b) i linie te przecinają się w jednym miejscu. Współrzędne tego punktu określają parametry linii prostej przechodzącej przez współliniowe piksele.

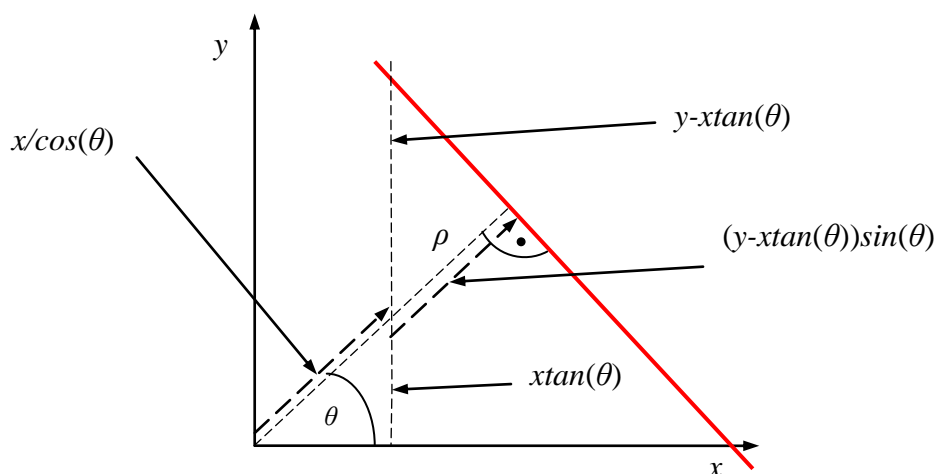


Rys. 2.6. Dwa współliniowe punkty

Reprezentacja w przestrzeni (m, b) posiada jednak zasadniczą wadę – dla linii pionowych parametry m oraz b przyjmują wartość nieskończoną. Dlatego, w przypadku transformacji Hougha, przyjęto inną reprezentację linii prostych. Jest to reprezentacja w przestrzeni (ρ, θ) [Duda72]:

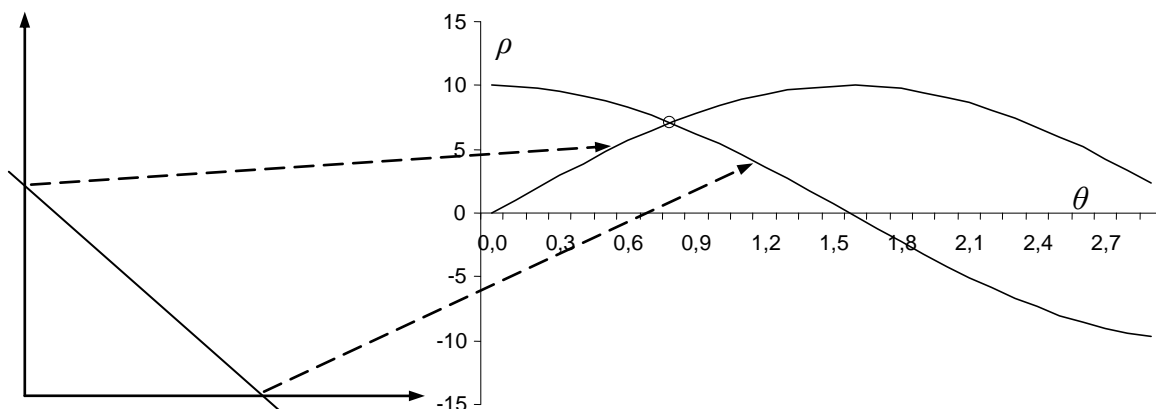
$$\begin{aligned}
 \rho &= \frac{x}{\cos(\theta)} + (y - x \tan(\theta)) \sin(\theta) \\
 &= \frac{x}{\cos(\theta)} + y \sin(\theta) - x \frac{\sin^2(\theta)}{\cos(\theta)} \\
 &= x \left(\frac{1 - \sin^2(\theta)}{\cos(\theta)} \right) + y \sin(\theta) \\
 &= x \cos(\theta) + y \sin(\theta)
 \end{aligned} \tag{2.7}$$

gdzie ρ jest odległością linii prostej od początku przyjętego układu współrzędnych, zaś θ jest kątem pomiędzy odcinkiem ρ , a dodatnią półosią OX (patrz rys. 2.7). Reprezentacja ta, w przeciwieństwie do reprezentacji w przestrzeni (m, b) , posiada tę zaletę, iż parametry ρ oraz θ nie przyjmują wartości nieskończonych. W przypadku analizy obrazów, współrzędne (x, y) punktu (piksela) w równaniu (2.7) są stałe, zaś ρ oraz θ są zmiennymi. Dla każdej kolejnej wartości kąta θ wyznacza się ze wzoru (2.7) odpowiadającą mu wartość ρ . W rezultacie każdemu punktowi (x, y) odpowiada w przestrzeni (ρ, θ) krzywa sinusoidalna. Jeśli na obrazie znajdują się dwa lub więcej współliniowe piksele, to odpowiadające im krzywe sinusoidalne przecinają się w jednym punkcie. Współrzędne (ρ, θ) tego punktu określają parametry linii, wzdłuż której położone są współliniowe piksele.



Rys. 2.7. Parametryzacja prostej za pomocą zmiennych ρ oraz θ

Transformacja Hougha jest implementowana za pomocą podziału ciągłej przestrzeni (ρ, θ) na zbiór skończonych komórek, zwanych akumulatorami. W czasie pracy algorytmu każdy punkt obrazu jest przekształcany w dyskretną krzywą sinusoidalną w przestrzeni (ρ, θ) . Wartości akumulatorów leżących wzdłuż tej krzywej są zwiększane o jeden. Jeśli przez daną komórkę przejdzie wiele krzywych sinusoidalnych, to osiągnie ona stosunkowo dużą wartość w porównaniu z sąsiednimi komórkami. W rezultacie, każdej linii prostej na obrazie odpowiada lokalne maksimum w przestrzeni (ρ, θ) . Położenie tego maksimum określa parametry odpowiadającej mu linii prostej, zaś jego wysokość odpowiada długości linii prostej.



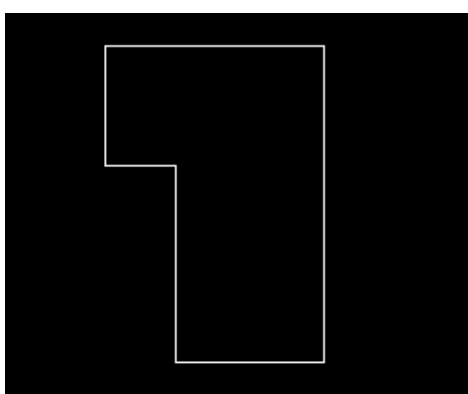
Rys. 2.8. Dwa współliniowe punkty i odpowiadające im krzywe sinusoidalne w przestrzeni Hougha

Wynikowy algorytm obliczania transformacji Hougha obrazu metodą inkrementowania akumulatorów przedstawiono w tabeli (2.1).

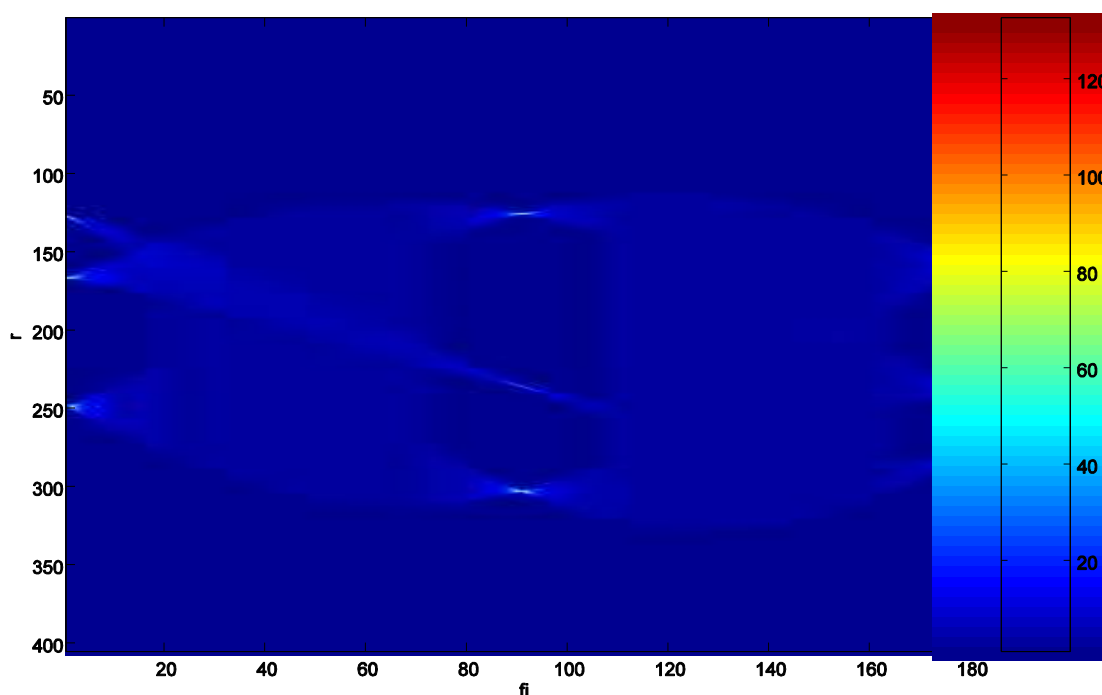
Tabela 2.1. Algorytm wyznaczania transformacji Hougha metodą inkrementowania akumulatorów

- Wyzeruj wszystkie komórki akumulatora
- Dla każdego piksela o wartości 1 powtórz:
 - dla $\theta=0^\circ, \dots, 179^\circ$
 - wyznacz $\rho = x \cos(\theta) + y \sin(\theta)$
 - $\text{Acc}(\rho, \theta) = \text{Acc}(\rho, \theta) + 1$

Na rysunku (2.9) przedstawiono przykładowy obraz binarny, a na rysunku 2.10 – jego transformację Hougha. Jak widać, transformacja ma sześć maksimów lokalnych, odpowiadających sześciu liniom prostym.



Rys. 2.9. Przykładowy obraz binarny



Rys. 2.10. Transformacja Hougha obrazu z rys. 2.9.

Oprócz standardowej transformacji Hougha istnieje również kilka jej odmian.

- **Combinatorial Hough Transform** (CHT) [Bent90, Illi88]. Transformacja ta w porównaniu ze standardową transformacją Hougha pozwala na uzyskanie większej rozdzielczości kątowej oraz radialnej znajdujących linii, kosztem jednak większej złożoności obliczeniowej. Możliwe jest uzyskanie rozdzielczości mniejszej niż 1 piksel.
- **Probabilistic Hough Transform** (PHT) [Berg91, Gala99]. Transformacja ta opiera się na założeniu, iż do skutecznej detekcji linii na obrazie wystarczy analiza tylko pewnej części wszystkich pikseli znajdujących się na nim [Kiry91]. Dlatego używa ona tylko części pikseli obrazu źródłowego. Są one wybierane losowo z jednakowym prawdopodobieństwem.
- **Adaptive Hough Transform** (AHT) [Illi87, Ecab04]. Istota tej transformacji polega na dynamicznym dopasowywaniu rozmiaru przestrzeni (ρ, θ) . Transformacja ta rozpoczyna swe działanie z małą liczbą akumulatorów. Po przetworzeniu pewnej liczby pikseli następuje adaptacyjne zwiększenie liczby akumulatorów w rejonach, gdzie uzyskano dotychczas największe wartości. Proces ten jest powtarzany iteracyjnie aż do osiągnięcia predefiniowanej rozdzielczości. W porównaniu do standardowej transformacji Hougha odmiana ta ma znacznie mniejsze wymagania dotyczące zajmowanej pamięci komputera.
- **Randomized Hough Transform** (RHT) [Kälv94]. Istota działania tej transformacji polega na losowym wybieraniu dwóch pikseli z analizowanego obrazu, a następnie wyznaczeniu na ich podstawie punktu w przestrzeni (ρ, θ) . Odpowiadająca temu punktowi komórka akumulatora zostaje zwiększona o 1. Działanie transformacji kończy się po dokonaniu z góry określonej liczby losowań. Istotną cechą tej transformacji jest to, iż z uwagi na losowy charakter wybierania kolejnych par pikseli, wyniki uzyskiwane za jej pomocą nie są w pełni powtarzalne.

- **Gray-scale Hough Transform** (GHT) [Lo95]. W odróżnieniu od pozostałych, transformacja ta działa bezpośrednio na obrazie monochromatycznym a nie na obrazie binarnym. W tym przypadku komórki akumulatora są rozłożone w trójwymiarowej przestrzeni (ρ , θ , G), gdzie G jest wartością przetwarzanego aktualnie piksela.

W niniejszej pracy zbadano skuteczność działania tych transformacji w kontekście detekcji linii Kikuchiego. Wyniki zostały przedstawione w rozdziale 6.

2.3.2. Metody obliczeniowe

Z matematycznego punktu widzenia, proces inkrementacji akumulatorów transformacji Hougha jest niczym innym jak całkowaniem obrazu wzdłuż odpowiadającym im liniom prostym. Dlatego standardowa transformacja Hougha może być rozpatrywana jako specjalny przypadek transformacji Radona, której działanie polega na całkowaniu obrazu wzdłuż pojedynczej linii prostej. Można więc ją wyznaczyć za pomocą transformacji Radona dla kąta całkowania w zakresie od $\theta=0^\circ$ do $\theta=180^\circ$.

W zaproponowanym w pracy algorytmie 2., opisanym w rozdziale 5., zmodyfikowano akumulatorowy algorytm transformacji Hougha, natomiast w opracowanym algorytmie 3., przedstawionym w rozdziale 6., użyto podejścia bazującego na FFT.

2.3.3. Zastosowania

Transformacja Hougha jest powszechnie stosowana do detekcji linii prostych na obrazach, w tym również do detekcji linii Kikuchiego [Lass98]. Na ogół spotykane w literaturze zastosowania tej transformacji połączone są z prostym, wstępnym przetwarzaniem obrazu źródłowego, polegającym najczęściej na detekcji krawędzi obiektów oraz prostej binaryzacji obrazu monochromatycznego [Fitt98]. W porównaniu z dotychczasowymi zastosowaniami transformacji Hougha, przedstawione w niniejszej pracy rozwiązania charakteryzują się zastosowaniem znacznie bardziej złożonych i zaawansowanych metod wstępnego przetwarzania obrazu źródłowego oraz autorskich metod przetwarzania wyniku działania transformacji Hougha. Zastosowane w rozprawie przetwarzanie wstępne obejmuje: korekcję tła obrazu, filtrację filtrami kierunkowymi, selektywne odszumianie, zwiększenie kontrastu za pomocą transformacji *curvelet*, binaryzację w oparciu o właściwości statystyczne obrazu oraz operacje morfologiczne na obrazie binarnym. Z kolei, przetwarzanie wyniku transformacji Hougha obejmuje zarówno niestosowane gdzie indziej metody ekstrakcji maksimów w przestrzeni transformacji, jak również nowe metody weryfikacji uzyskanych wyników.

2.4. Transformacja falkowa

Istota transformacji falkowej polega na analizie sygnału za pomocą zbioru funkcji bazowych, zwanych falkami. Istnieją dwa rodzaje tej transformacji: ciągła i dyskretna.

2.4.1. Ciągła transformacja falkowa

W ciągłej transformacji falkowej CWT (*Continuous Wavelet Transform*) [Daub92] dana funkcja zostaje przedstawiona za pomocą superpozycji falek. W efekcie uzyskuje się rozdzielanie informacji zawartych w jednym komponencie na kilka pasm o różnym zakresie częstotliwości, różnej skali czasowej i małej wzajemnej korelacji [Burr98].

Niech $\psi(t)$ oznacza funkcję spełniającą warunek:

$$C_\psi = \int_{-\infty}^{+\infty} \frac{|\hat{\psi}(\omega)|^2}{|\omega|} dt < \infty \quad (2.8)$$

gdzie $\hat{\psi}(t)$ jest zdefiniowane następująco:

$$\hat{\psi}(\omega) = \int \psi(t) e^{-j\omega t} dt \quad (2.9)$$

Aby warunek (2.8) był spełniony, funkcja $\psi(t)$ musi spełniać warunek zerowej wartości średniej, co w praktyce wymusza jej oscylacyjny charakter.

Funkcja $\psi(t)$ zwana jest falką *matczyną*. Związany z nią jest zbiór funkcji – falek *synowskich*, powstających w wyniku przesunięcia i dylatacji w czasie falki matczynej.

$$\psi_{\tau,s}(t) = \sqrt{|s|} \psi\left(\frac{t-\tau}{s}\right) \quad (2.10)$$

gdzie $s \in \mathbf{R}$, $s \neq 0$ i jest ono czynnikiem skalującym odpowiadającym za rozdzielczość analizy, $\tau \in \mathbf{R}$ i jest przesunięciem w czasie, zaś współczynnik $\sqrt{|s|}$ służy zachowaniu takiej samej energii wszystkich falek niezależnie od parametru s . Na ogół przyjmuje się, że $s=2^j$, oraz $\tau=k2^j$ gdzie j, k są liczbami naturalnymi.

Ostatecznie ciągła transformacja falkowa jest zdefiniowana następująco:

$$\gamma(\tau,s) = \int_{-\infty}^{+\infty} f(t) \frac{1}{\sqrt{|s|}} \psi^*\left(\frac{t-\tau}{s}\right) dt \quad (2.11)$$

gdzie $\psi^*(t)$ jest funkcją sprzężoną z funkcją $\psi(t)$. Wartość $\gamma(\tau,s)$ jest miarą podobieństwa aktualnego fragmentu sygnału $f(t)$ do falki $\psi_{\tau,s}(t)$ przy danym położeniu τ oraz rozdzielczości s .

Funkcja $f(t)$ może być odtworzona za pomocą transformacji odwrotnej:

$$f(t) = \frac{1}{C_\psi} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \gamma(\tau,s) \psi\left(\frac{t-\tau}{s}\right) d\tau \frac{ds}{|s|^2} \quad (2.12)$$

gdzie C_ψ jest stałą określoną równaniem (2.8).

Ponieważ parametrami transformacji falkowej są położenie oraz przeskalowanie falki, dlatego transformacja ta pozwala na uzyskanie informacji o zmianie poszczególnych składowych częstotliwościowych w czasie.

Najważniejsze własności ciągłej transformacji falkowej to [Burr98]:

- Skalowanie

$$g(t) = \frac{1}{\sqrt{l}} f\left(\frac{t}{l}\right) \Leftrightarrow \gamma_g(\tau,s) = \gamma_f\left(\frac{\tau}{l}, \frac{s}{l}\right) \quad (2.13)$$

- Zachowanie energii

$$\int_{-\infty}^{\infty} |f(t)|^2 dt = \frac{1}{C_\psi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} |\gamma_f(\tau,s)|^2 \frac{d\tau ds}{\tau^2} \quad (2.14)$$

- Przesunięcie

$$g(t) = f(t-l) \Leftrightarrow \gamma_g(\tau, s) = \gamma_f(\tau, s-l) \quad (2.15)$$

2.4.2. Dyskretna transformacja falkowa

Dyskretna transformacja falkowa DWT (*Discrete Wavelet Transform*) [Mall99] sygnału $x[n]$ jest wyznaczana poprzez przefiltrowanie sygnału za pomocą serii kaskadowo połączonych filtrów. Sygnał jest równocześnie filtrowany za pomocą filtra dolnoprzepustowego $g[k]$ (związanego z falką *matczyną*):

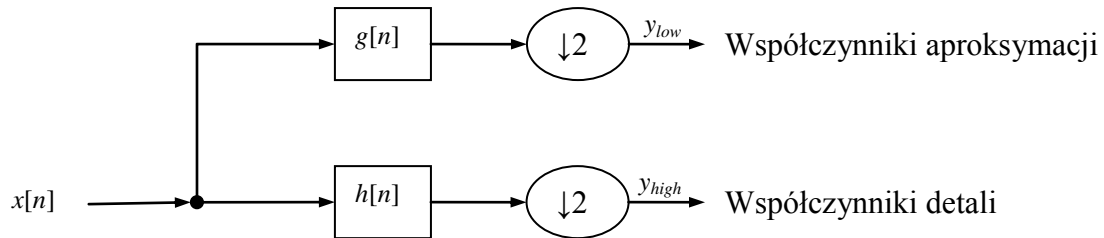
$$y_{low}[n] = (x * g)[n] = \sum_{k=-\infty}^{k=+\infty} x[k] \cdot g[n-k] \quad (2.16)$$

oraz filtra górnoprzepustowego $h[n]$ (związanego z falką *synowską*).

$$y_{high}[n] = (x * h)[n] = \sum_{k=-\infty}^{k=+\infty} x[k] \cdot h[n-k] \quad (2.17)$$

gdzie $g[k]$ i $h[n]$ oznaczają odpowiedzi impulsowe odpowiednich filtrów.

W rezultacie otrzymuje się dwa przefiltrowane sygnały, stanowiące dekompozycję sygnału wejściowego na sygnał o niskich częstotliwościach y_{low} oraz wysokich częstotliwościach y_{high} . Kolejne wartości próbek sygnałów y_{low} oraz y_{high} nazywane są współczynnikami transformacji falkowej. Ponieważ ograniczono o połowę szerokość pasma częstotliwościowego sygnałów, dlatego zgodnie z twierdzeniem Nyquista [Ziel02] można z każdego z nich usunąć co drugą próbkę (rys. 2.11).



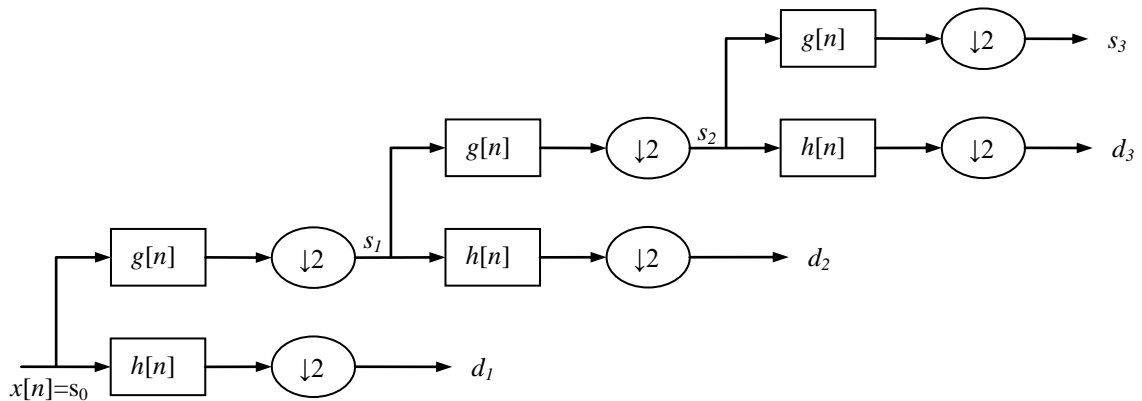
Rys. 2.11. Falkowa dekompozycja sygnału

Dyskretna transformacja falkowa jest na ogół realizowana za pomocą wielopoziomowej dekompozycji sygnału

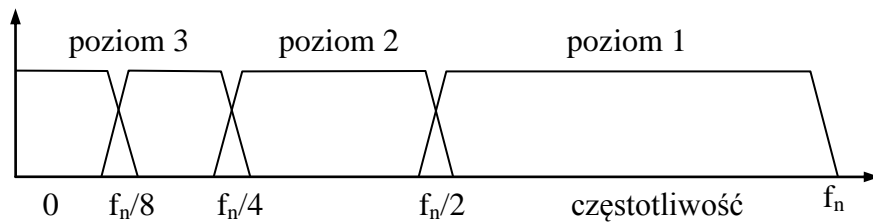
$$\begin{aligned} s_{l+1}[n] &= \sum_{k=-\infty}^{k=+\infty} s_l[2 \cdot n + k] \cdot g_l[k] \\ d_{l+1}[n] &= \sum_{k=-\infty}^{k=+\infty} s_l[2 \cdot n + k] \cdot h_l[k] \end{aligned} \quad (2.18)$$

gdzie g_l oraz h_l są odpowiednio filtrami dolno i górnoprzepustowymi na l -tym poziomie dekompozycji. Na tym poziomie sygnał wejściowy s_l jest dzielony na dwa sygnały: s_{l+1} oraz d_{l+1} . Oba te sygnały posiadają dwukrotnie mniejszą liczbę próbek w stosunku do sygnału wejściowego s_l . Sygnał s_{l+1} stanowi niskoczęstotliwościową wersję sygnału wejściowego s_l (tzw. aproksymata), zaś d_{l+1} – jego reprezentację wysokoczęstotliwościową (tzw. detale). Na ich podstawie jest możliwe bezstratne odtworzenie s_l . Następnie sygnał dolnopasmowy s_{l+1}

jest powtórnie poddawany dekompozycji, zaś sygnał górnopasmowy d_{l+1} jest zapisywany na wyjściu transformacji (rys. 2.12). Z uwagi na proces dekompozycji długość sygnału wejściowego powinna być równa 2^n , gdzie n jest liczbą poziomów dekompozycji.



Rys. 2.12. Wielopoziomowa dekompozycja sygnału



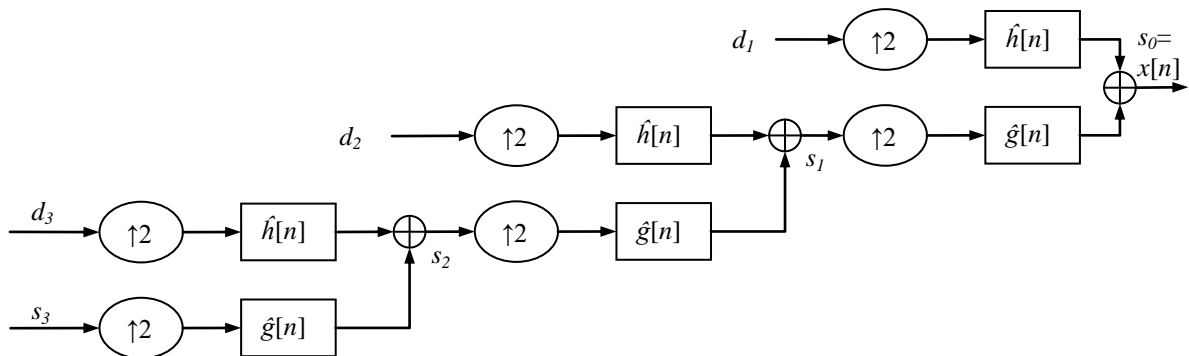
Rys. 2.13. Pasma częstotliwościowe dla kolejnych poziomów dekompozycji

Odwrotna dyskretna transformacja falkowa (synteza) liczona jest następująco:

$$s_{l-1}[2n] = \sum_{i=-\infty}^{i=+\infty} s_l[n+i] \cdot \hat{g}[2i+1] + \sum_{j=-\infty}^{j=+\infty} d_l[n+j] \cdot \hat{h}[2j+1] \quad (2.19)$$

$$s_{l-1}[2n+1] = \sum_{i=-\infty}^{i=+\infty} s_l[n+i] \cdot \hat{g}[2i] + \sum_{j=-\infty}^{j=+\infty} d_l[n+j] \cdot \hat{h}[2j]$$

gdzie $\hat{h}[n]$ oraz $\hat{g}[n]$ są odpowiedziami impulsowymi filtra górno- i dolnoprzepustowego, powstałymi przez odwrócenie kolejności współczynników odpowiednich filtrów analizy.



Rys. 2.14. Schemat trójpoziomowej syntezy falkowej sygnału

W przypadku sygnałów dwuwymiarowych (obrazów) algorytm dyskretnej transformacji falkowej liczony jest w ten sposób, że dokonywana jest oddzielnie analiza wierszy oraz kolumn obrazu:

- a) w pierwszym kroku wiersze obrazu są, za pomocą układów z rysunku (2.12), dekomponowane na współczynniki aproksymacji i detali;
- b) w drugim kroku kolejne kolumny otrzymanych wcześniej podobrazów aproksymacji i detali są, za pomocą tych samych układów z rysunku (2.12), ponownie dekomponowane na współczynniki aproksymacji i detali.

Po dekompozycji obraz jest reprezentowany przez cztery macierze współczynników falkowych, które łącznie mają tyle samo elementów, co obraz oryginalny (rys. 2.15): sygnał aproksymacji (dolnopasmowy LL) oraz trzy sygnały górnopasmowe: horyzontalny LH (szczegóły w kierunku poziomym), wertykalny HL (szczegóły w kierunku pionowym) oraz diagonalny HH (szczegóły w kierunku ukośnym). Przykładowo, pasmo HL powstaje w wyniku przefiltrowania obrazu źródłowego za pomocą filtra górnoprzepustowego w kierunku poziomym (oznaczenie H), a następnie przefiltrowania go za pomocą filtra dolnoprzepustowego (oznaczenie L) w kierunku pionowym. Z uwagi na proces dekompozycji długość sygnału wejściowego powinna być równa 2^n , gdzie n jest liczbą poziomów dekompozycji.

Równania opisanej powyżej falkowej dekompozycji obrazu są następujące:

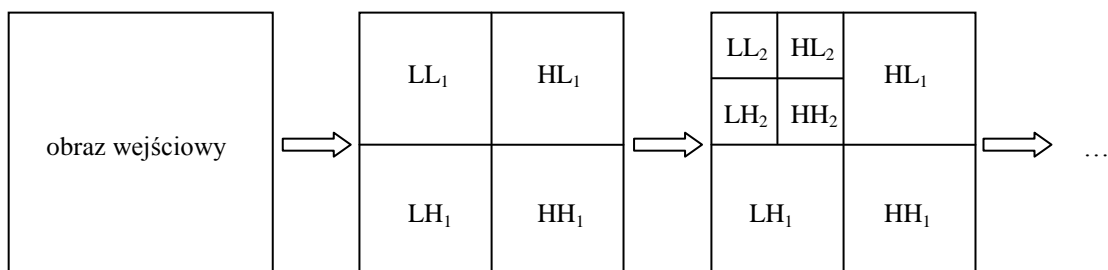
$$LL_{j+1}(k_x, k_y) = \sum_{l_y=-\infty}^{+\infty} g(l_y - 2k_y) \cdot \left(\sum_{l_x=-\infty}^{+\infty} g(l_x - 2k_x) \cdot LL_j(l_x, l_y) \right) \quad (2.20)$$

$$LH_{j+1}(k_x, k_y) = \sum_{l_y=-\infty}^{+\infty} h(l_y - 2k_y) \cdot \left(\sum_{l_x=-\infty}^{+\infty} g(l_x - 2k_x) \cdot LL_j(l_x, l_y) \right) \quad (2.21)$$

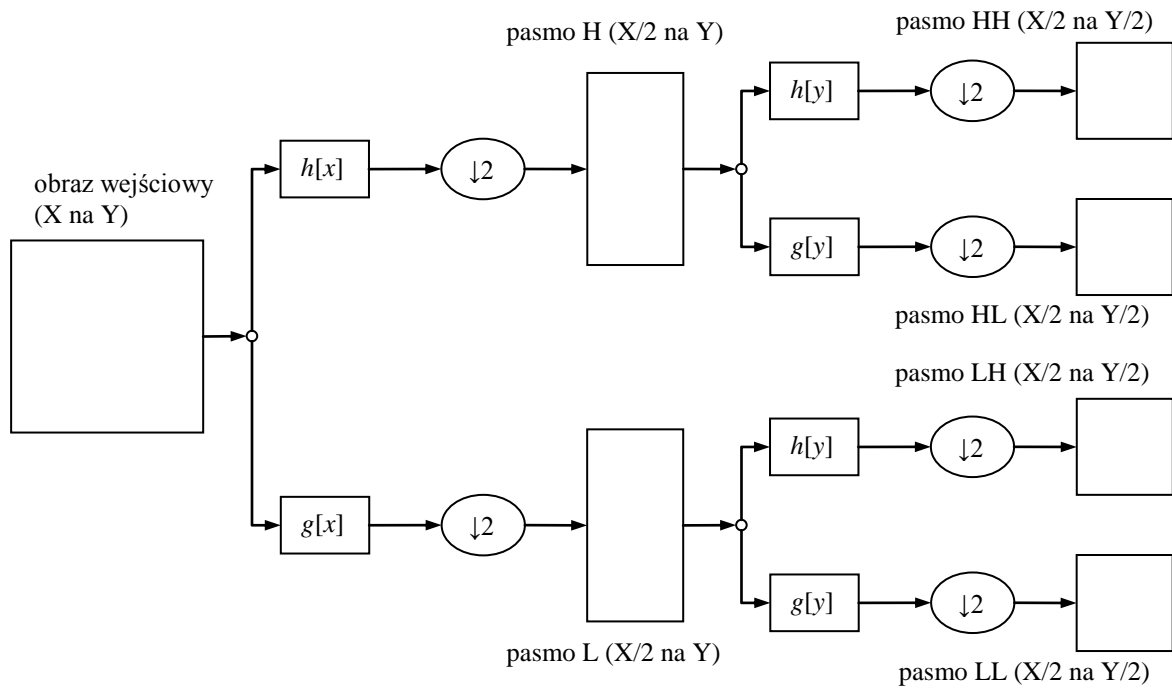
$$HL_{j+1}(k_x, k_y) = \sum_{l_y=-\infty}^{+\infty} g(l_y - 2k_y) \cdot \left(\sum_{l_x=-\infty}^{+\infty} h(l_x - 2k_x) \cdot LL_j(l_x, l_y) \right) \quad (2.22)$$

$$HH_{j+1}(k_x, k_y) = \sum_{l_y=-\infty}^{+\infty} h(l_y - 2k_y) \cdot \left(\sum_{l_x=-\infty}^{+\infty} h(l_x - 2k_x) \cdot LL_j(l_x, l_y) \right) \quad (2.23)$$

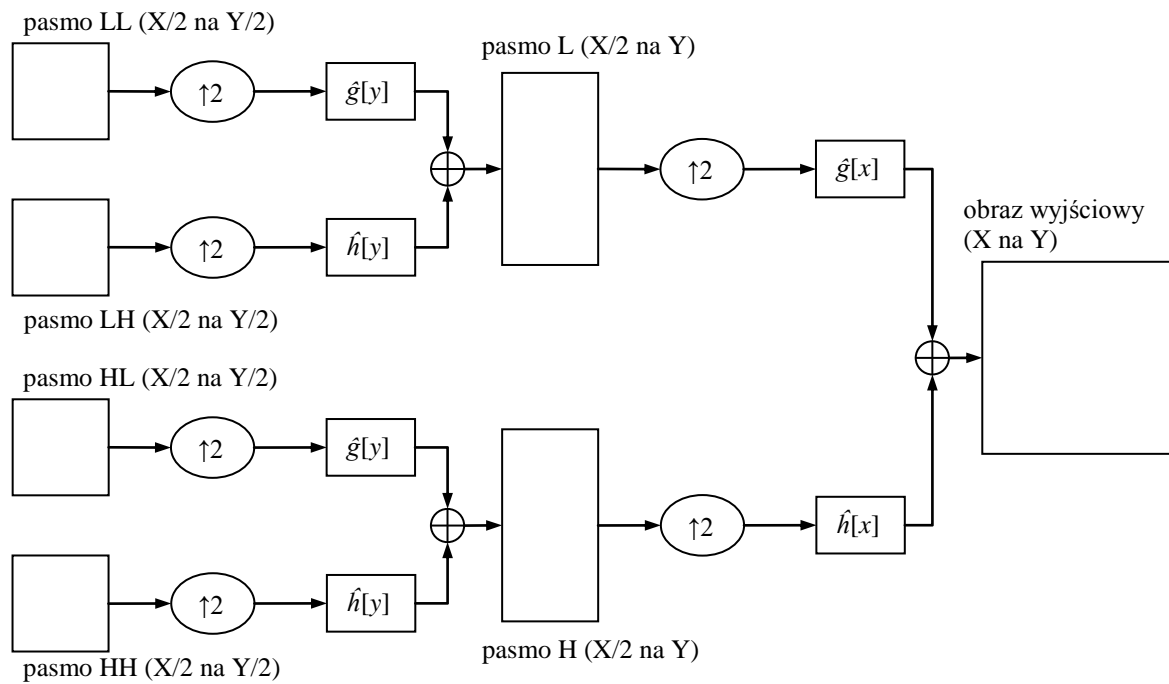
W pracy transformacja falkowa jest zastosowana na etapie wstępnego przetwarzania obrazów mikroskopowych w celu ich odsumiania.



Rys. 2.15. Dekompozycja obrazu na podpasma



Rys.2.16. Falkowa dekompozycja (analiza) obrazu



Rys. 2.17. Falkowa synteza obrazu

2.4.3. Transformacja *ridgelet* [Cand98]

Dwuwymiarowa ciągła transformacja *ridgelet* może być zdefiniowana następująco: Wybiera się ciągłą funkcję spełniającą warunek: $\psi : \mathbf{R} \rightarrow \mathbf{R}$

$$\int \frac{|\psi(\xi)|^2}{|\xi|^2} d\xi < \infty \quad (2.24)$$

który jest spełniony jeśli:

$$\int \psi(t) dt = 0 \quad (2.25)$$

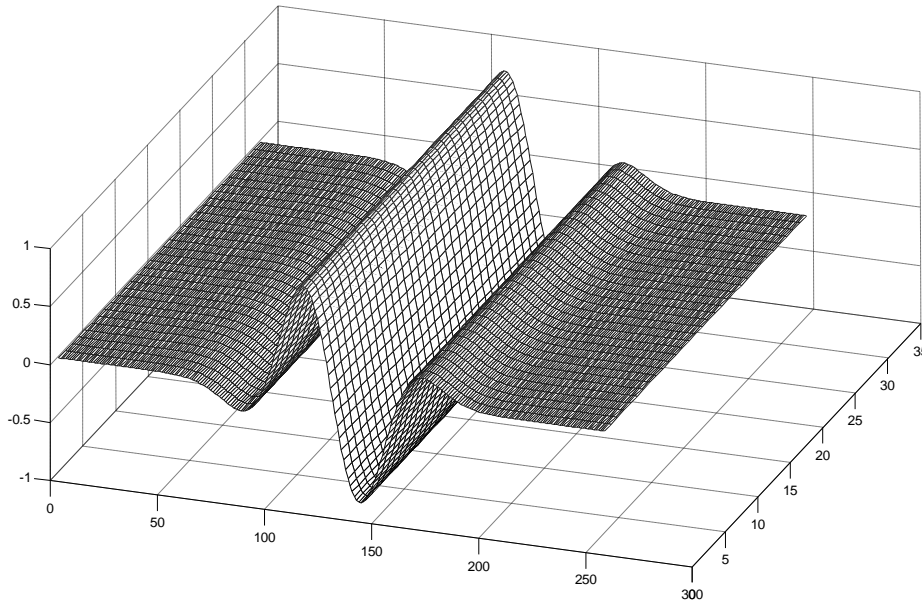
Ponadto zakłada się, że:

$$\int_0^\infty \frac{|\psi(\xi)|^2}{|\xi|^2} d\xi = 1 \quad (2.26)$$

Dla każdego $a > 0$ oraz każdego $b \in \mathbf{R}$ i każdego $\theta \in [0; 2\pi)$ zdefiniowana jest funkcja *ridgelet*:

$$\psi_{a,b,\theta}(x) = \frac{1}{\sqrt{a}} \psi\left(\frac{x_1 \cos(\theta) + x_2 \sin(\theta) - b}{a}\right) \quad (2.27)$$

Funkcja $\psi_{a,b,\theta}(x)$ jest stała wzdłuż prostej $x_1 \cos(\theta) + x_2 \sin(\theta) = b$



Rys. 2.18. Przykład funkcji *ridgelet*

Jeśli dana jest całkowna funkcja $f(x)$, to współczynniki transformacji *ridgelet* wyrażone są następująco:

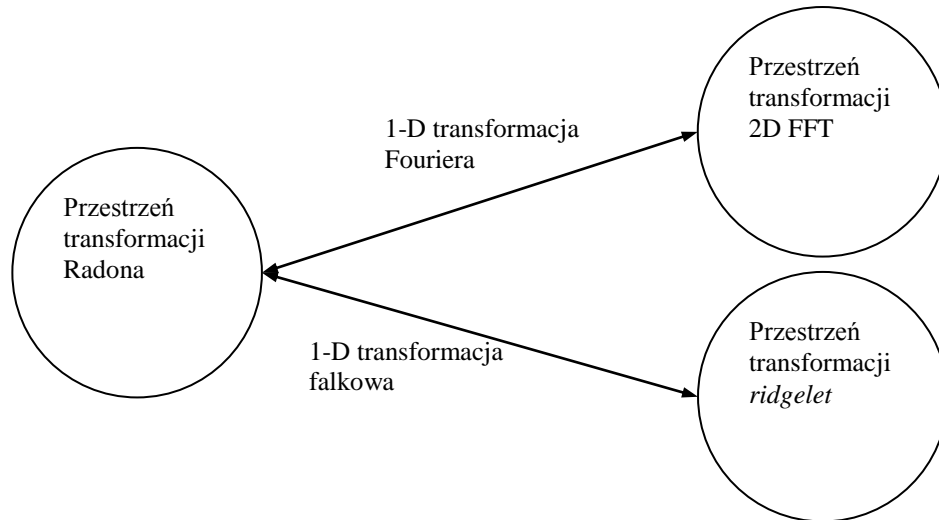
$$R_j(a, b, \theta) = \int \bar{\psi}_{a,b,\theta}(x) f(x) dx \quad (2.28)$$

Transformacja odwrotna ma postać:

$$f(x) = \int_0^{2\pi} \int_{-\infty}^{+\infty} \int_0^{+\infty} R_f(a, b, \theta) \psi_{a,b,\theta}(x) \frac{da}{a^3} db \frac{d\theta}{4\pi} \quad (2.29)$$

Przy stałym θ i zmiennym t dwuwymiarowa transformacja *ridgelet* może być interpretowana jako zastosowanie transformacji falkowej do transformacji Radona.

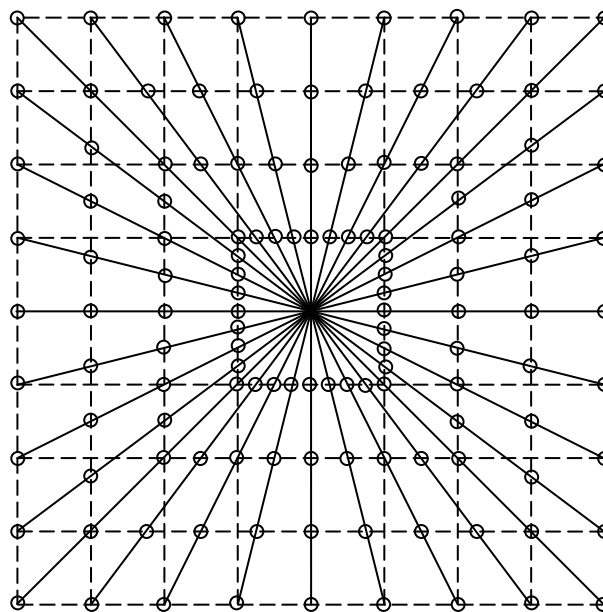
$$R_f(a, b, \theta) = \int \mathbf{R}f(\theta, t) \psi\left(\frac{t-b}{a}\right) dt \quad (2.30)$$



Rys. 2.19. Związek pomiędzy transformacjami Radona, Fouriera, *ridgelet* [Do03]

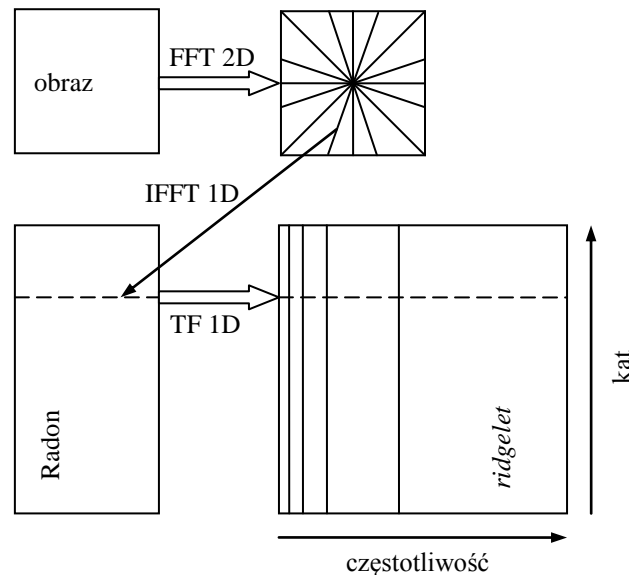
Szybki algorytm obliczania transformacji *ridgelet*

Z uwagi na szybkość algorytmu, transformację *ridgelet* oblicza się zazwyczaj za pomocą transformacji Radona, wyznaczanej z wykorzystaniem szybkiej transformacji Fouriera. Najpierw wyznacza się 2D FFT. Następnie dokonuje się interpolacji widma 2D wzdłuż linii prostych, przechodzących przez środek dwuwymiarowej przestrzeni częstotliwościowej (rys. 2.20).



Rys. 2.20. Siatka interpolacji

W dalszej kolejności wyznaczana jest jednowymiarowa odwrotna transformacja Fouriera dla próbek położonych wzdłuż każdej z interpolowanych linii. Ostatni etap polega na wyznaczeniu jednowymiarowej transformacji falkowej wzdłuż radialnego wymiaru transformacji Radona (rys. 2.21).



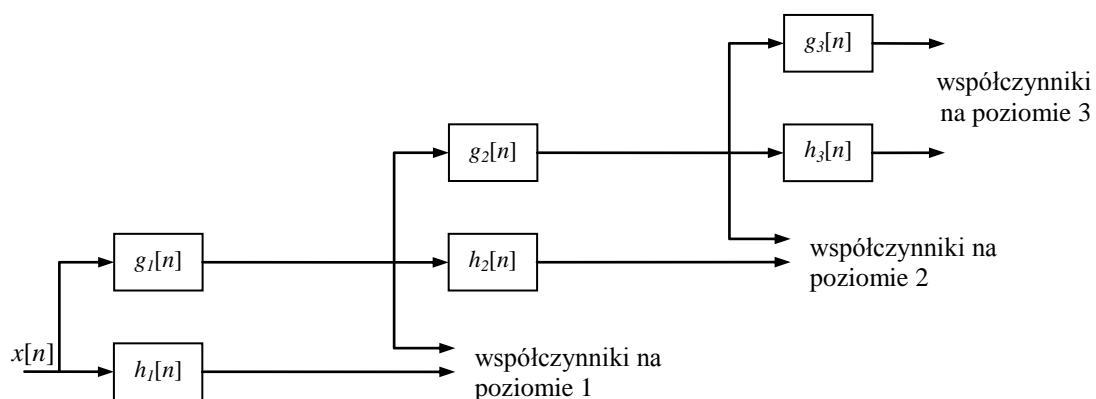
Rys. 2.21. Schemat dyskretnej transformacji ridgelet [Star03]

Algorytm do szybkiego obliczania transformacji *ridgelet* jest atrakcyjny z punktu widzenia złożoności obliczeniowej, gdyż liczba jego operacji wynosi $O(n^2 \log(n))$ dla obrazu o wymiarach n na n pikseli. Ostatecznie transformacja *ridgelet* dla obrazu n na n pikseli jest macierzą o wymiarach $2n$ na $2n$.

Transformacja *ridgelet* jest odwracalna i nienadmiarowa. Stanowi ona optymalną reprezentację linii prostych na obrazie. W pracy była ona zastosowana do wyznaczania transformacji *curvelet*.

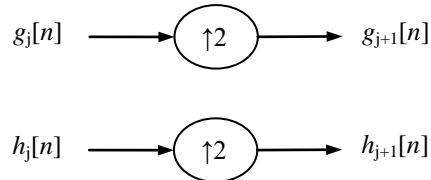
2.4.4. Transformacja *curvelet* [Star02]

Istota tej transformacji polega na dekompozycji obrazu za pomocą transformacji falkowej, a następnie analizie każdego podpasma za pomocą transformacji *ridgelet*. Takie podejście stwarza możliwość analizy obrazu z wykorzystaniem bloków o różnej wielkości.

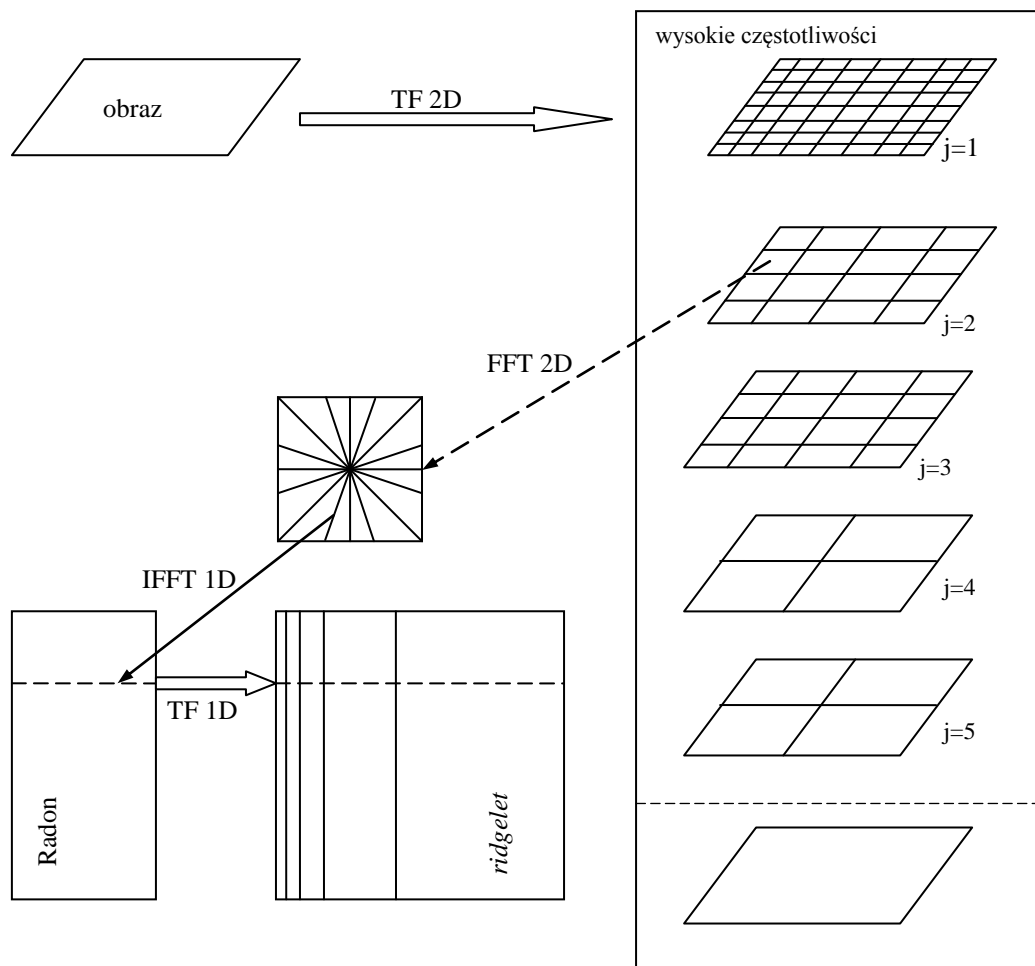


Rys. 2.22. Dekompozycja sygnału w algorytmie *à trous* [Wiki1]

Algorytm transformacji *curvelet* wykorzystuje algorytm *à trous*, zwany również stacjonarną transformacją falkową (*Stationary Wavelet Transform*). Istota tego algorytmu polega na tym, że nie dokonuje się w nim zmniejszenia częstotliwości próbkowania sygnału po każdej operacji filtracji (patrz rys. 2.22). Zamiast tego, na każdym poziomie dekompozycji, same filtry są nadpróbkowywane przed dokonaniem filtracji (patrz rys. 2.23)



Rys. 2.23. Nadpróbkowanie filtrów na kolejnych poziomach transformacji



Rys. 2.24. Schemat transformacji *curvelet* [Star03]

Tabela 2.2. Algorytm transformacji curvelet [Star03]

<p>a) Wyznacz dwuwymiarową transformację falkową obrazu. Najczęściej stosuje się algorytm <i>à trous</i>.</p> <p>b) $B_1=B_{\min}$</p> <p>c) Dla każdego $j=1,\dots,J$:</p> <ul style="list-style-type: none"> • podziel podpasmo w_j za pomocą bloków o rozmiarze B_j i zastosuj transformację <i>ridgelet</i> do każdego bloku • jeśli j modulo 2=1, to $B_{j+1}=2B_j$ • w przeciwnym wypadku $B_{j+1}=B_j$.
--

W pracy transformacja *curvelet* była zastosowana na etapie wstępnego przetwarzania obrazu źródłowego do zwiększania jego kontrastu.

2.5. Odszumianie obrazów

Obraz cyfrowy jest na ogół zapisywany jako macierz wartości odpowiadających odcieniom lub kolorom poszczególnych pikseli. W przypadku obrazów TKP/CBED mamy do czynienia z obrazami monochromatycznymi, tj. składającymi się z pikseli, które charakteryzują tylko jeden parametr – ich jasność (luminancja). Wartość każdego piksela jest ustalana za pomocą pomiaru jasności światła, dokonywanego zwykle za pomocą przetwornika CCD z odpowiednim układem optycznym. W każdej komórce przetwornika dokonuje się zliczenia padających fotonów w określonym odcinku czasu. Gdy jasność źródła światła jest stała, wtedy liczba fotonów odbieranych przez każdą komórkę przetwornika CCD fluktuuje wokół pewnej wartości średniej. Fluktuacje te przejawiają się na obrazie w postaci szumu zwanego fotoelektrycznym $n_f(x, y)$. Posiada on rozkład Poissona, który w granicznym przypadku jest aproksymowany rozkładem Gaussa [Jain89]. Tak więc w pierwszym przybliżeniu obraz uzyskiwany za pomocą przetwornika CCD można przedstawić następująco:

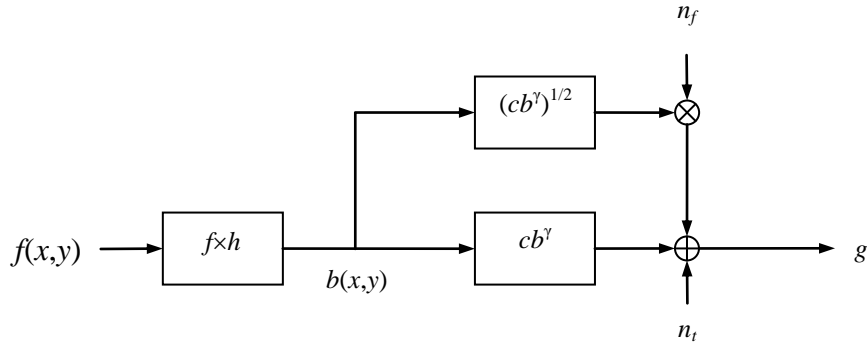
$$g(x, y) = c(b(x, y))^\gamma + \left(cb(x, y)^\gamma \right)^{\frac{1}{2}} \cdot n_f(x, y) \quad (2.31)$$

gdzie $b(x, y)$ jest obrazem wejściowym przetwornika, $g(x, y)$ jest obrazem wyjściowym, zaś γ jest stałą.

Dodatkowo, jeśli przetwornik nie jest dostatecznie dobrze chłodzony, wtedy w każdej komórce generowany jest dodatkowy szum termiczny $n_t(x, y)$. Szum ten posiada rozkład Gaussa o wartości średniej równej zero. Ponadto, oryginalny obraz $f(x, y)$ może być zniekształcony przez optykę przetwornika $h(x, y)$. Uwzględniając te czynniki, otrzymuje się ostateczną postać równania opisującego obraz uzyskiwany z przetwornika [Pita01]:

$$g(x, y) = c(f(x, y) \times h(x, y))^\gamma + \left(c(f(x, y) \times h(x, y))^\gamma \right)^{\frac{1}{2}} \cdot n_f(x, y) + n_t(x, y) \quad (2.32)$$

gdzie symbol \times oznacza splot 2D.



Rys. 2.25. Proces formowania obrazu cyfrowego [Pita01]

Z matematycznego punktu widzenia, szum może być modelowany (interpretowany) na kilka sposobów [Bakh99]:

- Jako biały lub kolorowy proces stochastyczny. W tym przypadku dla odsumiania istotna jest znajomość częstotliwościowej charakterystyki szumu.
- Jako stacjonarny lub niestacjonarny proces losowy. W tym przypadku ważną rolę odgrywają fluktuacje w czasie statystycznych właściwości szumu.
- Jako sekwencja liczb losowych o różnych funkcjach rozkładu gęstości prawdopodobieństwa. Tutaj istotna dla odsumiania rolę odgrywa znajomość funkcji rozkładu gęstości prawdopodobieństwa.

Najczęściej spotykaną i stosowaną miarą poziomu szumu na obrazie jest tzw. SNR (*Signal Noise Ratio*), która może być zdefiniowana następująco:

$$\text{SNR} = \frac{\sigma(v)}{\sigma(n)} \quad (2.33)$$

gdzie $\sigma(v)$ oznacza empirycznie wyznaczone odchylenie standardowe $v(i)$. Wielkość $\sigma(v)$ jest zdefiniowana następująco:

$$\sigma(v) = \left(\frac{1}{|I|} \sum_i (v(i) - \bar{v})^2 \right)^{\frac{1}{2}} \quad (2.34)$$

gdzie

$$\bar{v} = \frac{1}{|I|} \sum_{i \in I} v(i) \quad (2.35)$$

W praktyce na ogół brak jest wiedzy odnośnie statystycznych właściwości szumu podlegającego filtracji. Jednak w pewnych przypadkach możliwe jest dysponowanie niektórymi parametrami szumu, jak np. wartość średnia lub wariancja, a w rzadszych przypadkach również funkcją rozkładu gęstości prawdopodobieństwa szumu. W takiej sytuacji możliwe jest jeszcze większe stłumienie szumu, a tym samym powiększenie wskaźnika SNR dla przefiltrowanego sygnału.

Istnieją dwa zasadnicze podejścia do odsumiania obrazów: filtracje w dziedzinie przestrzeni oraz filtracje w dziedzinie współczynników transformacji obrazu. Każdy z tych typów posiada liniową i nieliniową odmianę.

2.5.1. Liniowa filtracja w dziedzinie przestrzeni

Najczęściej spotykanym przykładem tego typu filtracji jest filtr uśredniający. Istota jego działania polega na zastępowaniu każdego piksela obrazu wartością średnią z otaczających go pikseli, rozmieszczonych na ogół w kwadratowym oknie W [Plat00].

$$y(m, n) = \sum_{i=-M}^M \sum_{j=-N}^N x(m-i, n-j) \cdot w(i, j) \quad (2.36)$$

gdzie $w(i, j)$ oznacza odpowiednio zaprojektowaną, najczęściej dolnoprzepustową odpowiedź impulsową filtra. Jeżeli $w(i, j) = 1$, filtr ten jest optymalny dla szumu gaussowskiego w sensie minimalizacji błędu średniokwadratowego [Motw04]. W tym przypadku szum jest NM -krotnie tłumiony [Jain89]. Filtry uśredniające mają tendencję do zbytniego rozmywania krawędzi oraz innych drobnych obiektów na obrazie.

Innym przykładem filtracji obrazów w dziedzinie przestrzeni jest liniowa filtracja Wienera [Lim90]. Metoda ta wymaga znajomości wariancji v filtrowanego szumu. Daje ona dobre rezultaty tylko wtedy, gdy obraz jest gładki, tj. nie zawiera zbyt dużej liczby ostrych krawędzi. Istota filtracji Wienera polega na estymacji lokalnej wartości średniej i wariancji wokół każdego piksela. Wartość średnia jest wyrażona wzorem:

$$\mu = \frac{1}{NM} \sum_{x, y \in \eta} A(x, y) \quad (2.37)$$

zaś wariancja wzorem:

$$\sigma^2 = \frac{1}{NM} \sum_{x, y \in \eta} A^2(x, y) - \mu^2 \quad (2.38)$$

gdzie η jest sąsiedztwem piksela $A(x, y)$, szerokość sąsiedztwa wynosi N , zaś jego wysokość M . Wartość przefiltrowanego piksela dana jest następująco:

$$B(x, y) = \mu + \frac{\sigma^2 - v^2}{\sigma^2} (A(x, y) - \mu) \quad (2.39)$$

gdzie v jest wariancją szumu i jest parametrem metody.

2.5.2. Nieliniowa filtracja w dziedzinie przestrzeni

W tym przypadku filtracja jest dokonywana bez próby identyfikacji właściwości statystycznych szumu. Istota tej metody polega na dokonaniu filtracji dolnoprzepustowej przy założeniu, że szum znajduje się w obszarze wyższych częstotliwości. Jednym z przykładów tego typu filtracji jest filtracja medianowa: wartość przefiltrowanego piksela jest medianą z wartości sąsiadujących z nim pikseli rozmieszczonych w oknie W [Mari91]:

$$Y(m, n) = \text{mediana}\{X(m-k, n-l), (k, l) \in W\} \quad (2.40)$$

gdzie $X(m, n)$ oraz $Y(m, n)$ są odpowiednio: obrazem wejściowym i wyjściowym.

Okno W ma na ogół kształt kwadratu. Jeśli liczba pikseli w oknie jest parzysta, to jako medianę przyjmuje się wartość średnią z dwóch środkowych pikseli.

Oprócz standardowej filtracji medianowej istnieją również jej odmiany, jak np. ważona mediana [Brow84, Yang95] czy zmiękczone (*relaxed*) mediana [Hanz99].

2.5.3. Metoda pochodnych cząstkowych [Lysa03]

Innym, bardziej złożonym, przykładem filtracji nieliniowej w dziedzinie przestrzeni jest metoda pochodnych cząstkowych PDE (*Partial Differential Equation*).

Założmy, że obraz źródłowy $u(x, y)$ jest zaszumiony szumem addytywnym $u_0(x, y) = u(x, y) + \eta(x, y)$, gdzie $(x, y) \in \Omega$. Szum w obrazie może być rozpatrywany jako szybkie oscylacje sygnału w obrębie małego obszaru obrazu. Założmy, że poziom szumu jest w przybliżeniu znany.

$$\|u - u_0\|^2 = \int_{\Omega} (u - u_0)^2 dx dy \approx \sigma^2 \quad (2.41)$$

Ogólnie problem odzsumiania może być przedstawiony jako zagadnienie minimalizacji funkcjonału $R(u)$ będącego miarą oscylacji sygnału, przy warunku (2.41) [Rudi92, Chan00b]:

$$\begin{cases} \min R(u) \\ \text{przy warunku (2.41)} \end{cases} \quad (2.42)$$

Najczęściej stosuje się funkcjonal postaci:

$$R_1(u) = \int_{\Omega} (|u_{xx}| + |u_{yy}|) dx dy \quad (2.43)$$

gdzie $u_{xx} = \frac{\partial^2 u}{\partial x^2}$, $u_{yy} = \frac{\partial^2 u}{\partial y^2}$.

Metoda poszukiwania minimum $R(u)$ polega na zastosowaniu funkcjonału Lagrange'a:

$$L(u, \lambda) = \int_{\Omega} (|u_{xx}| + |u_{yy}|) dx dy + \frac{\lambda}{2} \cdot \left(\int_{\Omega} (u - u_0)^2 dx dy - \sigma^2 \right) \quad (2.44)$$

Rozwiązanie optymalne otrzymuje się dla:

$$\frac{\partial L}{\partial \lambda} = \frac{1}{2} \int_{\Omega} (u - u_0)^2 dx dy - \frac{1}{2} \sigma^2 = 0 \quad (2.45)$$

$$\frac{\partial L}{\partial u} = \int_{\Omega} \left(\frac{u_{xx}}{|u_{xx}|} + \frac{u_{yy}}{|u_{yy}|} \right) dx dy + \lambda \cdot \int_{\Omega} (u - u_0) dx dy \quad (2.46)$$

u będące rozwiązaniem powyższych równań spełnia następujące nieliniowe równanie różniczkowe:

$$\left(\frac{u_{xx}}{|u_{xx}|} \right)_{xx} + \left(\frac{u_{yy}}{|u_{yy}|} \right)_{yy} + \lambda \cdot (u - u_0) = 0 \quad (2.47)$$

Rozwiązanie równania (2.47) dla $t > 0$ oraz $(x, y) \in \Omega$ ma postać:

$$u = - \left(\frac{u_{xx}}{|u_{xx}|} \right)_{xx} - \left(\frac{u_{yy}}{|u_{yy}|} \right)_{yy} - \lambda \cdot (u - u_0) \quad (2.48)$$

$$\lambda = - \frac{1}{\sigma^2} \int_{\Omega} \left(\frac{u_{xx}}{|u_{xx}|} \cdot (u - u_0)_{xx} + \frac{u_{yy}}{|u_{yy}|} \cdot (u - u_0)_{yy} \right) dx dy \quad (2.49)$$

W przypadku obrazów dyskretnych, współrzędne kolejnych pikseli są umieszczone na prostokątnej siatce:

$$\begin{aligned}x &= i\Delta x & i &= 0, 1, \dots, I \\y &= j\Delta y & j &= 0, 1, \dots, J\end{aligned}\quad (2.50)$$

gdzie I oraz J są wymiarami obrazu, zaś Δx oraz Δy są odległościami pomiędzy kolejnymi pikselami, odpowiednio w kierunku poziomym i pionowym. Na ogół przyjmuje się, że $\Delta x=1$ oraz $\Delta y=1$. W tym przypadku rozwiązania równań (2.48 oraz 2.49) przyjmują postać równań różnicowych:

$$u_{l+1} = u_l - \Delta t D_{xx}(\alpha_e(u_l) \cdot D(u_l)) - \Delta t D_{yy}(\beta_e(u_l) \cdot D_{yy}(u_l)) - \Delta t \lambda_l (u_l - u_0) \quad (2.51)$$

$$\lambda_l = -\frac{1}{\sigma^2} \sum_{\Omega} [(\alpha_e(u_l) \cdot D_{xx}(u_l)) \cdot D_{xx}(u_l - u_0) - (\beta_e(u_l) \cdot D_{yy}(u_l)) \cdot D_{yy}(u_l - u_0)] \Delta x \Delta y \quad (2.52)$$

gdzie u_0 jest obrazem początkowym,

$$D_{xx}(u(x, y)) = \frac{u(x-1, y) - 2 \cdot u(x, y) + u(x+1, y)}{(\Delta x)^2},$$

$$D_{yy}(u(x, y)) = \frac{u(x, y-1) - 2 \cdot u(x, y) + u(x, y+1)}{(\Delta y)^2},$$

$$\alpha_e = \frac{1}{|D_{xx}(u(x, y))| + e}, \quad \beta_e = \frac{1}{|D_{yy}(u(x, y))| + e},$$

a stała $e=10^{-10}$ i zapobiega dzieleniu przez zero.

2.5.4. Filtracja przestrzenno-częstotliwościowa

Filtracja przestrzenno-częstotliwościowa polega na podzieleniu obrazu na kwadratowe bloki (na ogół o wymiarach 8x8 pikseli), a następnie wyznaczeniu 2D DFT dla każdego z nich. Tłumienie szumu jest uzyskiwane za pomocą filtra, który zeruje współczynniki transformacji odpowiadające wyższym częstotliwościom. Następnie, dla zmodyfikowanych współczynników jest liczona transformacja odwrotna. Metoda ta jest złożona obliczeniowo oraz może powodować powstawanie zniekształceń (artefaktów) na przefiltrowanym obrazie.

2.5.5. Filtracja w dziedzinie współczynników transformacji falkowej

Filtracja za pomocą transformacji falkowej może być podzielona na liniową oraz nieliniową.

2.5.5.1. Filtracja liniowa

W tym przypadku filtracja polega na zastosowaniu liniowego filtra Wienera w przestrzeni transformacji falkowej [Stre00]. Ten rodzaj filtracji daje najbardziej optymalne wyniki, gdy szum ma rozkład Gaussa, zaś kryterium jej jakości stanowi wielkość błędu średnio-kwadratowego [Choi98].

2.5.5.2. Filtracja nieliniowa

Jest to najczęściej spotykana odmiana filtracji za pomocą transformacji falkowej. Wykorzystuje ona tę właściwość transformacji falkowej, że znaczna część energii użytecznego sygnału zawiera się w niewielkiej ilości współczynników oraz fakt, iż szum biały na obrazie jest odwzorowywany na szum biały w dziedzinie współczynników transformacji falkowej (szum jest równomiernie rozłożony na wszystkie współczynniki) [Dono95a]. Tak więc energia użytecznego sygnału zostaje zawarta w kilku współczynnikach, zaś energia szumu – nie. W efekcie współczynniki transformacji odpowiadające sygnałowi użytecznemu mają znacznie większe wartości niż współczynniki odpowiadające szumowi. Dlatego metoda ta jest oparta o założenie, że sygnał użyteczny można odróżnić od szumu jedynie na podstawie wartości współczynników transformacji falkowej. W tym celu najczęściej stosuje się miękkie lub twarde progowanie (*Soft or Hard Thresholding*). Operator twardego progowania ma następującą postać:

$$HT(X, \gamma) = \begin{cases} X & \text{dla } |X| \geq \gamma \\ 0 & \text{dla } |X| < \gamma \end{cases} \quad (2.53)$$

gdzie X jest wartością wejściową, zaś γ jest wartością progu. Natomiast operator miękkiego progowania dany jest następująco [Star99]:

$$ST(X, \gamma) = \begin{cases} \text{sgn}(X) \cdot (|X| - \gamma) & \text{dla } |X| \geq \gamma \\ 0 & \text{dla } |X| < \gamma \end{cases} \quad (2.54)$$

W wyniku progowania współczynniki transformacji o wartościach poniżej przyjętego progu zostają wyzerowane (twarde progowanie) lub znacząco zmniejszone (miękkie progowanie).

Ostatecznie algorytm odsumiania za pomocą transformacji falkowej ma postać przedstawioną w tabeli (2.3) [Dono94]:

Tabela 2.3. Odszumianie za pomocą transformacji falkowej

1. oblicz transformację falkową zaszumianego sygnału; w wyniku uzyskuje się współczynniki aproksymacji $a_{j,k}$, oraz detali $d_{j,k}$;
2. dokonaj progowania współczynników detali za pomocą twardego lub miękkiego progowania (2.53, 2.54);
3. oblicz odwrotną transformację falkową na podstawie współczynników aproksymacji oraz zmodyfikowanych współczynników detali.

Chociaż twarde progowanie może lepiej tłumić szum niż miękkie, to jednak progowanie miękkie rzadziej powoduje powstawanie na przefiltrowanym obrazie artefaktów. Ponadto twarde progowanie nie może być wykorzystane we wszystkich algorytmach odsumiania [Jans01].

Opracowano różne metody automatycznego wyznaczania progu, na przykład:

- nieadaptacyjna metoda VISUShrink [Dono94, Dono95a] – wartość progu zależy od liczby pikseli na obrazie;
- metody adaptacyjne: SUREShrink [Dono95b], BayesShrink [Chan00a], CrossValidation [Jans99, Naso96], filtracja Wienera w dziedzinie transformacji falkowej [Star94, Port01].

Założenie, iż można rozróżnić sygnał od szumu jedynie na podstawie wartości współczynników transformacji falkowej, nie jest spełnione, gdy energia szumu jest większa od energii sygnału. W takim przypadku należy brać pod uwagę przestrzenny rozkład wartości współczynników. Współczynniki odpowiadające użytecznemu sygnałowi rozkładają się na ogół wzdłuż linii o określonym kształcie (np. linie proste lub krzywe), zaś współczynniki odpowiadające szumowi są na ogół rozłożone równomiernie.

Oprócz prostego progowania współczynników transformacji falkowej, zastosowano w niniejszej pracy udoskonaloną metodę filtracji za pomocą transformacji falkowej (*Neighshrink*). W celu optymalnego doboru wartości progu wykorzystuje ona współczynniki transformacji sąsiadujące z bieżącym współczynnikiem. Metoda *Neighshrink* działa w sposób następujący [Chen04]: Niech $d_{j,k}$ oznacza bieżący współczynnik, zaś $B_{j,k}$ jest oknem wokół współczynnika $d_{j,k}$:

$$S_{j,k}^2 = \sum_{(i,l) \in B_{j,k}} d_{i,l}^2 \quad (2.55)$$

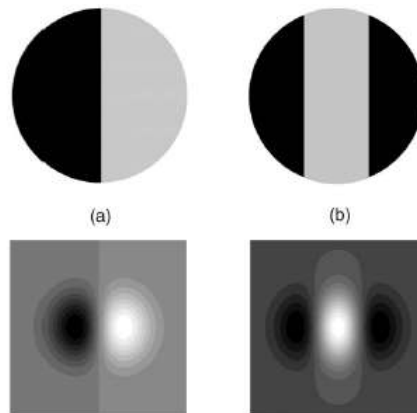
Współczynnik $d_{j,k}$ jest modyfikowany zgodnie z następującą zależnością:

$$\tilde{d}_{j,k} = d_{j,k} \cdot \left(1 - \frac{k \cdot \sigma^2 \cdot \log(n^2)}{S_{j,k}^2} \right)_+ \quad (2.56)$$

gdzie, σ jest wariancją szumu, n jest liczbą pikseli w oknie, zaś znak „+” oznacza, że zachowane zostają jedynie dodatnie wartości wyrażenia w nawiasie, zaś ujemnym zostaje przyporządkowana wartość 0. Parametr k jest dobierany doświadczalnie.

2.6. Filtracja za pomocą przestrajanych filtrów kierunkowych

Załóżmy, że zadaniem jest detekcja pewnych wzorców na obrazie $f(x, y)$ [Free91]. Wzorce te mogą występować w dowolnym miejscu obrazu oraz posiadać dowolną orientację (rys. 2.26). Wzorcami tymi mogą być np. krawędzie obiektów lub linie proste na obrazie. Ich detekcji można dokonać za pomocą tzw. filtrów kierunkowych, tj. filtrów, które wzmacniają cechy obrazu posiadające orientację zgodną z orientacją filtra. Procedura detekcji będzie polegać na wyliczeniu w każdym punkcie obrazu jego splotu z przesuniętymi i obróconymi wersjami wzorca $f_0(x, y) = h(-x, -y)$. Duża wartość splotu oznacza obecność wzorca w danym miejscu obrazu.



Rys. 2.26. Przykładowe wzorce oraz odpowiadające im filtry [Jaco04]

Matematycznie operację tę można zapisać następująco [Jaco04]:

$$\begin{aligned}\theta^*(\mathbf{x}) &= \arg \max_{\theta} (f(\mathbf{x}) * h(\mathbf{R}_{\theta} \mathbf{x})) \\ r^*(\mathbf{x}) &= f(\mathbf{x}) * h(\mathbf{R}_{\theta} \mathbf{x})\end{aligned}\quad (2.57)$$

gdzie r^* jest wartością splotu w punkcie $\mathbf{x}=(x, y)$, zaś θ^* jest orientacją wzorca. Macierz \mathbf{R}_{θ} jest macierzą rotacji:

$$\mathbf{R}_{\theta} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}\quad (2.58)$$

Jednak bezpośrednio stosowanie równań (2.57, 2.58) jest niepraktyczne z uwagi na konieczność implementacji dużej liczby filtrów (im mniejsza kwantyzacja kąta, tym większa liczba filtrów). Dlatego w celu zmniejszenia złożoności obliczeniowej stosuje się specjalną odmianę wzorców, zwaną przestrajanymi filtrami kierunkowymi. Filtr taki może być bardzo efektywnie obracany za pomocą liniowej kombinacji kilku filtrów składowych. W tym przypadku wzorce mają postać:

$$h(x, y) = \sum_{k=1}^M \sum_{i=0}^k \alpha_{k,i} \frac{\partial^{k-i}}{\partial x^{k-i}} \frac{\partial^i}{\partial y^i} g(x, y)\quad (2.59)$$

gdzie $g(x, y)$ jest dowolną izotropową funkcją okna.

Jako funkcję $g(x, y)$ najczęściej stosuje się funkcję Gaussa. Filtr postaci (2.59) jest nazywany detektorem M -tego rzędu i jest on sterowalny, to znaczy splot sygnału $f(x, y)$ z dowolnie obróconą wersją filtra $h(x, y)$ może być wyrażony następująco:

$$f(\mathbf{x}) * h(\mathbf{R}_{\theta} \mathbf{x}) = \sum_{k=1}^M \sum_{i=0}^k b_{k,i}(\theta) f_{k,i}(\mathbf{x})\quad (2.60)$$

gdzie funkcje $f_{k,i}(x, y)$ są przefiltrowanymi wersjami sygnału $f(x, y)$. Funkcje $f_{k,i}(x, y)$ oblicza się następująco:

$$f_{k,i}(x, y) = f(x, y) * \underbrace{\left(\frac{\partial^{k-i}}{\partial x^{k-i}} \frac{\partial^i}{\partial y^i} g(x, y) \right)}_{g_{k,i}(x, y)}\quad (2.61)$$

Wagi $b_{k,i}(\theta)$, zależne od orientacji filtra, wyrażone są następująco:

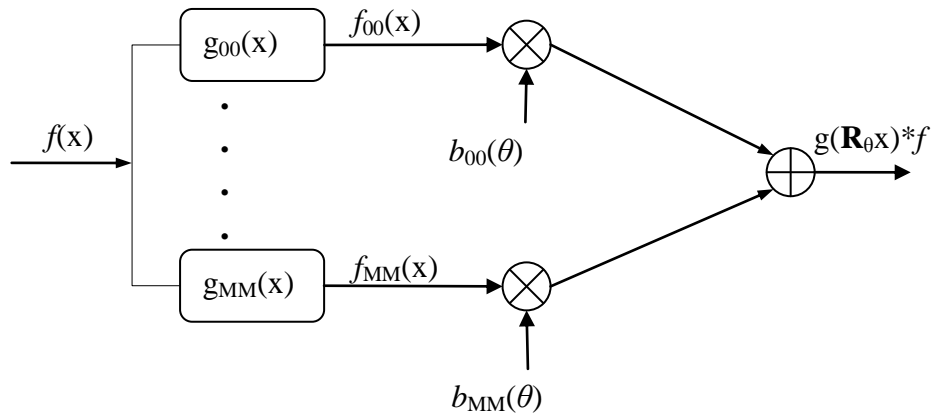
$$b_{k,i}(\theta) = \left(\sum_{j=0}^k \alpha_{k,i} \sum_{l,m \in S(k,j,i)} \binom{k-j}{l} \binom{j}{m} (-1)^m \cos(\theta)^{j+(l-m)} \sin(\theta)^{(k-j)-(l-m)} \right)\quad (2.62)$$

gdzie $S(k, j, i)$ jest zbiorem i $S(k, j, i) = \{l, m | 0 \leq l \leq k-i; 0 \leq m \leq i; k-(l-m) = j\}$.

Graficzna reprezentacja równań (2.60 oraz 2.61) została przedstawiona na rys (2.27).

Gdy znane są wartości $f_{k,i}(x, y)$, wtedy splot $f(\mathbf{x}) * h(\mathbf{R}_{\theta})$ może być bardzo szybko wyznaczony poprzez ważoną sumę współczynników, będącą trygonometrycznym wielomianem zmiennej θ .

W niniejszej pracy przestrajane filtry kierunkowe były wykorzystane na etapie poszukiwania ewentualnych brakujących linii do filtracji kierunkowej obrazu.



Rys. 2.27. Schemat filtracji za pomocą przestrzajanego filtra kierunkowego [Jaco04]

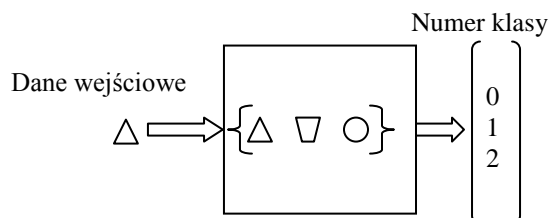
2.7. Podsumowanie

Wszystkie opisane w niniejszym rozdziale metody analizy i przetwarzania obrazów wykorzystano w części eksperymentalnej pracy, mającej na celu detekcję linii Kikuchiego w obrazach mikroskopowych struktur krystalicznych. W szczególności wykorzystano:

- 1) w metodzie opisanej w rozdziale 5., transformację Hougha do detekcji linii,
- 2) transformację Radona, w szybkiej metodzie wyznaczania transformacji Hougha, zastosowanej w rozdziale 6.,
- 3) transformację falkową do odsumowania obrazów (rozd. 6),
- 4) transformację *ridgelet* i *curvelet* do zwiększania kontrastu obrazu (rozd. 6),
- 5) filtry kierunkowe do filtracji obrazu (rozd. 6).

3. Metody klasyfikacji stosowane w pracy

W niniejszym rozdziale przedstawiono klasyfikatory stosowane w części eksperymentalnej pracy. Jeżeli założymy, że sygnały wejściowe przynależą do kilku klas, to w odpowiedzi na sygnał wejściowy klasyfikator powinien wskazać klasę, do której ten sygnał należy. Istota działania klasyfikatora rozróżniającego trzy klasy pokazana jest na rysunku (3.1). Ogólnie metody klasyfikacji można podzielić na dwie grupy: klasyfikacja nienadzorowana (*unsupervised*) oraz nadzorowana (*supervised*).



Rys. 3.1. Przykład klasyfikacji

3.1. Algorytmy nienadzorowane

Klasyfikacja nienadzorowana, zwana również grupowaniem (*clustering*), polega na podziale grupy danych przy założeniu, że nic nie wiemy o ich strukturze. W wyniku klasyfikacji każda z utworzonych grup zawiera elementy, które posiadają określone własności charakterystyczne dla danej grupy. Grupowanie jest dokonywane na podstawie pewnego kryterium, którym na ogół jest funkcja podlegająca minimalizacji. Przykładami algorytmów nienadzorowanych mogą być: algorytm k -średnich, algorytm rozmytych k -średnich, algorytm Gustafsona-Kessla.

3.1.1. Algorytm k -średnich

Celem algorytmu jest przypisanie do odpowiednich grup (klastrow) M n -wymiarowych wektorów danych przy jak najmniejszym średnim błędzie. Każdy klastrow ma swój środek (centroid) określony przez wektor \mathbf{v}_i . Istotą działania tej metody jest minimalizacja średniego błędu, obliczanego następująco:

$$D^2 = \frac{1}{K} \sum_{i=1}^K \sum_{j \in A_i} d^2(\mathbf{x}_j, \mathbf{v}_i) \quad (3.1)$$

gdzie A_i jest zbiorem danych należących do i -tego klastra, zaś \mathbf{v}_i jest jego środkiem. Wyrażenie $d^2(\mathbf{x}_j, \mathbf{v}_i) = \|\mathbf{x}_j - \mathbf{v}_i\|^2$ jest wybraną miarą odległości pomiędzy kolejną daną a

środkiem i -tego klastra. Minimalizacja błędu D jest dokonywana za pomocą algorytmu przedstawionego w tabeli (3.1). Algorytm ten dokonuje wstępnego podziału danych na z góry ustaloną liczbę klas, a następnie podział ten jest iteracyjnie poprawiany poprzez przenoszenie niektórych danych do innych klas, tak aby uzyskać minimalną wartość błędu D .

Tabela 3.1. Algorytm k -średnich [Blas1]

<p>1) Dla danego zbioru danych \mathbf{X} wybierz liczbę klastrów $1 < c < N$. Zainicjalizuj losowo położenie środków klastrów. Ustal maksymalny błąd D_{max}.</p> <p>2) Powtarzaj dla $l=1,2,\dots$</p> <p>a) Wyznacz odległości: $D_{ik}^2 = (\mathbf{x}_k - \mathbf{v}_i)^T (\mathbf{x}_k - \mathbf{v}_i)$, $1 \leq i \leq c$, $1 \leq k \leq N$</p> <p>b) Podziel M wektorów danych na c grup. Wektor \mathbf{x}_j ($j \in [1, M]$) jest przypisywany do i-tej grupy wtedy i tylko wtedy, gdy zachodzi nierówność $d(\mathbf{x}_j, \mathbf{v}_i) \leq d(\mathbf{x}_j, \mathbf{v}_k)$ dla wszystkich \mathbf{v}_k różnych od \mathbf{v}_i.</p> <p>c) Wyznacz położenie środków dla każdej z grup. $\mathbf{v}_i^{(l)} = \frac{\sum_{k=1}^{N_i} \mathbf{x}_k}{N_i}$, $\mathbf{x}_k \in A_i$, gdzie N_i jest liczbą danych przypisanych do klastra A_i.</p> <p>d) dopóki $\frac{D_{m-1} - D_m}{D_m} \geq D_{max}$</p>
--

3.1.2. Algorytm rozmytych k -średnich

W algorytmie rozmytych k -średnich (*fuzzy c-means*) pojedyncza dana może należeć do dwóch lub więcej klastrów. Możliwe jest to dzięki temu, że przynależność do danego klastra jest wyrażona przez dowolną liczbę z przedziału $[0;1]$, gdzie 0 odpowiada całkowity brak przynależności, zaś 1 odpowiada całkowita przynależność do klastra. Algorytm jest oparty na minimalizacji następującej funkcji celu:

$$J_m = \sum_{i=1}^N \sum_{j=1}^C u_{ij}^m \|\mathbf{x}_i - \mathbf{v}_j\|^2 \quad 1 \leq m \leq \infty \quad (3.2)$$

gdzie m jest dodatnią liczbą większą od 1, zaś u_{ij} jest stopniem przynależności wektora \mathbf{x}_i do klastra j . Wektor \mathbf{x}_i zawiera i -ty pomiar, zaś \mathbf{v}_j jest środkiem j -tego klastra. Algorytm dokonuje iteracyjnej optymalizacji funkcji celu, uaktualniając przy każdej iteracji środki klastrów \mathbf{v}_j oraz stopnie przynależności u_{ij} . Wielkości te oblicza się następująco:

$$u_{ij} = \frac{1}{\sum_{k=1}^C \left(\frac{\|\mathbf{x}_i - \mathbf{v}_j\|}{\|\mathbf{x}_i - \mathbf{v}_k\|} \right)^{\frac{2}{m-1}}}, \quad \mathbf{v}_j = \frac{\sum_{i=1}^N u_{ij}^m \cdot \mathbf{x}_i}{\sum_{i=1}^N u_{ij}^m} \quad (3.3)$$

Algorytm kończy działanie, gdy $\|\mathbf{U}^{(l)} - \mathbf{U}^{(l-1)}\| < \varepsilon$, gdzie ε jest liczbą z przedziału $[0;1]$, k jest kolejną iteracją, zaś \mathbf{U} jest macierzą współczynników u_{ij} . Cały algorytm jest przedstawiony w tabeli 3.2.

Tabela 3.2. Algorytm rozmytych k -średnich [Bez81]

<p>1. Dla danego zbioru danych \mathbf{X}, wybierz liczbę klastrów $1 < c < N$, wykładnik wazący $m > 1$, warunek zakończenia $\varepsilon > 0$ oraz macierz \mathbf{A}. Losowo inicjuj macierz podziału $\mathbf{U}^{(0)}$.</p> <p>2. Powtarzaj dla $l=0,1,\dots$</p> <p>a) wyznacz środki klastrów $\mathbf{v}_i^{(l)} = \frac{\sum_{k=1}^N (u_{ik}^{(l-1)})^m \mathbf{x}_k}{\sum_{k=1}^N (u_{i,k}^{(l-1)})^m}$, $1 \leq i \leq c$</p> <p>b) wyznacz odległości $D_{ikA}^2 = (\mathbf{x}_k - \mathbf{v}_i)^T \mathbf{A} (\mathbf{x}_k - \mathbf{v}_i)$, $1 \leq i \leq c$, $1 \leq k \leq N$</p> <p>c) uaktualnij macierz podziału $\mu_{i,k}^{(l)} = \frac{1}{\sum_{j=1}^c (D_{ikA} / D_{jkA})^{\frac{2}{m-1}}}$</p> <p>3. dopóki $\ \mathbf{U}^{(l)} - \mathbf{U}^{(l-1)}\ < \varepsilon$</p>

3.1.3. Algorytm Gustafsona-Kessla

Algorytm Gustafsona-Kessla stanowi rozszerzenie algorytmu rozmytych k -średnich. Modyfikacja polega na zastosowaniu adaptacyjnej normy odległości w celu detekcji klastrów o różnych kształtach w jednym zbiorze danych. Każdy klaster ma swoją normę zawierającą macierz \mathbf{A}_i . Norma ta obliczana jest następująco:

$$d_{ikA_i}^2(\mathbf{x}_i, \mathbf{v}_j) = (\mathbf{x}_i - \mathbf{v}_j)^T \cdot \mathbf{A}_i \cdot (\mathbf{x}_i - \mathbf{v}_j) \quad (3.4)$$

Macierze \mathbf{A}_i są macierzami optymalizacyjnymi, dzięki czemu możliwe jest dostosowanie normy do lokalnej struktury topologicznej danych.

W przypadku algorytmu Gustafsona-Kessla funkcja celu ma postać:

$$J_m = \sum_{i=1}^N \sum_{j=1}^c u_{ij}^m d_{ikA_i}^2(\mathbf{x}_i, \mathbf{v}_j) \quad 1 \leq m \leq \infty \quad (3.5)$$

Funkcja ta nie może być bezpośrednio zminimalizowana względem macierzy \mathbf{A}_i , ponieważ jest względem niej liniowa. To zaś oznacza, że J_m może być zminimalizowana tylko przez uczynienie macierzy \mathbf{A}_i w jak najmniejszym stopniu dodatnio określoną. Aby to uzyskać, macierz ta musi być w jakiś sposób ograniczona, co najczęściej uzyskuje się przez ograniczenie jej wyznacznika. Zmianom macierzy \mathbf{A}_i przy stałym wyznaczniku odpowiada optymalizacja kształtu klastra przy jego stałej objętości:

$$\|\mathbf{A}_i\| = \rho_i \quad \rho_i > 0 \quad (3.6)$$

gdzie ρ_i jest ustalone dla każdego klastra. Stosując metodę mnożników Lagrange'a uzyskuje się:

$$\mathbf{A}_i = [\rho_i \det(\mathbf{F}_i)]^{\frac{1}{n}} \cdot \mathbf{F}_i^{-1} \quad (3.7)$$

gdzie \mathbf{F}_i jest macierzą kowariancji dla i -tego klastra. Macierz ta wyznaczana jest następująco:

$$\mathbf{F}_i = \frac{\sum_{k=1}^N (u_{ik})^m (\mathbf{x}_k - \mathbf{v}_i)(\mathbf{x}_k - \mathbf{v}_i)^T}{\sum_{k=1}^N (u_{ik})^m} \quad (3.8)$$

Tabela 3.3. Algorytm Gustafsona-Kessla [Babus02]

1. Dla danego zbioru danych \mathbf{X} wybierz liczbę klastrów $1 < c < N$, wykładnik wazący $m > 1$, warunek zakończenia $\varepsilon > 0$ oraz macierz \mathbf{A} . Losowo inicjuj macierz podziału $\mathbf{U}^{(0)}$.

2. **Powtarzaj** dla $l = 1, 2, 3, \dots$:

a) Wyznacz środki klastrów $\mathbf{v}_i^{(l)} = \frac{\sum_{k=1}^N (u_{ik}^{(l-1)})^m \mathbf{x}_k}{\sum_{k=1}^N (u_{ik}^{(l-1)})^m}, 1 \leq i \leq c$ (3.9)

b) Wyznacz macierze kowariancji dla klastrów

$$\mathbf{F}_i^{(l)} = \frac{\sum_{k=1}^N (u_{ik}^{(l-1)})^m (\mathbf{x}_k - \mathbf{v}_i^{(l)})(\mathbf{x}_k - \mathbf{v}_i^{(l)})^T}{\sum_{k=1}^N (u_{ik}^{(l-1)})^m}, \quad 1 \leq i \leq c \quad (3.10)$$

Dodaj przeskalowaną macierz identycznościową:

$$\mathbf{F}_i := (1 - \gamma)\mathbf{F}_i + \gamma(\mathbf{F}_0)^{\frac{1}{n}} \mathbf{I} \quad (3.11)$$

Wyznacz wartości własne λ_{ij} oraz wektory własne φ_{ij} ,

znajdź $\lambda_{i,\max} = \max_j \lambda_{ij}$ oraz zbiór

$$\lambda_{i,\max} = \frac{\lambda_{ij}}{\beta} \quad \forall j \quad \text{dla których} \quad \frac{\lambda_{i,\max}}{\lambda_{ij}} \geq \beta \quad (3.12)$$

Odtwórz macierz \mathbf{F}_i :

$$\mathbf{F}_i = [\varphi_{i,1} \dots \varphi_{i,n}] \text{diag}(\lambda_{i,1} \dots \lambda_{i,n}) [\varphi_{i,1} \dots \varphi_{i,n}]^{-1} \quad (3.13)$$

c) Wyznacz odległości

$$d_{ikA_i}^2(\mathbf{x}_k, \mathbf{v}_i) = (\mathbf{x}_k - \mathbf{v}_i^{(l)})^T \left[(\rho_i \det(\mathbf{F}_i))^{\frac{1}{n}} \mathbf{F}_i^{-1} \right] (\mathbf{x}_k - \mathbf{v}_i^{(l)}) \quad (3.14)$$

d) uaktualnij macierz podziału $u_{ik}^{(l)} = \frac{1}{\sum_{j=1}^c \left(\frac{d_{ikA_i}(\mathbf{x}_k, \mathbf{v}_i)}{d_{jk}(\mathbf{x}_k, \mathbf{v}_j)} \right)^{\frac{2}{m-1}}}, 1 \leq i \leq c, 1 \leq k \leq N$ (3.15)

3. **dopóki** $\|\mathbf{U}^{(l)} - \mathbf{U}^{(l-1)}\| < \varepsilon$

3.2. Algorytmy nadzorowane

Z klasyfikacją nadzorowaną, lub inaczej analizą dyskryminacyjną mamy do czynienia wtedy, gdy znana jest struktura danych, tj. wtedy, gdy dysponujemy choćby częściową charakterystyką klas, z których pochodzą obiekty. Istota działania algorytmów nadzorowanych polega na uczeniu się w oparciu o zbiór uczący. Nauczony klasyfikator dokonuje klasyfikacji nowych danych na podstawie wiedzy zdobytej w procesie uczenia. Przykładami tych algorytmów mogą być: algorytm k -najbliższych sąsiadów, algorytm rozmytych k -najbliższych sąsiadów, algorytm MMD (*Mean Minimum Distance*), metoda wektorów nośnych oraz sieci neuronowe.

3.2.1. Algorytm k -najbliższych sąsiadów

W tabeli (3.4) przedstawiono algorytm najbliższego sąsiada. Różnica pomiędzy algorytmem najbliższego sąsiada, a algorytmem k -najbliższych sąsiadów polega na tym, że do grupy dołączamy nie tylko najbliższego sąsiada rozpatrywanego elementu, lecz jego k sąsiadów. Zaletą tego algorytmu w porównaniu z algorytmem najbliższego sąsiada są dobre wyniki uzyskiwane dla grup o nierównomiernym rozkładzie gęstości.

Tabela 3.4. Algorytm najbliższego sąsiada [Poli1]

- 1) Przyjmij wartość progową.
- 2) Oblicz odległości kolejnych elementów od ich najbliższych sąsiadów.
- 3) Połącz w jedną grupę elementy, których odległość jest mniejsza od ustalonej wartości progowej.

3.2.2. Algorytm rozmytych k -najbliższych sąsiadów

Algorytm rozmytych k -najbliższych sąsiadów należy do grupy niehierarchicznych algorytmów klasyfikacji [Poli1]. Jego istotą jest początkowy losowy wybór położenia środków grup. Po obliczeniu funkcji przynależności każdego z punktów do wszystkich grup, w kolejnych iteracjach funkcje te są każdorazowo przeliczane. Takie postępowanie powoduje, że środki grup „wędrują” do swoich prawidłowych położenia. Środki poszczególnych grup organizujących przestrzeń obliczane są według zależności:

$$\underline{\mu}_j = \frac{\sum_{j=1}^n P(\omega_i | \underline{x}_j)^b \underline{x}_j}{\left(\sum_{j=1}^n P(\omega_i | \underline{x}_j) \right)^b} \quad (3.16)$$

gdzie $P(\omega_i | \underline{x}_j)$ oznacza prawdopodobieństwo warunkowe przynależności j -go elementu do i -tej grupy, natomiast b jest parametrem, którego wartość musi być różna od 1 (najczęściej przyjmuje się, że b jest równe 2). Funkcja przynależności jest normalizowana następująco:

$$\sum_{i=1}^c P(\omega_i | \underline{x}_j) = 1, \text{ gdzie } j = 1, \dots, n \quad (3.17)$$

Przynależność elementu do każdej z grup $P(\omega_i | \underline{x}_j)$ obliczana jest w następujący sposób:

$$P(\omega_i | \underline{x}_j) = \frac{\left(\frac{1}{d_{ij}}\right)^{\frac{1}{b-1}}}{\sum_{r=1}^c \left(\frac{1}{d_{rj}}\right)^{\frac{1}{b-1}}} \quad (3.18)$$

gdzie $d_{ij}^2 = \|\underline{x}_j - \underline{\mu}_i\|^2$ jest odległością punktu \underline{x}_j od środka grupy $\underline{\mu}_i$, natomiast b jest parametrem, którego wartość musi być różna od 1 (najczęściej przyjmuje się, że b jest równe 2).

Tabela 3.5. Algorytm rozmytych k -najbliższych sąsiadów [Poli1]

1. Inicjalizacja:
 - a) wyznacz losowo położenie środków poszukiwanych grup,
 - b) dla każdego elementu oblicz wartości funkcji przynależności do wszystkich grup.
2. Oblicz odległości punktów od środków grup.
3. Oblicz $P(\omega_i | \underline{x}_j)$.
4. Ponownie oblicz środki grup $\underline{\mu}_i$.
5. Jeżeli:
 - a) brak zmian w $\underline{\mu}_i$ oraz $P(\omega_i | \underline{x}_j)$, zwróć $\underline{\mu}_1, \dots, \underline{\mu}_c$,
 - b) w przeciwnym wypadku wróć do punktu 2.

Poza klasycznym algorytmem iteracyjnym wykorzystującym zbiory rozmyte, możliwa jest taka adaptacja niektórych algorytmów opartych na zbiorach klasycznych, aby działały skutecznie w oparciu o zbiory przybliżone. Możliwe jest także opracowanie algorytmów grupowania łączących w sobie więcej niż jeden z klasycznych algorytmów. Przykładem takiego algorytmu grupowania opartego na zbiorach klasycznych, który łatwo daje się zaadoptować, jest algorytm najbliższego sąsiada.

3.2.3. Algorytm MMD

Algorytm MMD jest w swej istocie bardzo podobny do algorytmu najbliższego sąsiada, z tą różnicą, że nie jest on opisany jako algorytm bazujący na teorii grafów.

Tabela 3.6. Algorytm MMD [Koon76]

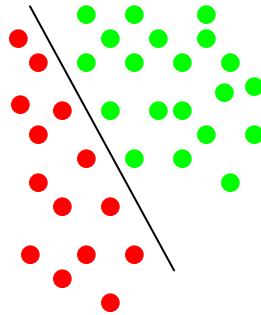
1. Oblicz odległości pomiędzy najbliższymi sąsiadami.
2. Wyznacz wartość średniej d_{mean} wyznaczonych odległości.
3. Usuń szum, czyli wszystkie elementy, których odległości od najbliższych sąsiadów są większe niż $k d_{mean}$ (najczęściej przyjmuje się, że k wynosi 2).
4. Ponownie oblicz wartości d_{mean} dla tak utworzonego zbioru.
5. Połącz w grupy najbliższych sąsiadów, jeżeli odległości między nimi są mniejsze niż $k d_{mean}$.

3.3. Metoda wektorów nośnych

U podstaw metody wektorów nośnych SVM (*Support Vector Machines*) leży koncepcja przestrzeni decyzyjnej, którą dzieli się budując granice separujące obiekty o różnej przynależności klasowej [Cris00]. Metoda ta pozwala na klasyfikację danych w przypadku, gdy [Gunn98]:

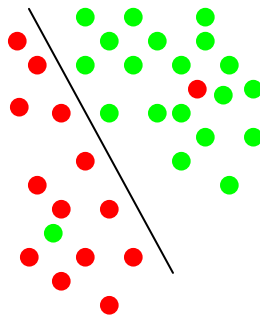
- można je rozdzielić za pomocą jednej prostej (liniowa separowalność),
- nie można ich rozdzielić za pomocą jednej prostej (nieseparowalność liniowa),
- do rozdzielenia danych konieczne jest użycie krzywej (nieliniowa separowalność).

Na rysunku (3.2) przedstawiono przypadek liniowo separowalny. Mamy tu dwie klasy kółek: zielone i czerwone. Linia graniczna rozdziela je wyraźnie. Nowy, nieznanый obiekt, jeżeli znajdzie się po prawej stronie granicy, zostanie zaklasyfikowany jako zielony, a w przeciwnym wypadku, jako czerwony.



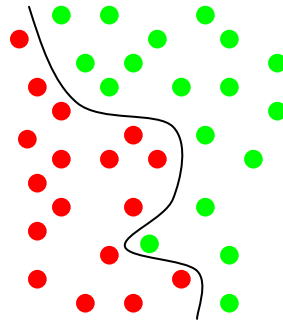
Rys. 3.2. Przypadek liniowo separowany [Inte1]

Powyższy rysunek jest ilustracją bardzo prostego przykładu klasyfikatora liniowego, dzielącego obszar prób na dwie części za pomocą prostej. Większość praktycznych zadań klasyfikacyjnych nie jest jednak tak oczywista. Może się zdarzyć, że pojedyncze elementy jednego zbioru znajdują się w obszarze drugiego. Wtedy mamy do czynienia z przypadkiem nieseparowalnym liniowo.



Rys. 3.3. Nieseparowalność liniowa

Może się także zdarzyć, że do poprawnego klasyfikowania potrzebne są bardziej skomplikowane struktury niż linia prosta (klasyfikacja nieliniowa). Przykładem może być rysunek (3.4), który jasno wskazuje, że do rozdzielenia zielonych i czerwonych punktów konieczna jest teraz krzywa (obiekt bardziej skomplikowany niż prosta). Krzywa ta, ale również poprzednie proste, są przykładami klasyfikatorów hiperpowierzchniowych.



Rys. 3.4. Klasyfikacja nieliniowa [Intel]

Metoda wektorów nośnych realizuje zadania klasyfikacyjne, konstruując w wielowymiarowej przestrzeni hiperpłaszczyznę oddzielającą przypadki należące do różnych klas. Optymalną hiperpłaszczyznę separującą buduje się w iteracyjnym algorytmie uczącym, minimalizującym pewną funkcję błędu. SVM jest jedną z nowszych metod klasyfikacji danych. Jest to metoda nadzorowanego uczenia, to znaczy wymaga podania *explicite*, które elementy zbioru uczącego należą do której klasy.

3.3.1. Przypadek liniowo separowalny

Rozważmy przypadek, gdy dane należą do \mathbf{R}^2 oraz są reprezentantami dwóch klas: A i B . Zakładamy, że A oraz B są liniowo separowalne, tzn. istnieje taka linia, że wszystkie punkty należące do klasy A leżą po jednej stronie tej linii, a wszystkie punkty należące do klasy B leżą po jej drugiej stronie. Linia ta jest dana następująco:

$$H = \{\mathbf{x} \in \mathbf{R}^2 : \omega\mathbf{x} + b = 0\} \quad (3.19)$$

gdzie $\omega \in \mathbf{R}^2$, zaś b jest skalar.

Matematycznie warunek liniowej separowalności można wyrazić następująco:

$$\begin{aligned} \mathbf{x}_i \in A \quad \text{dla} \quad \omega\mathbf{x}_i + b > 0 \\ \mathbf{x}_i \in B \quad \text{dla} \quad \omega\mathbf{x}_i + b < 0 \end{aligned} \quad (3.20)$$

W przypadku zbiorów liniowo separowalnych celem algorytmu SVM jest znalezienie takiego wektora $\omega \in \mathbf{R}^2$ oraz skalar $b \in \mathbf{R}$, że

$$\begin{aligned} \omega\mathbf{x}_i + b \geq 1 \quad \text{dla} \quad \mathbf{x}_i \in A \\ \omega\mathbf{x}_i + b \leq -1 \quad \text{dla} \quad \mathbf{x}_i \in B \end{aligned} \quad (3.21)$$

Równości $\omega\mathbf{x}_i + b = 1$ oraz $\omega\mathbf{x}_i + b = -1$ definiują dwie linie, odpowiednio: H_1 oraz H_2 . Pomiedzy tymi liniami nie znajdują się żadne dane (rys. 3.5). Punkty leżące na liniach H_1 oraz H_2 zwane są wektorami nośnymi (*Support Vectors*). Separacja klas będzie najbardziej optymalna, gdy odległość między liniami H_1 oraz H_2 będzie maksymalna. Można udowodnić, że odległość ta wynosi $2/\|\omega\|$. Maksymalizacji tej odległości odpowiada minimalizacja wyrażenia $\|\omega\|^2/2$ (funkcja kosztu). Tak więc cały algorytm sprowadza się do zagadnienia programowania kwadratowego następującej postaci:

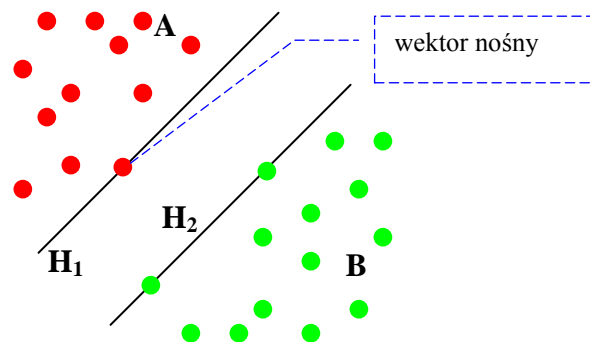
$$\text{znajdź } \min \left\{ \frac{1}{2} \|\omega\|^2 \right\}$$

$$\text{przy warunkach } \begin{cases} \omega \mathbf{x}_i + b \geq 1 & \text{dla } \mathbf{x}_i \in A \\ \omega \mathbf{x}_i + b \leq -1 & \text{dla } \mathbf{x}_i \in B \end{cases}$$

Zagadnienie to może zostać efektywnie rozwiązane numerycznie w wielu środowiskach obliczeniowych, np. w Matlabie. Najczęściej jest do tego stosowany algorytm sekwencyjnej minimalnej optymalizacji SMO (*Sequential Minimal Optimization*). Algorytm ten dzieli złożony przypadek na serię prostszych, rozwiązywanych analitycznie. Pozwala to na znaczne skrócenie czasu rozwiązywania. Ponadto ilość pamięci wymagana przez ten algorytm jest liniowo proporcjonalna do ilości danych.

Klasyfikacja nowej danej \mathbf{x}^* przebiega następująco:

$$\begin{aligned} \omega \mathbf{x}_i + b &\geq 1 & \text{dla } \mathbf{x}_i \in A \\ \omega \mathbf{x}_i + b &\leq -1 & \text{dla } \mathbf{x}_i \in B \end{aligned} \quad (3.22)$$



Rys. 3.5. Klasyfikacja za pomocą metody SVM

3.3.2. Przypadek nieseparowalny liniowo

W praktyce jest możliwe, iż z powodu błędów pomiaru lub innych przyczyn pewne elementy zbioru A znajdują się w obszarze zbioru B i *vice versa*. W tym przypadku oba zbiory nie są liniowo separowalne i dlatego konieczna jest modyfikacja zależności (3.21):

$$\begin{aligned} \omega \mathbf{x}_i + b &\geq 1 - \xi_i & \text{dla } \mathbf{x}_i \in A \\ \omega \mathbf{x}_i + b &\leq -1 + \xi_i & \text{dla } \mathbf{x}_i \in B \end{aligned} \quad (3.23)$$

gdzie ξ_i są stałymi skalarami, a $i=1,2,\dots,N$ (N jest liczbą danych uczących). Wielkość $\sum_{i=1}^N \xi_i$

jest granicznym błędem uczenia i musi być uwzględniona w funkcji kosztu. Ostatecznie problem minimalizacji przyjmuje postać:

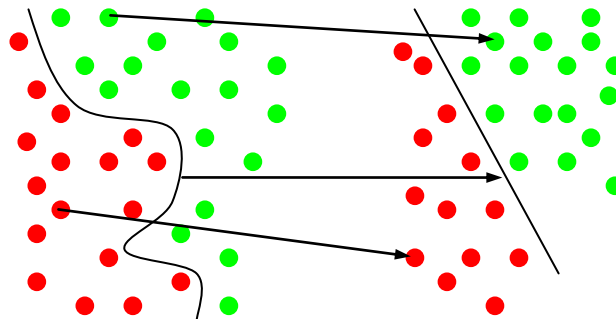
$$\text{znajdź } \min \left\{ \frac{1}{2} \|\omega\|^2 + C \left(\sum_{i=1}^N \xi_i \right)^k \right\}$$

$$\text{przy warunkach } \begin{cases} \omega \mathbf{x}_i + b \geq 1 - \xi_i & \text{dla } \mathbf{x}_i \in A \\ \omega \mathbf{x}_i + b \leq -1 + \xi_i & \text{dla } \mathbf{x}_i \in B \\ \xi_i \geq 0 \quad \forall i \end{cases}$$

gdzie C jest stałą, którą nazywa się pojemnością i która jest dobierana eksperymentalnie, zaś k jest liczbą całkowitą. Trzeba podkreślić, że C ma duży wpływ na błąd i jej wartość musi być ostrożnie dobierana, ze względu na niebezpieczeństwo nadmiernego dopasowania modelu mogące prowadzić do błędnej klasyfikacji nowych danych.

3.3.3. Klasyfikacja nieliniowa

Często zdarza się, że klasy A oraz B nie mogą być rozdzielone za pomocą płaszczyzny, ale mogą za pomocą np. okręgu, elipsy lub innej krzywej. Pojawia się zatem problem, jak uogólnić liniowy klasyfikator na możliwość separowania klas za pomocą dowolnej krzywej. Rozwiązaniem jest wprowadzenie dodatkowej funkcji jądrowej (*Kernel Function*), która odwzorowuje rzeczywiste dane na ich obrazy [Burg98]. Odwzorowanie to działa w ten sposób, że możliwe jest zastosowanie liniowego klasyfikatora na obrazach danych wejściowych.



Rys. 3.6. Odwzorowanie za pomocą funkcji jądrowej [Intel]

Oryginalne obiekty z lewej strony rysunku zostały rzutowane (transformowane) za pomocą funkcji jądrowej na przestrzeń ilustrowaną z prawej. Co ważne, w nowej przestrzeni dwie klasy są liniowo separowalne, co pozwala uniknąć skomplikowanej postaci granicy klas. Równania opisujące klasyfikator nieliniowy mają tę samą postać, jak równania klasyfikatora liniowego, z tą różnicą, że zamiast iloczynów skalarnych występuje w nich funkcja jądra.

Najczęściej spotykane funkcje jądrowe to:

- Wielomianowe (homogeniczne)

$$k(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y})^d \quad (3.24)$$

- Wielomianowe (niehomogeniczne)

$$k(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^d \quad (3.25)$$

- Funkcja radialna

$$k(\mathbf{x}, \mathbf{y}) = e^{-\gamma \|\mathbf{x} - \mathbf{y}\|^2} \quad \gamma > 0 \quad (3.26)$$

- Funkcja gaussowska

$$k(\mathbf{x}, \mathbf{y}) = e^{-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma^2}} \quad (3.27)$$

- Funkcja sigmoidalna

$$k(\mathbf{x}, \mathbf{y}) = \tanh(\kappa \mathbf{x} \cdot \mathbf{y} + c) \quad \kappa > 0, c < 0 \quad (3.28)$$

3.4. Sieci neuronowe

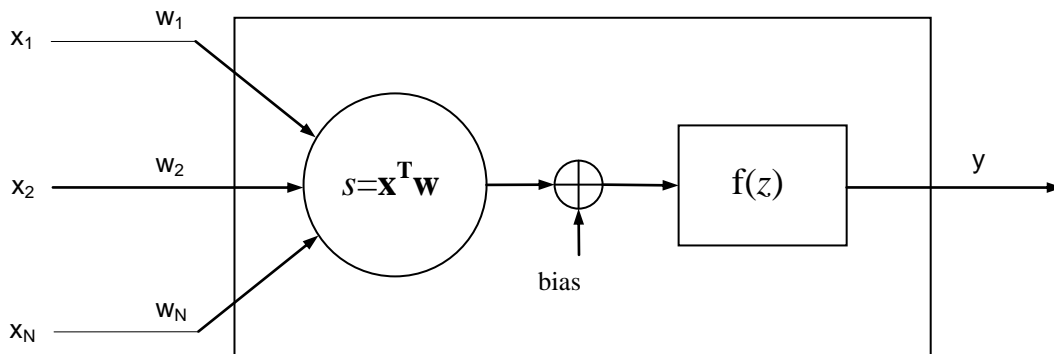
Do klasyfikacji danych wykorzystywano w niniejszej pracy także sieci neuronowe. Klasyfikacja jest jednym z najczęstszych ich zastosowań. Jeżeli założymy, że sygnały wejściowe przynależą do kilku klas, to w odpowiedzi na sygnał wejściowy klasyfikator neuronowy powinien wskazać klasę, do której ten sygnał należy. Niniejszy opis sieci neuronowych na podstawie [Roscl].

3.4.1. Matematyczny model neuronu

Stosowany w naukach technicznych matematyczny model neuronu biologicznego stanowi duże uproszczenie jego naturalnego pierwowzoru [Osow94]. Występujące w naturze neurony posiadają jądro, w którym następuje przetwarzanie sygnałów docierających do niego za pośrednictwem aksonów. Po przetworzeniu sygnał przekazywany jest innemu neuronowi za pośrednictwem dendrytu. Połączenie między dendrytem (wyjściem) jednego neuronu a aksonem (wejściem) drugiego nazywane jest połączeniem synaptycznym. Może ono posiadać różną siłę. W matematycznym modelu odpowiednikiem jądra jest pewna funkcja przetwarzania, zaś siły połączeń synaptycznych odpowiadają wagi przypisane poszczególnym wejściom. Funkcja przetwarzania jest złożeniem iloczynu skalarnego wektora wejść z wektorem wag i tzw. funkcji aktywacji (patrz rys. 3.7). Ponieważ naturalny neuron działa w ten sposób, że wytwarza sygnał wyjściowy dopiero wtedy, gdy sygnał wejściowy przekroczy pewną wartość progową, matematyczny model uzupełnia się czasem o dodatkową zmienną, zwaną *bias*, odpowiadającą wartości tego progu. Wartość tej zmiennej, podobnie jak wagi wejść, podlega optymalizacji w procesie uczenia sieci. Wprowadzenie zmiennej *bias* nie zawsze jest stosowane, z uwagi na fakt, iż jej obecność może czasami prowadzić do pogorszenia wyników uczenia sieci.

Wartość omawianej zmiennej jest dodawana do wartości iloczynu skalarnego, dając w sumie wartość wejściową dla funkcji aktywacji, za pomocą której obliczana jest wartość wyjściowa neuronu.

$$s = \sum_{j=1}^N x_j w_j = \mathbf{x}^T \mathbf{w}$$
$$z = s + bias \quad (3.29)$$
$$\mathbf{x}^T = [x_1, x_2, \dots, x_N]$$
$$\mathbf{w}^T = [w_1, w_2, \dots, w_N]$$



Rys. 3.7. Struktura sztucznego neuronu

Funkcja aktywacji może przyjmować różne postaci [Tade93]. W szczególnym przypadku, gdy nieznane są granice wartości wyjściowej, może ona być tożsamością. W pozostałych przypadkach stosowana jest niekiedy funkcja Heaviside'a:

$$y = \mathbf{1}(s) = \begin{cases} 1 & \text{dla } s > 0 \\ 0 & \text{dla } s \leq 0 \end{cases} \quad (3.30)$$

$$s \in \mathbf{R}, \quad y \in \{0,1\}$$

lub funkcja signum:

$$y = \text{sng}(s) = \begin{cases} 1 & \text{dla } s > 0 \\ -1 & \text{dla } s \leq 0 \end{cases} \quad (3.31)$$

$$s \in \mathbf{R}, \quad y \in \{-1,1\}$$

Funkcje progowe (Heaviside'a oraz signum) nie są jednak najlepszym odwzorowaniem procesów zachodzących w biologicznym neuronie. Przyczyną tego jest to, że przyjmują one tylko dwie wartości, przez co uniemożliwiają odwzorowanie wartości pośrednich. Ponadto, są one nieciągłe, przez co nie posiadają pochodnej, co z kolei wyklucza użycie algorytmu wstecznej propagacji błędu. Z tego powodu dużo częściej wykorzystuje się ciągłe funkcje aktywacji w postaci funkcji sigmoidalnej:

$$y = \text{sgm}(s) = \frac{1}{1 + e^{-\lambda s}}$$

$$s \in \mathbf{R}, \quad y \in (0;1) \quad (3.32)$$

$$\lambda \rightarrow \infty \Rightarrow \text{sgm}(s) \rightarrow \mathbf{1}(s)$$

lub tangensa hiperbolicznego:

$$y = \text{tgh}\left(\frac{\lambda s}{2}\right) = \frac{1 - e^{-\lambda s}}{1 + e^{-\lambda s}}$$

$$s \in \mathbf{R}, \quad y \in (-1;1) \quad (3.33)$$

$$\lambda \rightarrow \infty \Rightarrow \text{tgh}\left(\frac{\lambda s}{2}\right) \rightarrow \text{sng}(s)$$

Parametr λ decyduje o kącie nachylenia funkcji aktywacji. Najczęściej jego wartość jest równa 1.

Czasem można spotkać również inne funkcje aktywacji, jak np. funkcja Gaussa lub funkcja wykładnicza, która jest stosowana w przypadku, gdy nieznana jest jedynie górna granica wartości wyjściowej.

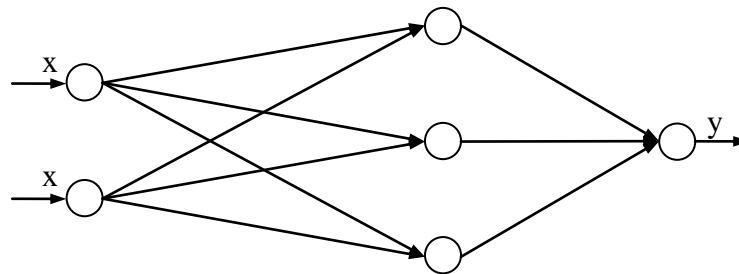
Opisane równaniem (3.29) zależności nie oddają w pełni przebiegu procesów w naturalnym neuronie, w którym tzw. potencjały postsynaptyczne (w matematycznym modelu, sygnały wejściowe) podlegają złożonym procesom sumowania przestrzennego oraz czasowego.

3.4.2. Warstwy sieci neuronowej

Najczęściej spotykaną formą organizacji neuronów jest grupowanie ich w warstwy. Najczęściej występująca struktura sieci typu MLP (*Multilayer Perceptron*), nazywana siecią w pełni połączoną, zakłada istnienie takiej architektury połączeń pomiędzy neuronami, w której każdy neuron warstwy wcześniejszej połączony jest ze wszystkimi neuronami warstwy

następnej. Oczywiście, struktura naturalnych komórek w ludzkim mózgu sprawia wrażenie pozbawionej wszelkich restrykcji w zakresie połączeń. Schemat sieci typu MLP jest jednak o wiele wygodniejszy zarówno z punktu widzenia rozważań teoretycznych, jak i implementacji w postaci programu komputerowego lub wytwarzania w formie układów elektronicznych. Redukcje połączeń nie stanowią przy tym problemu, bowiem odpowiednim współczynnikiem wagowym można nadać stałą wartość, równą 0. Wśród wielu rodzajów tworzonych sieci neuronowych znajdują się także te, które zawierają połączenia pomiędzy neuronami z warstw nie sąsiadujących ze sobą.

W klasycznej terminologii sieci neuronowych wyróżnia się warstwę wejściową, jedną lub dwie warstwy ukryte oraz warstwę wyjściową. Zasadniczą rolą tej warstwy jest pobieranie danych z otoczenia i przesyłania ich do pierwszej warstwy ukrytej, w której dane są przetwarzane i która wytwarza sygnał wyjściowy podawany na wejścia warstwy kolejnej. Na jego końcu znajduje się warstwa wyjściowa i w niej obliczane są wartości wyjść całej sieci, po czym przekazywane są na zewnątrz. Neurony sieci podlegają procesowi uczenia i tak jak komórki nerwowe gromadzą wiedzę o przybliżanych zależnościach w wektorze wag.



Rys. 3.8. Schemat struktury sieci neuronowej z jedną warstwą perceptronową oraz warstwą wejściową i wyjściową

3.4.3. Wstępne i końcowe przetwarzanie danych

Osobny problem stanowi konieczność wstępnego przetworzenia wartości zmiennych wejściowych. Zazwyczaj dokonuje się to w formie przeskalowania. W wyniku tego, że sieć, która używa np. sigmoidalnej funkcji aktywacji, może generować dane wyjściowe tylko ze ściśle wyznaczonego przedziału, czyli $[0; 1]$, przeskalowania prawie zawsze wymagać będą także wartości wyjść. Właściwym miejscem dla wstępnego przetwarzania danych wydaje się być warstwa wejściowa, zaś dla przeskalowania danych wyjściowych osobna warstwa wyjściowa.

Poza określeniem odpowiedniego zestawu zmiennych wejściowych oraz wyjściowych, jak również zgromadzeniem reprezentatywnego zbioru obserwacji, wykorzystywanie sieci neuronowej prawie zawsze wiąże się też ze wstępną oraz końcową obróbką danych, które do sieci są podawane i z niej uzyskiwane. Wstępne przetwarzanie danych ma służyć usunięciu ze zbioru obserwacji wszelkich szumów, zakłóceń, błędów obserwacji czy też cech nieistotnych, jak np. różnice w jasności obrazów z poszczególnych klas, których pozbycie się jest ważne w przypadku rozpoznawaniu obrazów lub mowy. Celem jest wyeliminowanie wszystkiego, co mogłoby zniekształcać rzeczywiste zależności występujące w zgromadzonych danych.

Wyposażona w sigmoidalne funkcje aktywacji sieć neuronowa typu MLP wymaga przeprowadzenia także skalowania lub standaryzacji danych podawanych na jej wejścia. Teoretycznie sieć tego typu jest co prawda w stanie operować na surowych danych, ale w praktyce odpowiednia transformacja jest niezbędna, gdyż jej brak skutkuje istotnymi zakłóceniami w procesie uczenia i gorszymi właściwościami nauczonej sieci. Duże różnice w zakresach wartości przyjmowanych przez zmienne, a więc i w zakresach zmienności

bezwzględnej, mogą wywoływać niepożądany efekt w działaniu neuronu, bowiem zakłócają wpływ poszczególnych wejść. Poza tym, prowadzą one także do nasycenia sigmoidalnej funkcji aktywacji, której pochodna zbiega się wtedy do wartości 0, a przez to blokuje proces uczenia. Sieci, które są uczone w oparciu o takie zmienne, są również bardziej podatne na utkwienie w minimach lokalnych funkcji celu.

Powyższe problemy są na ogół rozwiązywane dzięki skalowaniu zmiennych wejściowych względem odchylenia od wartości minimalnej do przedziału zmienności funkcji aktywacji. Ponieważ zakres wartości funkcji aktywacji, np. dla sigmoidy [0; 1], zazwyczaj nie pokrywa się z zakresem wartości typowym dla konkretnej zmiennej wyjściowej, prawie zawsze jest niezbędne także skalowanie w stosunku do wartości zmiennych wyjściowych sieci. Zaprezentowane poniżej wzory pozwalają na skalowanie do przedziału [0; 1] oraz wykonanie operacji odwrotnej i uzyskanie pierwotnego zakresu wartości zmiennej:

$$\begin{aligned}
 x_{i,j}^{(1)} &= \frac{x_{i,j} - x_j^{(\min)}}{x_j^{(\max)} - x_j^{(\min)}} \\
 x_{i,j}^{(1)} &\in (0;1) \\
 x_{i,j} &\in \left(x_j^{(\min)}; x_j^{(\max)} \right) \\
 y_{i,j} &= y_j^{(\min)} + y_{i,j}^{(L)} \left(y_j^{(\max)} - y_j^{(\min)} \right) \\
 y_{i,j} &\in \left(y_j^{(\min)}; y_j^{(\max)} \right) \\
 y_{i,j}^{(L)} &\in (0;1)
 \end{aligned} \tag{3.34}$$

Innym sposobem przekształcenia danych wejściowych jest standaryzacja, którą najczęściej dokonuje się przez odjęcie od zmiennej jej wartości średniej i podzielenie przez odchylenie standardowe:

$$\begin{aligned}
 x_{i,j}^{(1)} &= \frac{x_{i,j} - \bar{x}_j}{\sigma_{x_j}} \\
 \bar{x}_j^{(1)} &= 0 \\
 \sigma_{x_j^{(1)}} &= 1 \\
 \bar{x}_j &= \frac{1}{T} \sum_{i=1}^T x_{i,j} \\
 \sigma_{x_j} &= \sqrt{\frac{1}{T} \sum_{i=1}^T (x_{i,j} - \bar{x}_j)^2}
 \end{aligned} \tag{3.35}$$

Jeszcze innym typowym przekształceniem danych wejściowych jest normalizacja (dzielenie składowej przez normę wektora), a poza nią także różnego rodzaju transformacje nieliniowe. Zbiór danych treningowych może być też powiększany przez dodanie sztucznego szumu lub dołączenie sztucznych danych.

3.4.4. Klasyczny algorytm wstecznej propagacji błędów

Algorytm wstecznej propagacji błędów należy do tzw. metod gradientowych, które wykorzystują prawidłowość mówiącą, iż gradient funkcji wskazuje kierunek jej najszybszego wzrostu, a w przypadku zmiany znaków składowych na przeciwne, czyli pomnożeniu przez -1 , kierunek jej najszybszego spadku. Dzięki tej właściwości, można stosunkowo szybko zminimalizować funkcję celu posuwając się w kierunku jej najszybszego spadku.

Funkcja celu sieci neuronowej może mieć różne postaci, ale w klasycznej wersji używany jest kwadrat błędów sieci, którym najczęściej jest błąd średniokwadratowy. Wyznaczany jest on jako suma kwadratów różnic pomiędzy aktualnymi wartościami wyjść wygenerowanymi dla aktualnego wektora wejść a wartościami wzorcowymi, czyli takimi, do jakich sieć neuronowa powinna dążyć przetwarzając aktualny wektor wejść.

Algorytm wstecznej propagacji błędów sprowadza się do modyfikacji każdej z wag proporcjonalnie do wartości pochodnej cząstkowej funkcji celu. Modyfikacja wag w k -tej iteracji polega na odjęciu od wartości wag z iteracji poprzedniej ($k - 1$) wektora gradientu obliczonego dla bieżącej obserwacji:

$$\mathbf{w}_i^{(n)}(k) = \mathbf{w}_i^{(n)}(k-1) - \alpha \frac{\partial Q(k)}{\mathbf{w}_i^{(n)}} \quad (3.36)$$

$$\alpha > 0$$

Algorytm jest sparametryzowany w wyniku użycia tzw. współczynnika uczenia, zwanego inaczej długością kroku (w równaniu 3.36, symbol α). Współczynnik ten przyjmuje najczęściej wartości z zakresu od 0,5 do 0,9, choć w niektórych przypadkach może osiągać nawet znacznie większą wartość.

Wektor gradientu, który jest najważniejszym składnikiem wzoru (3.36), można przedstawić w postaci iloczynu pochodnej funkcji Q względem wyjścia j -tego neuronu z n -tej warstwy i pochodnej cząstkowej funkcji wyjścia tego samego neuronu względem własnego wektora wag:

$$\frac{\partial Q(k)}{\partial \mathbf{w}_i^{(n)}} = \frac{\partial Q(k)}{\partial y_{i,j}^{(n)}} \frac{\partial y_{i,j}^{(n)}}{\partial \mathbf{w}_i^{(n)}} \quad (3.37)$$

Wyrażenie (3.37) dobrze jest z kolei przedstawić w formie sumy iloczynów dwóch czynników: pochodnych funkcji Q względem wyjść neuronów z warstwy następnej ($n + 1$) i pochodnych funkcji wyjść tych samych neuronów względem tych z ich wejść, do których trafia sygnał wyjściowy pochodzący z j -tego neuronu warstwy poprzedniej (n -tej):

$$\begin{aligned} \frac{\partial Q(k)}{\partial y_{i,j}^{(n)}} &= \sum_{j=1}^{N+1} \frac{\partial Q(k)}{\partial x_{i,j}^{(n+1)}} = \sum_{j=1}^{N+1} \frac{\partial Q(k)}{\partial y_{i,j}^{(n+1)}} \frac{\partial y_{i,j}^{(n+1)}}{\partial x_{i,j}^{(n+1)}} \\ \frac{\partial Q(k)}{\partial y_{i,j}^{(n)}} &= \sum_{j=1}^{N+1} \frac{\partial Q(k)}{\partial y_{i,j}^{(n+1)}} \frac{\partial y_{i,j}^{(n+1)}}{\partial s_{i,j}^{(n+1)}} \frac{\partial s_{i,j}^{(n+1)}}{\partial x_{i,j}^{(n+1)}} \quad (3.38) \\ \frac{\partial Q(k)}{\partial y_{i,j}^{(n)}} &= \sum_{j=1}^{N+1} \frac{\partial Q(k)}{\partial y_{i,j}^{(n+1)}} y_{i,j}^{(n+1)} \left(1 - y_{i,j}^{(n+1)}\right) w_{j,i}^{(n+1)} \end{aligned}$$

Analiza wyprowadzenia powyższego wzoru (3.38) ukazuje, iż niezależnie od liczby warstw możliwe jest obliczenie pochodnej funkcji celu Q w stosunku do wyjść dowolnego

neuronu. W praktyce proces ten odbywa się w kierunku odwrotnym: od neuronów warstwy o numerze najwyższym do neuronu docelowego w warstwie o numerze niższym. W iteracyjnym wyznaczaniu kolejnych pochodnych wykorzystuje się wcześniej obliczone pochodne z warstw o wyższej numeracji. Jedynym wyjątkiem jest tu pochodna funkcji celu Q po wyjściach ostatniej warstwy perceptronowej:

$$\frac{\partial Q(k)}{\partial y_{i,j}^{(L)}} = y_{i,j}^{(L)} - p_{i,j}^{(L)} \quad (3.39)$$

Ostatnim elementem, niezbędnym do wyznaczenia potrzebnego w algorytmie gradientu, jest druga część iloczynu ze wzoru (3.37), a więc pochodna wyjścia neuronu po jego wagach. Dzięki wykorzystaniu sigmoidalnej funkcji aktywacji, która posiada stosunkowo prostą pierwszą pochodną, potrzebny wzór przybiera postać:

$$\frac{\partial y_{i,j}^{(n)}}{\partial \mathbf{w}_i^{(n)}} = \frac{\partial y_{i,j}^{(n)}}{\partial s_{i,j}^{(n)}} \frac{\partial s_{i,j}^{(n)}}{\partial \mathbf{w}_i^{(n)}} = y_{i,j}^{(n)} \left(1 - y_{i,j}^{(n)}\right) \mathbf{x}_i^{(n)} \quad (3.40)$$

Użycie innej formy funkcji przejścia zmienia jedynie pierwszą część wyrażenia (3.40). Funkcja ta musi być jednak ciągła i różniczkowalna, co nie jest spełnione np. w odniesieniu do funkcji progowej.

Algorytm wstecznej propagacji błędu posiada niestety dużo wad. Największą z nich jest to, iż nie gwarantuje on osiągnięcia minimum globalnego funkcji celu. Zamiast tego może prowadzić do osiągnięcia minimum lokalnego, które może dać wyniki satysfakcjonujące, ale też okazać się całkowicie bezużyteczne. Dlatego jedną z metod poszukiwania optymalnych wag neuronów jest wielokrotne ponawianie procesu uczenia w oparciu o zbiory początkowych wag wybieranych losowo, zazwyczaj z przedziału $[-0,5; 0,5]$.

Jeszcze inną trudność sprawia konieczność wyznaczenia odpowiedniego warunku zatrzymania algorytmu. Najczęściej stosowanym kryterium jest tutaj zmniejszenie się wartości błędu poniżej przyjętego progu.

3.4.5. Momentowa metoda wstecznej propagacji błędu

Najczęściej stosowaną modyfikacją algorytmu wstecznej propagacji błędu jest momentowa metoda wstecznej propagacji błędu. Jej istota sprowadza się do wyznaczenia wag w kolejnej iteracji zarówno w oparciu o gradient funkcji celu, jak i zmiany tych samych wag w iteracji wcześniejszej. Dzięki zastosowaniu tej modyfikacji, uzyskuje się stosunkowo bezpieczne przyspieszenie procesu uczenia i częściowe ograniczenie prawdopodobieństwa utknięcia w minimum lokalnym funkcji celu. Metoda ta wprowadza do opisanego wcześniej sposobu modyfikacji wag dodatkowy składnik, który jest zależny od parametru β , należącego do przedziału $(0, 1]$:

$$\begin{aligned} \mathbf{w}_i^{(n)}(k) &= \mathbf{w}_i^{(n)}(k-1) - \alpha \frac{\partial Q(k)}{\partial \mathbf{w}_i^{(n)}} + \beta \Delta \mathbf{w}_i^{(n)}(k-1) \\ \Delta \mathbf{w}_i^{(n)}(k-1) &= \mathbf{w}_i^{(n)}(k-1) - \mathbf{w}_i^{(n)}(k-2) \\ \beta &\in (0, 1] \end{aligned} \quad (3.41)$$

Obecność w algorytmie tego składnika wprowadza do procesu uczenia element bezwładności, który powoduje tłumienie chwilowych i gwałtownych oscylacji wektora gradientu funkcji celu.

W najprostszym i najczęściej występującym przypadku przyjmuje się, że w procesie uczenia wartości współczynników uczenia oraz momentu są stałe. Problem stanowi dobór optymalnych wartości tych współczynników, gdyż optima te zależne są od napotykanego kształtu funkcji celu. W przypadku napotkania „płaskowyzu” najkorzystniejsze są stosunkowo duże wartości tych współczynników. W takim przypadku powodują one bowiem przyspieszenie procesu uczenia, podczas gdy niskie wartości mają działanie odwrotne.

W przypadku przechodzenia przez obszar „wąwozu”, najkorzystniejszymi okazują się małe wartości. Przyjęcie zbyt dużych wartości współczynnika uczenia prowadziłyby do niemożności trwałego zejścia przez funkcję celu w obszar minimum „wąwozu”.

3.5. Podsumowanie

Opisane w niniejszym rozdziale metody klasyfikacji wykorzystano w części doświadczalnej pracy. W szczególności w rozdziale 6. zastosowano:

1. do klasyfikacji danych, metody: k -średnich, rozmytych k -średnich, Gustafsona-Kessla, k -najbliższych sąsiadów, rozmytych k -najbliższych sąsiadów, MMD oraz SVM;
2. sieć neuronową do detekcji lokalnych maksimum w przestrzeni transformacji Hougha.

Porównanie skuteczności poszczególnych metod jest przedstawione w tabeli 6.4. Jak widać, najkorzystniejsze okazało się użycie sieci neuronowych.

4. Algorytm 1. – skanowanie obrazu za pomocą specjalnej maski

4.1. Wstęp

W tym rozdziale przedstawiono pierwszy z opracowanych algorytmów do detekcji par linii Kikuchiego na obrazach mikroskopowych struktur polikrystalicznych. Istotą jego działania jest skanowanie obrazu za pomocą maski złożonej z trzech równoległych linii prostych. Dla każdego kolejnego położenia maski względem obrazu uzyskuje się piksele leżące wzdłuż tych trzech linii. Następnie, w oparciu o przyjęty model linii Kikuchiego na obrazie, przeprowadza się analizę właściwości statystycznych, a w szczególności wartości średniej, uzyskanych w ten sposób pikseli, co pozwala na identyfikację ww. linii.

Przyjęty model linii wyrażony jest za pomocą następujących założeń:

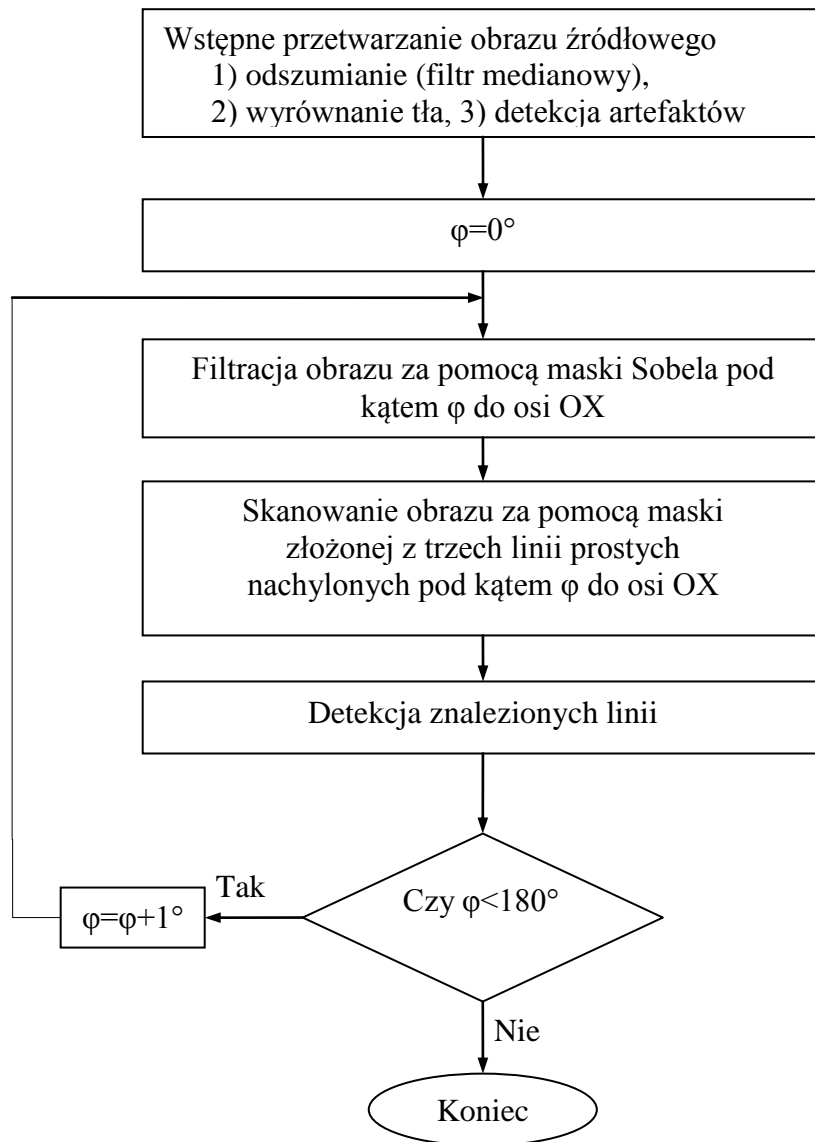
- linię charakteryzuje istnienie długiego ciągu pikseli o zbliżonej wartości i zwróconego w jednym kierunku;
- piksele znajdujące się w bezpośrednim sąsiedztwie linii mają wartości znacząco różne od pikseli położonych wzdłuż linii;
- długość ciągu pikseli jest znacznie większa niż szerokość;
- linia przebiega od jednej krawędzi obrazu do drugiej [Lass98].

4.2. Struktura algorytmu

Na rysunku (4.1) przedstawiono schemat blokowy opracowanego algorytmu. W kolejnych punktach opisano jego poszczególne części składowe.

4.3. Wstępne przetwarzanie obrazu źródłowego

Pierwszym etapem pracy algorytmu jest wstępne przetworzenie obrazu źródłowego. Celem tej operacji jest redukcja możliwych zakłóceń występujących na obrazie źródłowym (np. szumu lub nierównomiernego oświetlenia obrazu).



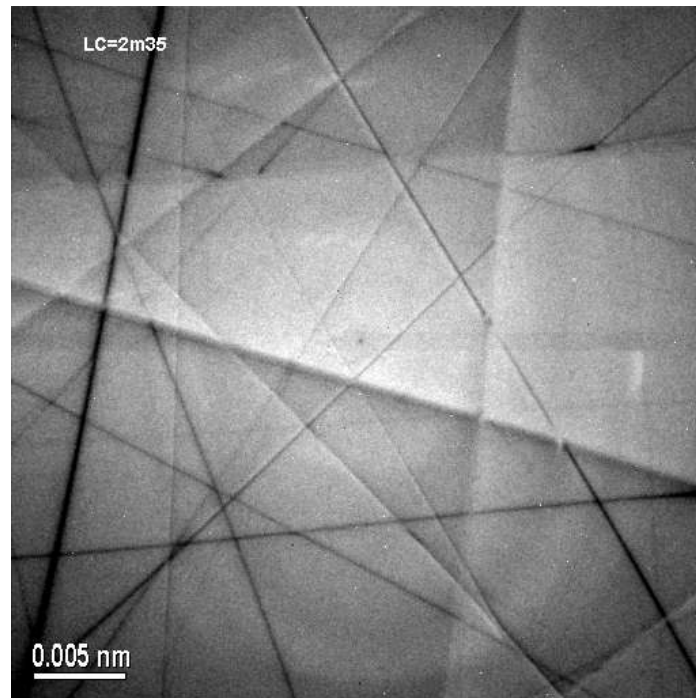
Rys. 4.1. Schemat blokowy algorytmu 1.

4.3.1. Odszumianie obrazu

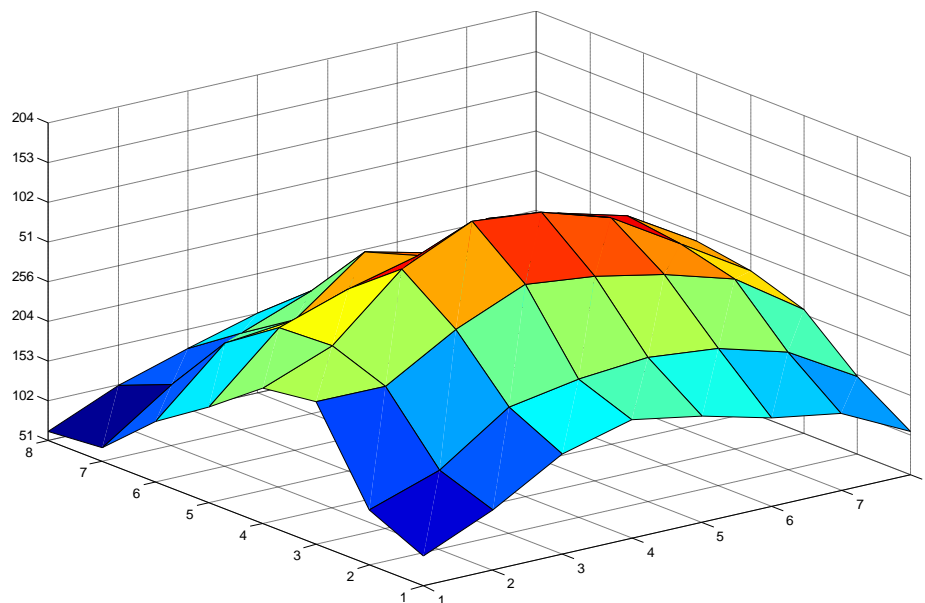
W celu zredukowania szumu występującego na obrazie zastosowano filtr medianowy. Filtr tego typu, w odróżnieniu od filtra uśredniającego, nie powoduje znacznego rozmywania krawędzi linii [Asto89], co ma znaczenie zwłaszcza w przypadku linii o niewielkiej szerokości i niewielkim kontraście w stosunku do otoczenia. Zbyt duże rozmycie utrudniłoby detekcję tego typu linii. Zastosowanie filtra medianowego pozwala na zachować na odszumionym obrazie stosunkowo ostre krawędzie linii.

4.3.2. Wyrównanie nierównomiernego tła obrazu

Ponieważ niektóre obrazy źródłowe charakteryzują się dość znaczną nierównomiernością oświetlenia tła obrazu, kolejnym krokiem wstępnego przetwarzania jest procedura wyrównania nierównomiernego oświetlenia tła obrazu. Rysunek (4.2) przedstawia przykładowy obraz z nierównomiernym tłem. Na rysunku (4.3) przedstawiono lokalną estymatę tła tego obrazu.



Rys. 4.2. Obraz z nierównomiernym tłem



Rys. 4.3. Estymata tła obrazu z rys. (4.2)

Procedura wyrównania nierównomiernego oświetlenia tła obrazu składa się z następujących kroków:

- podzieli obraz na kwadratowe bloki o rozmiarze 64 na 64 piksele;
- wyznacz wartość średnią dla pikseli znajdujących się w każdym z bloków:

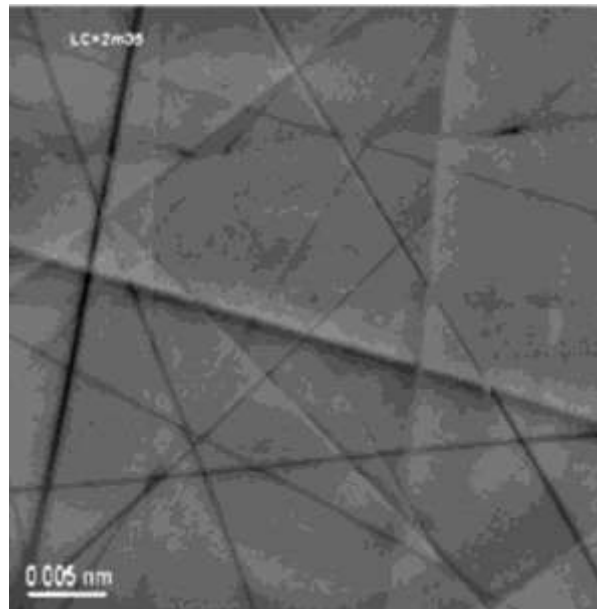
$$\bar{y}_i = \frac{1}{|A_i|} \sum_{k \in A_i} y_k \quad (4.1)$$

gdzie y_k jest k -tym pikselem, A_i jest i -tym blokiem, zaś $|A_i|$ jest liczbą pikseli należących do A_i ;

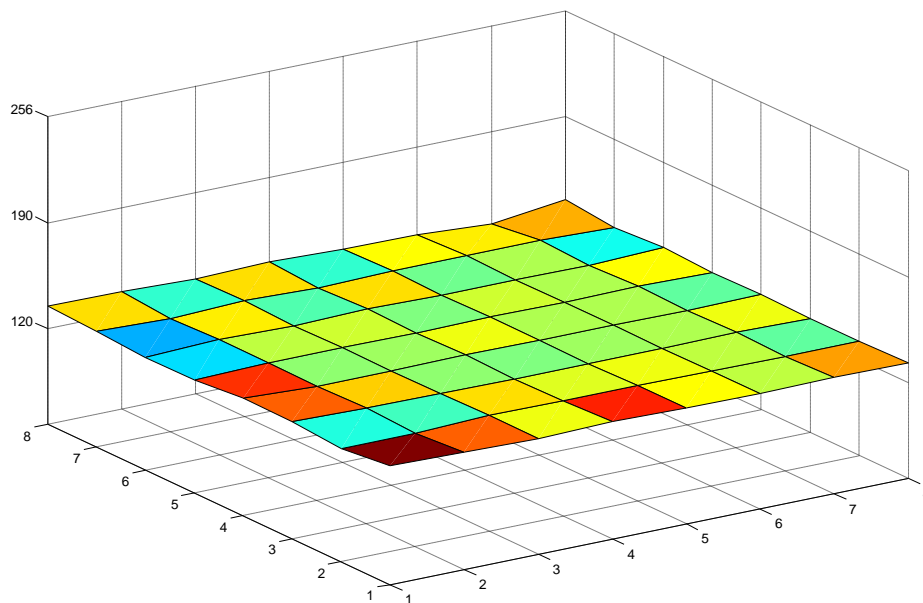
- do każdego piksela obrazu dodaj odpowiednią stałą (wartość 128 jest docelową wartością tła obrazu):

$$y_k = y_k + (128 - \bar{y}_i), \quad k \in A_i \quad (4.2)$$

W wyniku przeprowadzonej operacji, uzyskuje się obraz przedstawiony na rysunku (4.4), który posiada równomierne tło. Jego estymata jest pokazana na rys. (4.5).



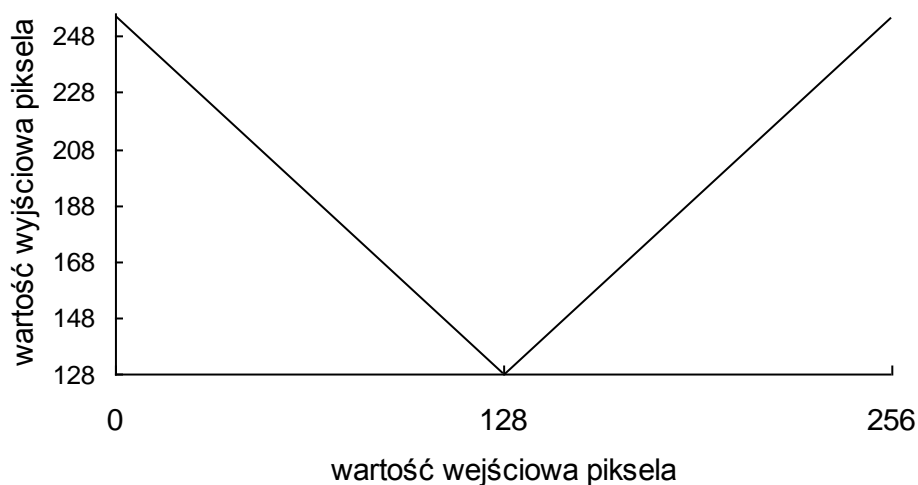
Rys. 4.4. Obraz po operacji wyrównania tła



Rys. 4.5. Estymata tła obrazu po operacji wyrównania tła

Na obrazach mikroskopowych niektóre linie Kikuchiego składają się z dwóch równoległych części: jednej ciemnej, a drugiej jasnej. Ponadto, niektóre linie Kikuchiego są jaśniejsze od tła, a inne ciemniejsze. Zdarza się również, iż niektóre linie są w różnych miejscach inaczej widoczne, to znaczy, że linia na pewnym odcinku może być jaśniejsza od tła, zaś na innym ciemniejsza. Ponieważ w dalszej części algorytm jest tak skonstruowany, że

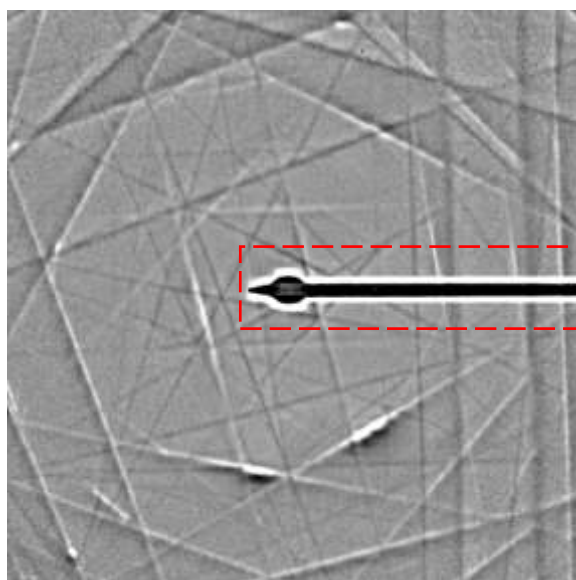
dokonywa detekcji linii jaśniejszych od tła, kolejnym krokiem jest przekształcenie obrazu za pomocą odpowiedniej krzywej tonalnej. Celem tego przekształcenia jest odwzorowanie ciemnych części niektórych linii w jasne. W ten sposób uzyskuje się uproszczenie i ujednoczenie tego aspektu w detekcji linii Kikuchiego.



Rys. 4.6. Krzywa transformacji obrazu

4.3.3. Detekcja i usunięcie artefaktów

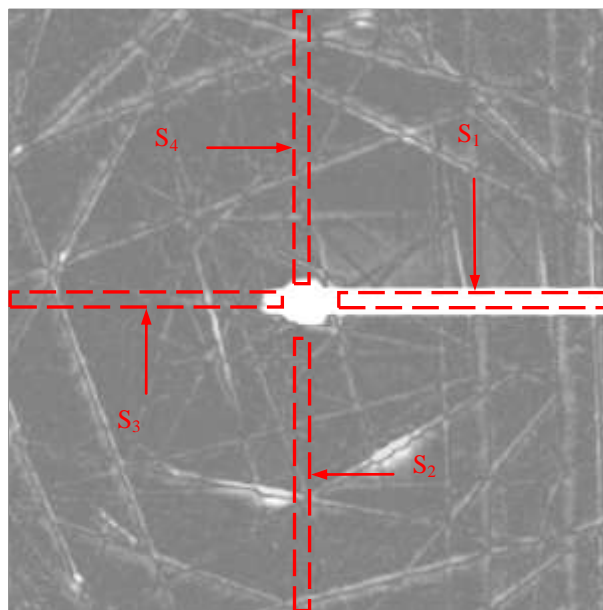
Na niektórych obrazach występują artefakty spowodowane obecnością przysłony mającej charakterystyczny kształt „zapalki” (patrz rys. 4.7). Związane jest to z tym, iż niekiedy w celu uniknięcia przeświecenia obrazu należy przysłonić jego środkową część. Obecność na obrazie przysłony może powodować detekcje fałszywych linii związanych z krawędziami przysłony. Dlatego istnieje konieczność jej detekcji oraz usunięcia z obrazu.



Rys. 4.7. Przysłona na obrazie mikroskopowym

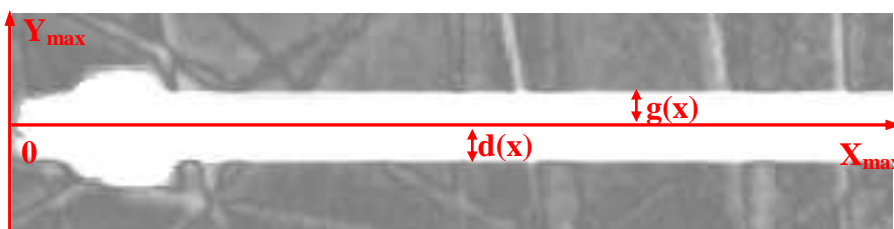
Ponieważ obraz źródłowy może być obrócony w stosunku do pokazanego na rys. (4.7) o kąt 90° , 180° lub 270° , dlatego dokonuje się detekcji przysłony oddzielnie w każdym z zaznaczonych na rys. (4.8) czerwonymi liniami obszarów. Stwierdzono, iż na badanych obrazach

wysokość poziomej przesłony nie przekracza 70 pikseli, dlatego wysokość poziomych obszarów detekcji (obszary S_1 oraz S_3 na rys. 4.8) wynosi 60 pikseli, zaś ich długość wynosi 200 pikseli. Pionowe obszary detekcji (S_2 oraz S_4) mają szerokość oraz wysokość równe odpowiednio 60 oraz 200 pikseli.



Rys. 4.8. Obszary detekcji przesłony

Detekcji dokonuje się na podstawie analizy wartości średniej pikseli znajdujących się w każdym z analizowanych obszarów. Jeśli analizowany obszar pokrywa się z przesłoną, to wartość średnia pikseli go tworzących osiąga dużą wartość. Z kolei, gdy analizowany obszar nie pokrywa się z przesłoną, wtedy wartość średnia jego pikseli jest bliska 128 (tło). Ostatecznie, na podstawie analizy 30 badanych obrazów, przyjęto, że jeśli wartość średnia jest większa od 230, wtedy badany obszar pokrywa się z przesłoną. Jeśli stwierdza się obecność przesłony, to dla odpowiedniego obszaru wyznacza się położenie jej krawędzi. Dokonuje się tego poprzez wyznaczenie zmiennych $d(x)$ oraz $g(x)$ będących funkcjami odległości krawędzi przesłony od poziomej (pionowej) symetrycznej obrazu (rys. 4.9).



Rys. 4.9. Wyznaczenie profilu przesłony

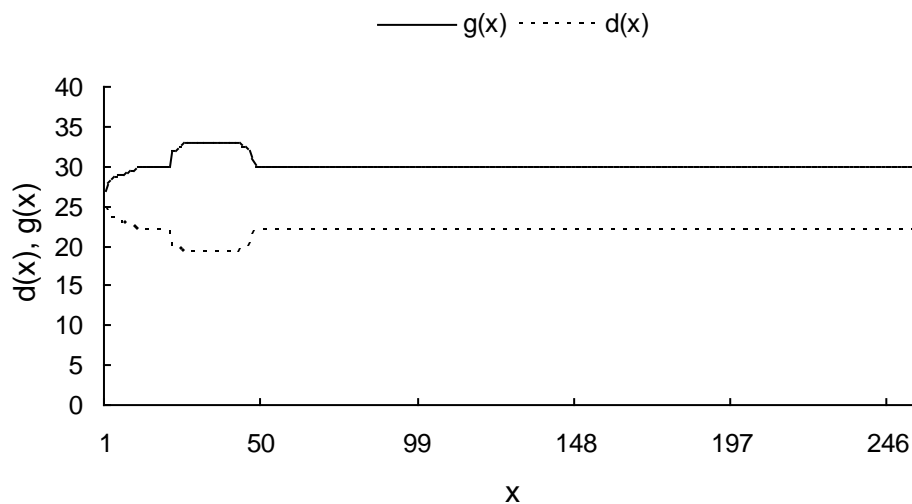
Tabela 4.1. Algorytm wyznaczenia profilu przesłony ($A(x,y)$ jest wartością piksela obrazu)

```

dla x = 0... $X_{\max}$ :
{
  dla y = 0... $Y_{\max}$ :
  {
    jeśli  $A(x,y) < 175$ 
    {
       $g(x) = y$ 
    }
  }
  dla y = 0... $Y_{\min}$ :
  {
    jeśli  $A(x,y) < 175$ 
    {
       $d(x) = y$ 
    }
  }
}

```

W wyniku działania algorytmu przedstawionego w tabeli (4.1.), otrzymuje się dwa jednowymiarowe sygnały (patrz rys. 4.9).



Rys. 4.9. Przebieg funkcji $d(x)$ w przypadku detekcji „zapalki”

Gdy wyznaczone są już przebiegi funkcji $d(x)$ oraz $g(x)$, dokonuje się interpolacji obrazu wzdłuż prostej, prostopadłej do krawędzi przesłony. W efekcie uzyskuje się skuteczną eliminację przesłony przy jednoczesnym zapewnieniu ciągłości obrazu w miejscach gdzie były jej krawędzie. (rys. 4.10).



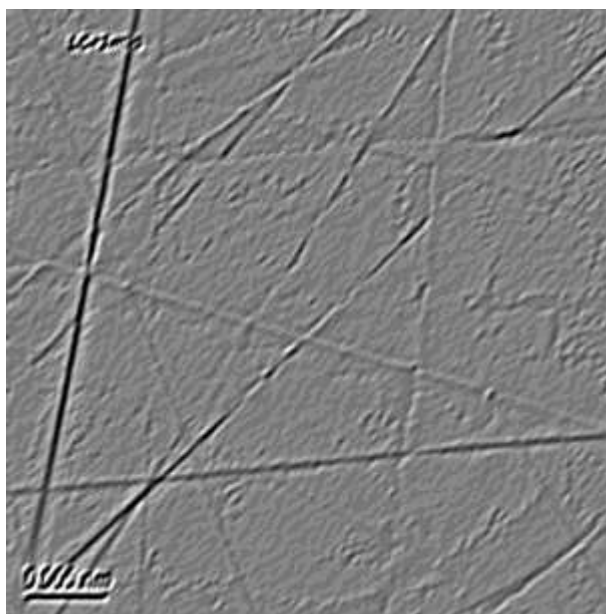
Rys. 4.10. Fragment obrazu po eliminacji przesłony

4.4. Pętla główna algorytmu

Od tego miejsca algorytm wchodzi w pętlę, która przebiega dla kolejnych wartości kąta (φ) nachylenia maski do osi odniesienia. Wartość kąta zmienia się w zakresie od 0° do 180° z krokiem wynoszącym 1° . Przed każdym kolejnym obiegiem pętli następuje utworzenie maski składającej się z trzech linii prostych, nachylonych pod kątem φ do wybranej osi odniesienia (jako oś odniesienia przyjęto oś OX).

4.4.1. Filtracja za pomocą maski Sobela

W celu poprawy rozpoznawalności słabo widocznych linii, zastosowano dwukrotną filtrację obrazu za pomocą poziomej maski Sobela [Mali02], obróconej o kąt φ względem osi OX. Filtracja ta stanowi estymatę drugiej pochodnej obrazu w kierunku określonym przez ww. kąt. Obróconą maskę uzyskuje się traktując macierz, z której zbudowana jest maska, jako obraz, a następnie stosując standardowe algorytmy obracania obrazu oparte o interpolację biliniową. W rezultacie otrzymuje się maskę, która charakteryzuje się tym, iż wzmacnia najbardziej te krawędzie, które są nachylone pod kątem φ do osi OX, zaś całkowicie tłumi krawędzie nachylone pod kątami $\varphi+90^\circ$ oraz $\varphi-90^\circ$.



Rys. 4.9 Obraz po filtracji maską Sobela obróconą o kąt $\varphi=45^\circ$

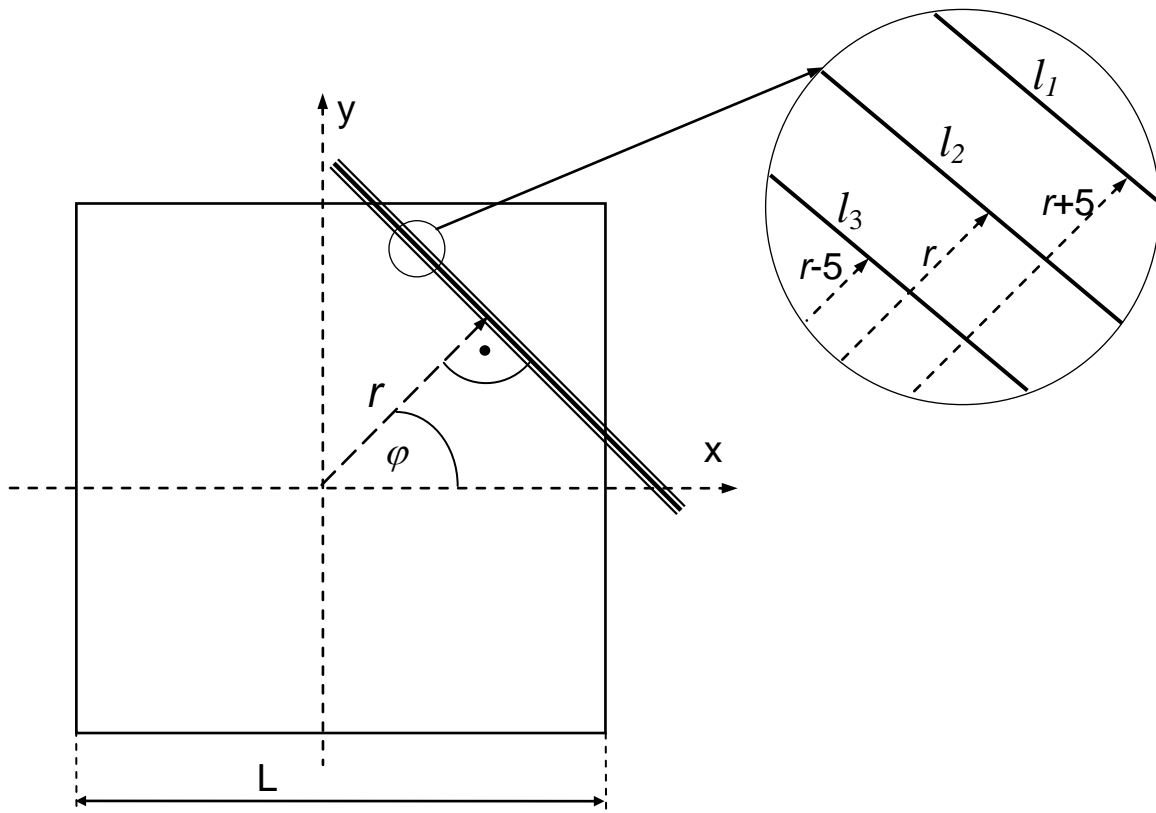
4.4.2. Skanowanie obrazu za pomocą maski złożonej z trzech linii prostych

W tym momencie zaczyna się skanowanie obrazu za pomocą wcześniej utworzonych trzech linii prostych (l_1 , l_2 , l_3). Odległości pomiędzy nimi określają maksymalną szerokość linii Kikuchiego, która może być wykryta. Aby linia mogła być wykryta, powinna się całkowicie zmieścić w obszarze pomiędzy prostą l_1 oraz l_3 . Z przeprowadzonych badań wynika, że szerokość występujących na obrazach linii Kikuchiego nie przekracza 8 pikseli. Dlatego przyjęto, że odległość pomiędzy poszczególnymi prostymi wynosi 5 pikseli, tak że całkowita szerokość obszaru pomiędzy prostymi l_1 oraz l_3 wynosi 10 pikseli, co zapewnia możliwość detekcji najszerszych linii Kikuchiego. Położenie maski względem przyjętego układu odniesienia określają dwa parametry: długość promienia r oraz kąt φ nachylenia tego promienia do osi poziomej. Dla każdej ustalonej wartości kąta φ następuje zmiana wartości

promienia r w zakresie od $-0,5L$ do $0,5L$ (gdzie L to wymiar obrazu w pikselach). Ujemnym wartościom promienia r odpowiadają położenia maski w drugiej i trzeciej ćwiartce układu współrzędnych. Dla kolejnych położenia maski na obrazie zostają zebrane te piksele obrazu, które leżą dokładnie wzdłuż aktualnego położenia maski. Ponieważ współrzędne kolejnych punktów maski mają na ogół wartości niecałkowite, dlatego w celu wyznaczenia wartości pikseli w tych punktach zastosowano liniową interpolację obrazu [Skar93]. W ten sposób uzyskano zbiór interpolowanych pikseli obrazu, leżących wzdłuż aktualnego położenia maski. Zbiór ten stanowi pomocniczy obraz składający się z trzech linii. Na rysunkach (4.11 oraz 4.12) przedstawiono przykładowy wygląd obrazów pomocniczych w przypadkach, gdy przesuwająca się wzdłuż obrazu maska nie pokrywa się z linią Kikuchiego, oraz gdy maska pokrywa się z linią Kikuchiego.

Jak widać z rysunku (4.12), w przypadku, gdy przesuwająca się wzdłuż obrazu maska pokrywa się z linią Kikuchiego, piksele położone wzdłuż prostej l_2 mają wartości znacznie różne od pikseli położonych wzdłuż prostych l_1 oraz l_3 . Właściwość ta odpowiada przyjętemu modelowi linii na obrazie i dlatego stanowi podstawę do detekcji linii Kikuchiego.

Dalsze przetwarzanie obrazu pomocniczego polega na poddaniu go operacji binaryzacji, która przebiega następująco:



Rys.4.10. Proces skanowania obrazu za pomocą trzech linii równoległych



Rys.4.11. Przypadek gdy maska nie pokrywa się z linią



Rys. 4.12. Przypadek gdy maska pokrywa się z linią

Wyznaczana jest wartość średnia a_2 z pikseli położonych wzdłuż prostej l_2 i ta średnia stanowi próg binaryzacji dla prostych l_1 oraz l_3 . Jeśli wartość piksela na prostej l_1 lub l_3 jest większa od średniej a_2 , to na obrazie binarnym odpowiada mu piksel o wartości 0, w przeciwnym wypadku odpowiedni piksel na obrazie binarnym ma wartość 1. Następnie wyznaczana jest wartość średnia $a_{1,3}$ z pikseli położonych wzdłuż prostych l_1 oraz l_3 , która stanowi próg binaryzacji dla prostej l_2 . Jeśli wartość piksela na prostej l_2 jest większa od $a_{1,3}$, to odpowiedni piksel na obrazie binarnym ma wartość 0, w przeciwnym przypadku – wartość 1. W ten sposób uzyskuje się obrazy binarne przedstawione na rysunkach (4.15 oraz 4.16).

Gdy przesuwająca się wzdłuż obrazu maska pokryje się z linią Kikuchiego, otrzymuje się obraz binarny, w którym prosta l_2 zawiera długi ciąg pikseli mających wartość 1, zaś proste l_1 oraz l_3 zawierają długie ciągi pikseli mających wartość 0.

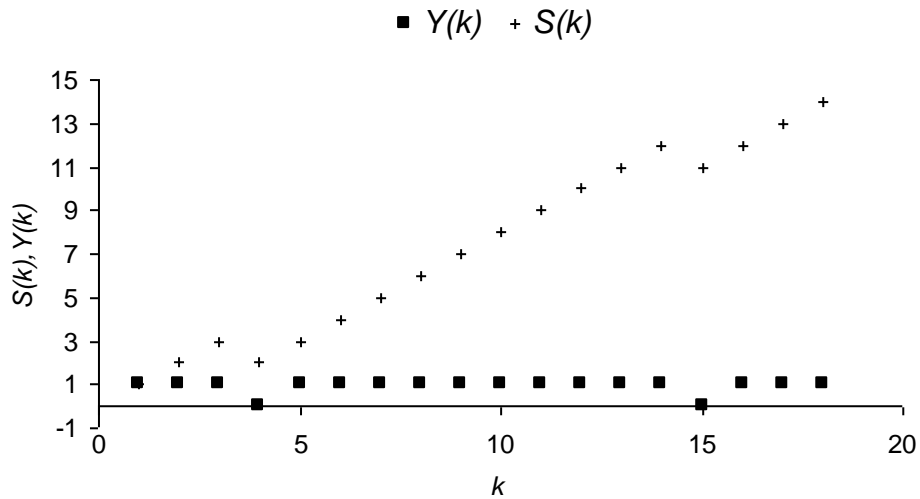
Każda z linii: l_1 , l_2 , l_3 ma długość N pikseli, która zależy od aktualnego położenia maski względem obrazu. Dla każdej linii jest wyznaczany współczynnik v_n ($n = 1, 2, 3; k = 1, 2, \dots, N$):

$$v_n = \frac{\max\{S(k)\}}{N} \quad (4.3)$$

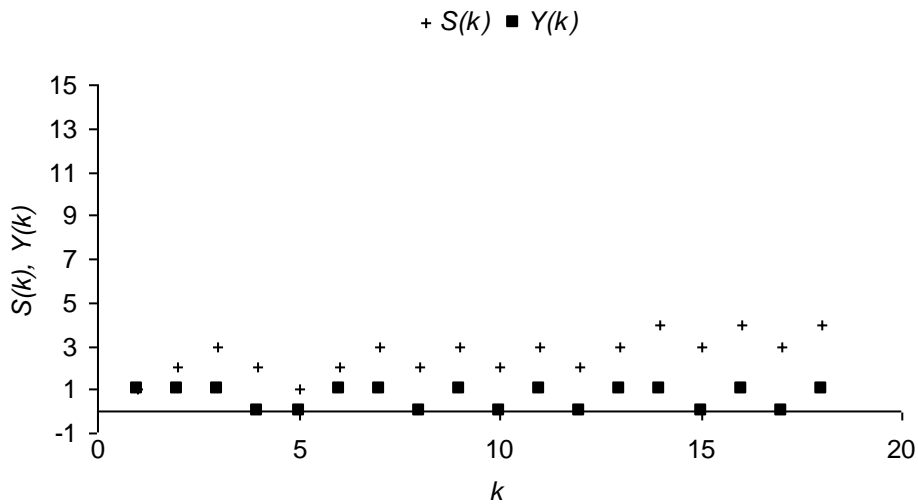
gdzie:

$$S(k) = \sum_{i=1}^k Y_{l_n}(i) \quad Y_{l_n}(i) = \begin{cases} 1 & \text{gdy } i\text{-ty piksel na prostej } l_n \text{ ma wartość } 1 \\ -1 & \text{gdy } i\text{-ty piksel na prostej } l_n \text{ ma wartość } 0 \end{cases}$$

Wartość występująca w liczniku równania (4.3) to maksimum sumy kumulacyjnej. Suma ta osiąga dużą wartość, gdy na pomocniczym obrazie binarnym występuje długi ciąg białych pikseli (rys. 4.13). W przypadku zaś, gdy na pomocniczym obrazie binarnym występują na przemian białe i czarne piksele, suma nie osiąga dużej wartości (rys. 4.14).



Rys. 4.13. Wartości $Y(k)$ i $S(k)$ w przypadku wystąpienia długiego ciągu „jedynek”



Rys. 4.14. Wartości $Y(k)$ i $S(k)$ w przypadku niewystępowania długiego ciągu „jedynek”



Rys. 4.15. Pomocniczy obraz binarny gdy maska nie pokrywa się z linią Kikuchiego



Rys. 4.16. Pomocniczy obraz binarny gdy maska pokrywa się z linią Kikuchiego

Współczynnik v_n przyjmuje wartości z przedziału $[-1; 1]$, przy czym wartość tego współczynnika jest bliska -1 wtedy, gdy na obrazie binarnym wzdłuż prostej l_n występuje długi ciąg pikseli o wartości 0 , zaś bliska 1 wtedy, gdy na prostej l_n występuje długi ciąg pikseli o wartości 1 (rys. 4.16).

Gdy wartość współczynników v_1 , v_2 , v_3 jest już ustalona, obliczany jest parametr p według wzoru:

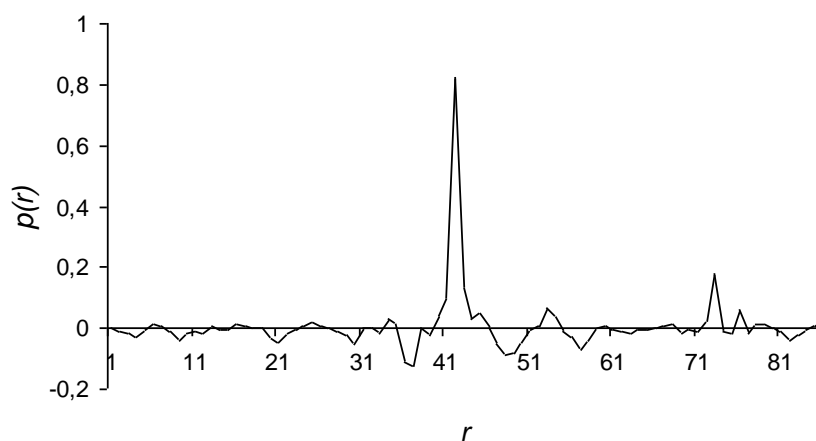
$$p(r) = \frac{1}{3} \cdot (v_2 - v_1 - v_3) \quad (4.4)$$

Parametr p przyjmuje wartości z przedziału $[-1; 1]$. Posiada on tę właściwość, że jego wartość jest bliska 1 w przypadku, gdy binarna reprezentacja prostej l_2 zawiera długi ciąg pikseli mających wartość 1 , a jednocześnie binarne reprezentacje prostych l_1 oraz l_3 zawierają długi ciąg pikseli mających wartość 0 (rys. 4.16). Sytuacja taka ma miejsce tylko wtedy, gdy przesuwaną się wzdłuż obrazu maska pokryje się z linią Kikuchiego. W przypadku, gdy maska nie pokrywa się z linią Kikuchiego, wartość parametru p jest bliska 0 lub mniejsza od 0 .

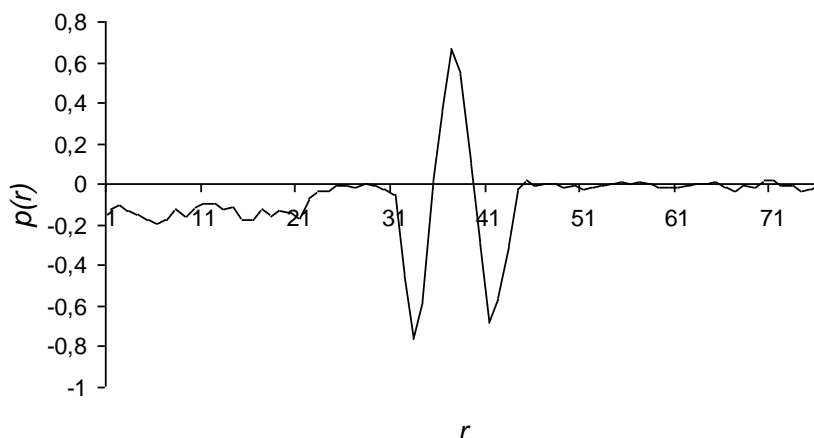
Parametr p jest wyznaczany dla każdego kolejnego położenia maski względem obrazu, czyli dla różnych wartości promienia r oraz kąta φ .

4.4.3. Detekcja linii na podstawie analizy zmienności parametru p

Gdy ustalona już jest wartość parametru p dla kolejnych położení maski względem obrazu, dokonuje się analizy przebiegu zmienności tego parametru. Jeśli dla danej wartości promienia r oraz kąta φ wartość parametru p jest większa od ustalonej wartości progowej $\theta_p = 0,5$, to na badanym obrazie znajduje się linia Kikuchiego. Szerokiej linii Kikuchiego odpowiada na wykresie parametru p np. Kilka kolejnych wartości parametru p większych od progu θ_p , spośród których wybierana jest największa wartość. Przykładowy przebieg wartości parametru p dla linii o różnej szerokości oraz ustalonym kącie φ przedstawiono na rysunkach (4.17 oraz 4.18).



Rys. 4.17. Przebieg parametru p w przypadku przejścia maski przez wąską linię Kikuchiego



Rys. 4.18. Przebieg parametru p w przypadku przejścia maski przez szeroką linię Kikuchiego

4.4.4. Wyznaczenie najbardziej prawdopodobnego położenia linii

Położenie każdej linii jest określone przez parę współrzędnych (r, φ) , które tworzą dwuwymiarową przestrzeń. Każdej linii odpowiada punkt lub zbiór kilku sąsiednich punktów. Dodatkowo, z każdym punktem związana jest wartość parametru p np. (patrz wzór 4.4). W przypadku wąskiej linii mamy do czynienia z pojedynczym punktem, zaś szerokiej linii odpowiada na ogół zbiór kilku blisko siebie położonych punktów. Dlatego w celu wyznaczenia najbardziej prawdopodobnego położenia linii, jest stosowany następujący algorytm:

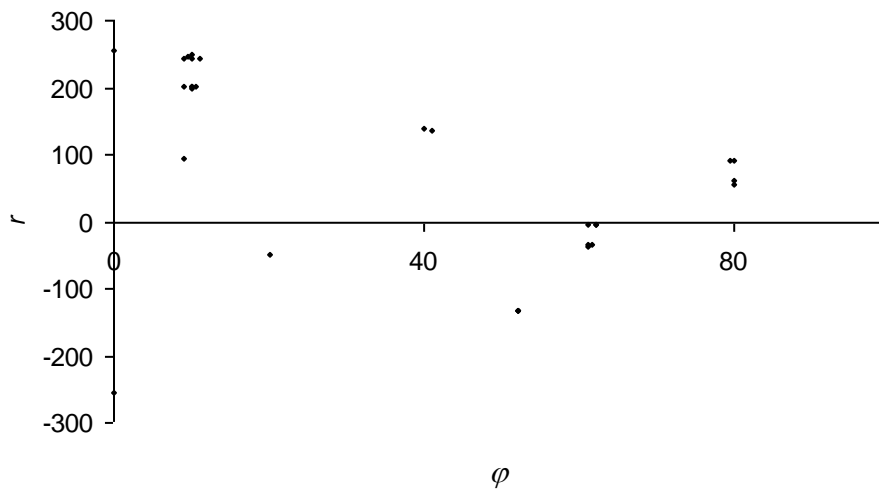
1. Powtarzaj dopóki nie przetworzono wszystkich punktów:

- przejdź do kolejnego nieprzetworzonego punktu $P = (r_i, \varphi_i)$
- znajdź zbiór $\Omega = \{P_k = (r_k, \varphi_k) : r_i - 2 \leq r_k \leq r_i + 2, \quad \varphi_i - 2 \leq \varphi_k \leq \varphi_i + 2\}$
- wyznacz położenie linii:

$$r_{lini} = \frac{\sum_{j \in \Omega} p_j r_j}{\sum_{j \in \Omega} p_j}, \quad \varphi_{lini} = \frac{\sum_{j \in \Omega} p_j \varphi_j}{\sum_{j \in \Omega} p_j} \quad (4.5)$$

gdzie p_j jest wartością parametru p odpowiadającą j -temu punktowi.

Spośród znalezionych linii wybierane są te, które stanowią pary linii równoległych. Parę linii Kikuchiego stanowią te dwie linie, które są nachylone pod tym samym kątem do przyjętej osi odniesienia oraz odległość między nimi zawiera się w przedziale $[\Delta r_{\min}, \Delta r_{\max}]$, określonym przez właściwości krystalograficzne badanego materiału, jak również bieżące ustawienia aparatury mikroskopowej. W ten sposób uzyskuje się zbiór znalezionych na obrazie par linii Kikuchiego.



Rys. 4.19. Przykładowy rozkład punktów na płaszczyźnie (r, φ)

4.5. Wyniki badań

Opisany powyżej algorytm został zaimplementowany w języku MATLAB. Zbadano właściwości algorytmu poprzez określenie wpływu wartości progu θ_p oraz rodzaju zastosowanej maski filtrującej na skuteczność wykrywania par linii Kikuchiego. Uzyskane wyniki zostały wykorzystane do optymalizacji wartości progu θ_p oraz wyboru maski filtrującej. Przyjęto typowe wartości $\Delta r_{\min}=20$ pikseli oraz $\Delta r_{\max}=120$ pikseli.

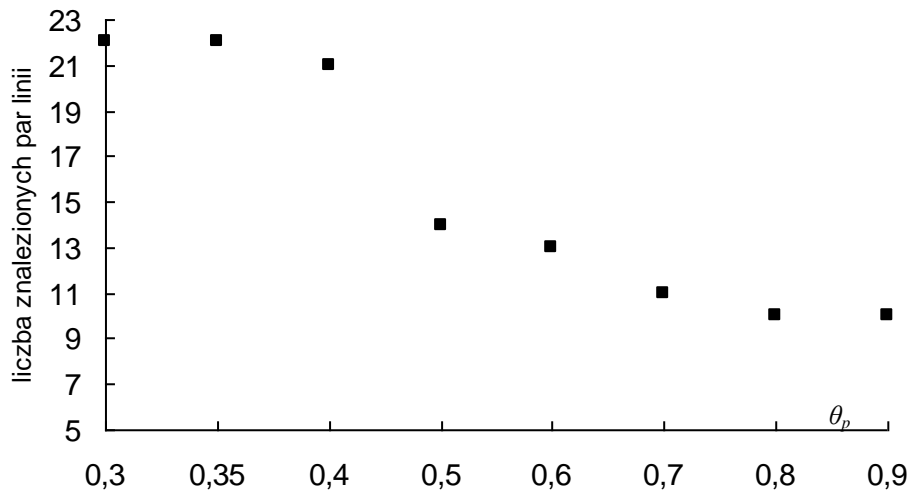
4.5.1. Wpływ wartości progu θ_p na skuteczność algorytmu

Zbadano wpływ wartości progowej θ_p na skuteczność działania algorytmu wyrażoną liczbą znalezionych par linii Kikuchiego oraz ilością błędnych decyzji. Badanie przeprowadzono na serii 30-tu obrazów testowych uzyskanych z mikroskopu elektronowego. Otrzymane wyniki zostały uśrednione i przedstawione w formie wykresu na rys. (4.20 oraz 4.21).

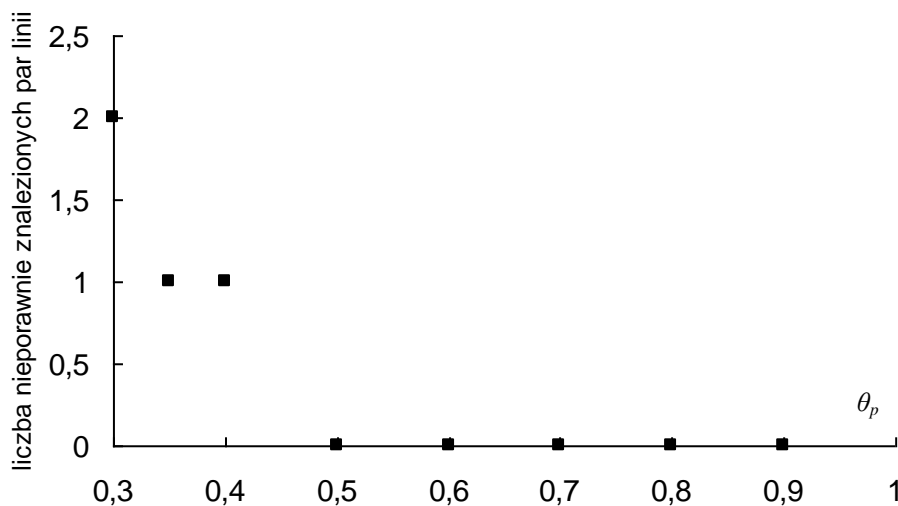
Na podstawie uzyskanych wyników przyjęto wartość progową $\theta_p = 0,5$. Dla tej wartości progu algorytm osiąga największą ilość wykrywanych par linii przy jednoczesnym braku błędnych decyzji. Brak błędnych decyzji jest istotny z punktu widzenia przeprowadzanych obliczeń orientacji krystalograficznych.

4.5.2. Wpływ zastosowanej maski filtrującej na skuteczność algorytmu

Przy ustalonej wartości $\theta_p = 0,5$ zbadano wpływ rodzaju zastosowanej maski filtrującej na skuteczność detekcji linii Kikuchiego. Badanie przeprowadzono na serii 30-tu obrazów testowych. Uzyskane wyniki zostały uśrednione i przedstawione w tabeli (4.2).



Rys.4.20. Liczba poprawnie znalezionych par linii na obraz w zależności od wartości progu θ_p



Rys.4.21. Liczba niepoprawnych linii na obraz w zależności od wartości progu θ_p

Tabela 4.2. Porównanie wpływu rodzaju maski filtrującej na liczbę znalezionych par linii

Rodzaj maski filtrującej	Liczba znalezionych par linii
Maska Sobela	13
Maska Perwitta	13
Maska Robertsa	9

Z przeprowadzonych badań wynika, że największą ilość znajdujących par linii uzyskano przy zastosowaniu maski Sobela lub Perwitta. Ostatecznie przyjęto, że algorytm wykorzystuje maskę Sobela.

4.5.3. Wpływ nierównomiernego oświetlenia obrazu na skuteczność algorytmu

Zbadano wpływ nierównomiernego oświetlenia obrazu źródłowego na skuteczność działania algorytmu. Badanie przeprowadzono na serii 20-tu obrazów testowych, przy optymalnej wartości progowej $\theta_p = 0,5$ oraz z zastosowaniem maski Sobela. Uzyskane wyniki zostały uśrednione i zebrane w tabeli (4.3).

Tabela 4.3. Wpływ nierównomiernego oświetlenia na skuteczność algorytmu

	Liczba poprawnie znalezionych par linii	Liczba niepoprawnie znalezionych par linii
Z wyrównaniem oświetlenia	13	0
Bez wyrównania oświetlenia	12	3

4.5.4. Porównanie skuteczności opracowanego algorytmu z algorytmem referencyjnym

Dla przyjętej wartości $\theta_p = 0,5$ przebadano skuteczność działania nowego algorytmu przez porównanie otrzymanych wyników z wynikami algorytmu automatycznej detekcji opartego na transformacie Hougha, opracowanego w Instytucie Metalurgii i Inżynierii Materiałowej Polskiej Akademii Nauk (IMIM-PAN). Badania przeprowadzono na serii 30-tu obrazów uzyskanych z mikroskopu elektronowego. Całkowita liczba wszystkich par linii na badanych 30-tu obrazach wynosi 312. Każdy z obrazów został poddany analizie za pomocą przedstawionego algorytmu oraz algorytmu bazującego na transformacie Hougha. Zbadano również, czy algorytmy sygnalizują obecność nieistniejących par linii. Otrzymane wyniki dla wszystkich obrazów zostały przedstawione w tabelach (4.4 i 4.5).

Jak widać, opracowany algorytm jest o wiele skuteczniejszy od referencyjnego. Podstawową wadą metody jest jednak jej 10-krotnie większa złożoność obliczeniowa i ponad 10-krotnie dłuższy czas obliczeń. Dla zaproponowanego algorytmu średni czas przetwarzania jednego obrazu jest bowiem równy 11200 milisekund, podczas gdy dla algorytmu IMIM-PAN – w przybliżeniu 1100 milisekund.

W kolejnych rozdziałach zaproponowano inne, szybsze metody detekcji linii Kikuchiego.

Tabela 4.4. Porównanie skuteczności algorytmów (uśrednione dla jednego obrazu)

Algorytm	Średnia liczba poprawnie znalezionych par linii w jednym obrazie	Średnia liczba stwierdzenia obecności nieistniejących par linii w jednym obrazie
Standardowy IMIM-PAN	7	1.20
Zaproponowany	9	0.07

Tabela 4.5. Zbiorcze wyniki dla 30-tu badanych obrazów

Algorytm	Liczba poprawnie znalezionych par linii	Liczba stwierdzenia obecności nieistniejących par linii
Standardowy IMIM-PAN	224 (71,8%)	36 (11.5%)
Zaproponowany	275 (88,1%)	2 (0.6%)

4.6. Podsumowanie

W niniejszym rozdziale zaprezentowano opracowaną, autorską metodę detekcji linii Kikuchiego, która bazuje na skanowaniu obrazu za pomocą maski złożonej z trzech linii równoległych. Charakteryzuje się ona bardzo dużą skutecznością detekcji linii w porównaniu z algorytmem referencyjnym z IMIM-PAN w Krakowie, jednak jest złożona i czasochłonna obliczeniowo. W następnym rozdziale będzie przedstawiona kolejna z opracowanych metod detekcji, nieco mniej dokładna niż wyżej zaproponowana, ale za to o wiele szybsza.

5. Algorytm 2. – modyfikacja transformacji Hougha

5.1. Wstęp

W niniejszym rozdziale przedstawiono kolejny z opracowanych algorytmów do detekcji linii Kikuchiego. Ponieważ metoda opracowana w rozdziale 4. charakteryzuje się długim czasem obliczeń, dlatego postawiono sobie za cel opracowanie metody o wiele szybszej, a równocześnie o zbliżonej dokładności. Założono, że pierwszym etapem algorytmu powinna być szybka detekcja linii metodą maksimum transformacji Hougha, a drugim – rzadziej wykonywana, wolniejsza weryfikacja wyników z etapu pierwszego, wykorzystująca metodę trzech linii, zaproponowaną w rozdziale 4. Zastosowano akumulatorową wersję transformacji Hougha, którą odpowiednio zmodyfikowano w celu zmniejszenia szumu w przestrzeni jej współczynników. Dzięki temu, do czasochłonnego etapu weryfikacji przechodzi mniej maksimumów, co znacznie skraca całkowity czas obliczeń.

5.2. Struktura algorytmu

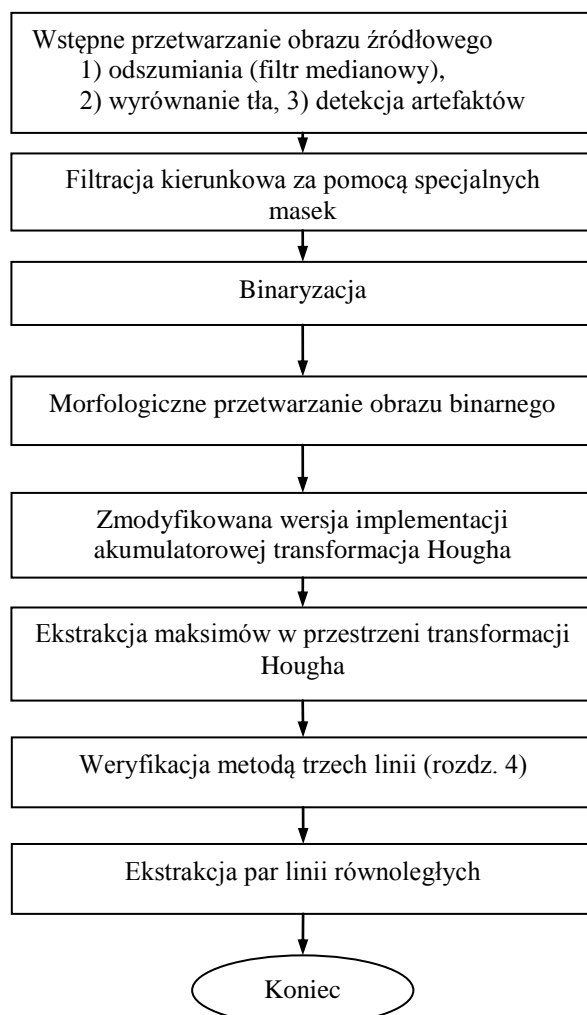
Schemat blokowy opracowanego algorytmu numer 2 jest przedstawiony na rysunku (5.1). Algorytm składa się z trzech części. W pierwszym kroku, stanowiącym wstępne przetwarzanie obrazu źródłowego, wykorzystywane są następujące operacje: usuwanie szumu, wyrównanie tła, detekcję artefaktów. W drugim kroku stosowana jest filtracja kierunkowa za pomocą zaproponowanych masek, binaryzacja oraz seria operacji morfologicznych. W kroku trzecim stosowana jest zmodyfikowana przez autora standardowa transformacja Hougha. Po zakończeniu przetwarzania dokonywana jest weryfikacja otrzymanych do tej pory wyników oraz końcowe wyodrębnienie par równoległych linii.

5.3. Wstępne przetwarzanie obrazu źródłowego

Obrazy źródłowe są na ogół zaszumione. Pozostawienie szumu powodowałoby, że dalsze przetwarzanie byłoby bardziej złożone obliczeniowo oraz istniałaby możliwość otrzymania mniej wiarygodnych wyników. Aby zredukować niekorzystny wpływ szumu, stosuje się, podobnie jak w poprzednim algorytmie, filtrację obrazu źródłowego za pomocą filtra medianowego. Kolejne kroki wstępnego przetwarzania obejmują, podobnie jak w poprzedniej metodzie, korekcję nierównomiernego tła obrazu oraz detekcję i usunięcie artefaktów.

5.4. Filtracja kierunkowa

Niektóre linie Kikuchiego byłyby trudne do wykrycia bez dodatkowego przetwarzania obrazu. Dlatego aby zwiększyć prawdopodobieństwo wykrycia słabo widocznych linii, odszumiony obraz jest filtrowany za pomocą czterech masek o następujących orientacjach: pozioma, pionowa, pod kątem 45° oraz 135° względem osi pionowej.



Rys. 5.1. Schemat blokowy algorytmu

Tabela 5.1. Pionowa maska

1	0	0	0	0	0	-1
0	0	0	0	0	0	0
0	0	0	0	0	0	0
1	0	0	0	0	0	-1
0	0	0	0	0	0	0
0	0	0	0	0	0	0
1	0	0	0	0	0	-1

Tabela 5.2. Maska zorientowana pod kątem 45°

0	0	0	0	-1	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	-1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0

Filtracja jest wykonywana jako dyskretny splot 2D. Maska pozioma jest uzyskiwana z maski pionowej w wyniku obrócenia jej o kąt 90° w kierunku przeciwnym do kierunku ruchu wskazówek zegara. Analogicznie uzyskiwane są maski zorientowane pod kątem 45° oraz 135° (w tym przypadku w celu obrócenia maski stosuje się interpolację liniową). W rezultacie filtracji kierunkowej otrzymuje się cztery obrazy, które następnie są oddzielnie przetwarzane. W wyniku zastosowania filtracji uzyskuje się:

- wzmocnienie słabo widocznych linii, dzięki czemu zwiększa się prawdopodobieństwo ich wykrycia;
- stłumienie linii znacząco różnych od bieżącej orientacji maski, przez co zmniejsza się prawdopodobieństwo uzyskania błędnych wyników na dalszym etapie pracy algorytmu.

Zastosowane maski zostały przedstawione w tabelach (5.1 oraz 5.2). Obraz źródłowy jest dwukrotnie filtrowany za pomocą każdej z czterech masek. Dwukrotna filtracja odpowiada estymacji drugiej pochodnej w kierunku odpowiadającym orientacji maski.

5.5. Binaryzacja

Ten etap polega na wydzieleniu z przefiltrowanego obrazu pikseli stanowiących obiekty (linie) oraz pikseli stanowiących tło. Zastosowana metoda progowania wykorzystuje automatyczny dobór wartości progu w oparciu o minimalizację entropii Renyi'ego [Sahh97]. Każdy piksel, którego wartość jest mniejsza niż wyznaczony próg, zostaje przypisany do tła obrazu, zaś piksele o wartości większej od progu zostają przypisane do obiektu.

5.6. Przetwarzanie obrazu binarnego

W celu lepszego przystosowania obrazu do dalszego przetwarzania, wykonywana jest następująca sekwencja operacji morfologicznych:

- Dylatacja** ma na celu usunięcie przerw w obiektach, które mogą powstać na etapie binaryzacji.
- Operacja ścienniania** powoduje, że wszystkie linie mają grubość jednego piksela. Zastosowano tu metodę opisaną w [Lam92].
- Usunięcie izolowanych pikseli** – te piksele nie przenoszą znaczącej ilości informacji o położeniu szukanych linii na obrazie.

- d) **Znalezienie najbardziej dopasowanych pikseli i estymacja orientacji poszczególnych linii.** Są to te piksele, które zapewniają najbardziej dokładną estymację parametrów linii. W przypadku linii prostych, najbardziej dopasowanymi pikselami są końce linii, które są znajdowane poprzez zbadanie sąsiedztwa każdego piksela. Jeśli w bezpośrednim sąsiedztwie badanego piksela znajduje się tylko jeden biały piksel, to badany piksel jest końcem linii. W rezultacie dla każdej linii otrzymuje się dwa piksele: $P_1=(x_1,y_1)$ oraz $P_2=(x_2,y_2)$. Piksele te zostają użyte do estymacji kierunku linii według poniższej formuły:

$$\varphi_k = \arctan\left(\frac{y_2 - y_1}{x_2 - x_1}\right) \text{ gdy } x_2 \neq x_1 \quad (5.1)$$

$$\varphi_k = 90^\circ \quad \text{gdy } x_2 = x_1$$



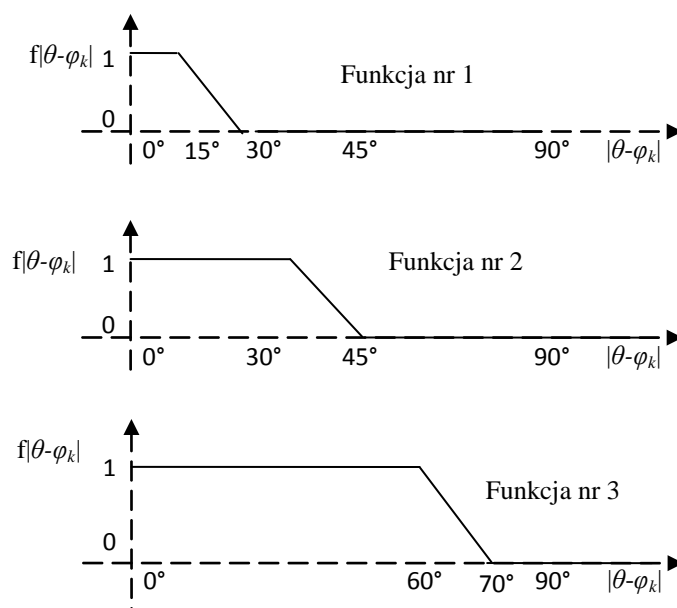
Rys. 5.2 Obraz binarny po operacji ścieniania

5.7. Zmodyfikowana transformacja Hougha

Po dokonaniu powyższych operacji obraz jest analizowany za pomocą odpowiednio zmodyfikowanej, akumulatorowej implementacji transformacji Hougha. Zaproponowana modyfikacja polega na wykorzystaniu estymowanej wcześniej informacji dotyczącej orientacji poszczególnych linii na obrazie (kąt φ_k). Dla każdej możliwej orientacji linii (θ) przechodzącej przez biały piksel obrazu binarnego, odpowiadająca wartość akumulatora jest zwiększana o wartość funkcji $f(|\theta - \varphi_k|)$. Kąt $\theta - \varphi_k$ jest kątem pomiędzy możliwą orientacją linii a estymowanym kątem nachylenia k -tej linii.

W celu wyznaczenia optymalnej funkcji $f(|\theta - \varphi_k|)$, zbadano jej trzy przykładowe przebiegi, przedstawione na rysunku (5.3).

Dla każdej z trzech funkcji $f(|\theta - \varphi_k|)$ zbadano średnią liczbę poprawnie znalezionych linii oraz średnią liczbę błędnych decyzji (stwierdzenie obecności linii, które w rzeczywistości na obrazie nie istnieją). Uzyskane wyniki zostały zaprezentowane w tabeli (5.4). Ostatecznie wybrano funkcję nr 2, gdyż daje ona najlepsze wyniki.



Rys. 5.3. Przykładowe przebiegi funkcji

Tabela 5.3. Algorytm zmodyfikowanej transformacji Hougha

<p>(x, y) – współrzędne niezerowego piksela na obrazie φ_k – estymowana orientacja linii, przechodząca przez dany piksel (x, y)</p> <hr/> <p>Wyzeruj wszystkie komórki akumulatora Dla każdego piksela o wartości 1 na obrazie binarnym powtarzaj w pętli</p> <pre> { dla $\theta = \varphi_k - 45^\circ$ do $\varphi_k + 45^\circ$ { $\rho = x \cdot \cos(\theta) + y \cdot \sin(\theta)$ $Acc(\theta, \rho) = Acc(\theta, \rho) + f(\theta - \varphi_k)$ } } </pre>
--

Tabela 5.4. Wyniki dla trzech różnych przebiegów funkcji z rysunku (5.3)

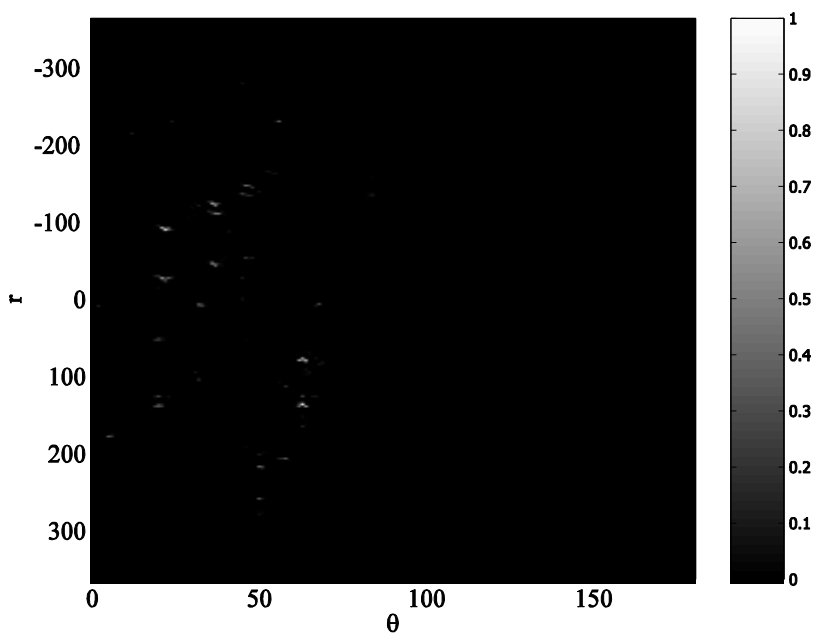
Funkcja	Poprawnie znalezione linie	Nieistniejące linie
Funkcja nr 1	16	3
Funkcja nr 2	24	6
Funkcja nr 3	25	10

W rezultacie, wartości komórek akumulatora odpowiadających orientacjom bliskim estymowanej orientacji linii są zwiększane o 1, zaś wartości komórek odpowiadających orientacjom znacząco różnym od estymowanej nie zostają zmienione. Procedura modyfikowania akumulatora w oparciu o estymowaną orientację linii prowadzi więc do znaczącej redukcji szumu obecnego w przestrzeni parametrów akumulatora. W rezultacie prawdopodobieństwo błędnych detekcji ulega znacznemu zmniejszeniu.

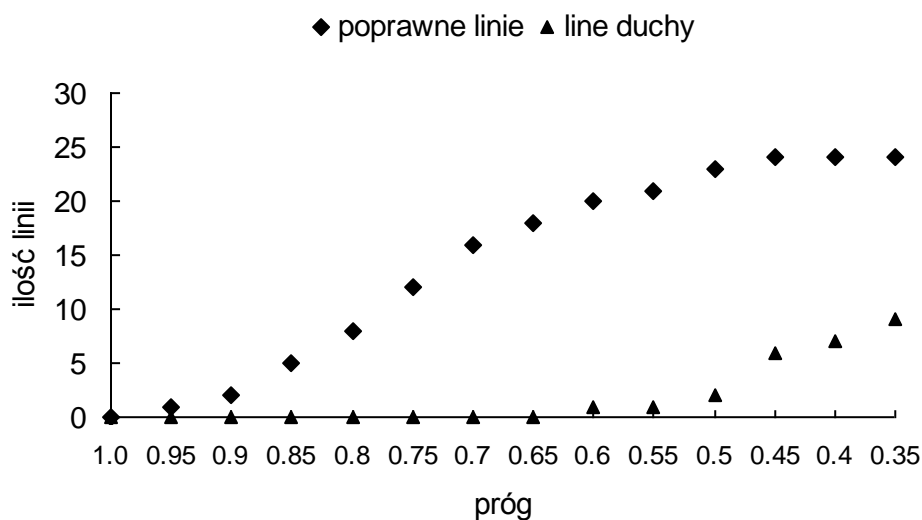
5.8. Ekstrakcja maksimum w przestrzeni transformacji Hougha

Kolejny krok przetwarzania polega na znalezieniu tych komórek akumulatora, których wartość przekracza pewien doświadczalnie wyznaczony próg.

W celu wyznaczenia wartości tego progu, zbadano ilość poprawnie znalezionych linii oraz ilość błędnie znalezionych linii w zależności od wartości progu. Uzyskane wyniki zostały przedstawione na rysunku (5.5). Na ich podstawie przyjęto wartość progu równą 0.45. Większa wartość progu prowadzi do mniejszej ilości poprawnie znalezionych linii, zaś mniejsza wartość powoduje znaczący wzrost liczby błędnych linii bez wyraźnego wzrostu liczby poprawnie znalezionych linii.



Rys. 5.4. Przykład zmodyfikowanej transformacji Hougha



Rys. 5.5. Skuteczność detekcji w zależności od wartości progu detekcji

5.9. Weryfikacja

W celu zmniejszenia prawdopodobieństwa detekcji nieistniejących linii, przeprowadza się dodatkową weryfikację wyników uzyskanych na podstawie zmodyfikowanej transformacji Hougha. Jeśli wartość komórki akumulatora mieści się w przedziale od 0.45 do 0.7, wtedy istnienie odpowiadającej tej komórce linii jest bezpośrednio weryfikowane na obrazie binarnym (w przypadku linii odpowiadającym komórkom o wartości powyżej 0.7 nie przeprowadza się dodatkowej weryfikacji z uwagi fakt, iż istnieje małe prawdopodobieństwo, że takiej komórce odpowiada „fałszywa” linia). W celu dokonania weryfikacji, tworzona jest specjalna maska złożona z trzech równoległych linii prostych. Środkowa linia posiada parametry wyznaczone na podstawie zmodyfikowanej transformacji Hougha. Pozostałe dwie linie są oddalone od środkowej prostej o pewną stałą odległość. Następnie z obrazu źródłowego zbierane są piksele leżące wzdłuż każdej z trzech prostych. Dla każdego z tych trzech zbiorów pikseli wyznaczany jest parametr i_k według następującego wzoru (N oznacza długość maski w pikselach):

$$v_k = \left(\max \left\{ \sum_{i=1}^N Y(i) \right\} \right) / N \quad k = 1, 2, 3 \quad (5.2)$$

gdzie:

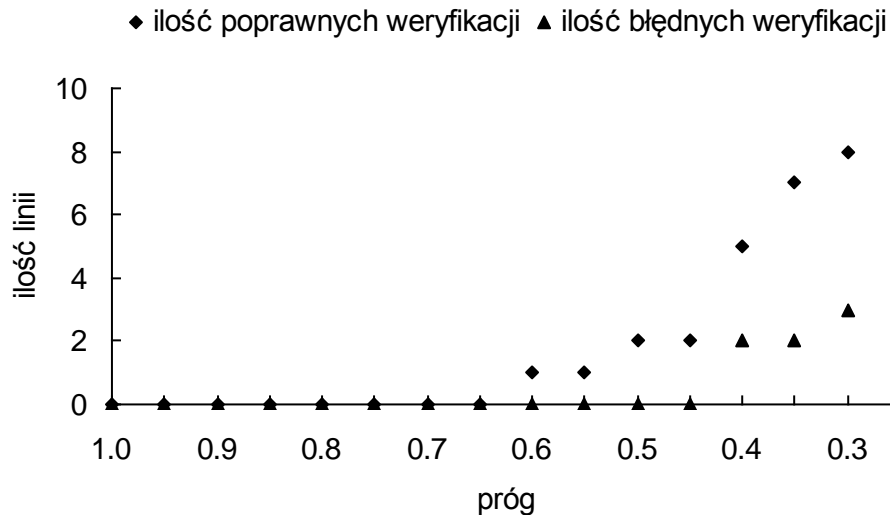
$$Y(i) = \begin{cases} 1 & \text{gdy } i\text{-ty piksel na prostej jest równy } 1 \\ -1 & \text{gdy } i\text{-ty piksel na prostej jest równy } 0 \end{cases}$$

Następnie wyznaczany jest parametr p :

$$p = \frac{1}{3}(v_2 - v_1 - v_3) \quad (5.3)$$

Jego wartość zmienia się w zakresie od -1 do 1. Jest ona bliska -1, gdy wzdłuż maski występuje długi ciąg pikseli o wartości 0. W przypadku, gdy maska pokrywa się z linią Kikuchiego, wzdłuż niej występuje długi ciąg pikseli o wartości 1, a wartość p jest bliska 1. Zatem gdy wartość parametru p dla danego położenia na obrazie (r, θ) jest powyżej ustalonej wartości progu, to algorytm przyjmuje, że w tym miejscu występuje linia Kikuchiego.

Wartość progu została wyznaczona doświadczalnie: dla różnych wartości progu zbadano średnią na obraz liczbę poprawnie i niepoprawnie zweryfikowanych linii Kikuchiego. Uzyskane wyniki są zaprezentowane na rysunku (5.6). Na ich podstawie przyjęto wartość progu równą 0.35. Wartość ta daje maksymalną liczbę poprawnie zweryfikowanych linii przy jednoczesnej minimalizacji liczby błędnie zweryfikowanych linii.



Rys. 5.6. Skuteczność weryfikacji w zależności od wartości progu

5.10. Wyodrębnienie par linii równoległych

Ostatnim etapem pracy algorytmu jest znalezienie par linii równoległych. W tym celu wykorzystywana jest taka sama metoda jak w rozdziale 4.4.4.

5.11. Wyniki badań

Zaproponowany algorytm został zaimplementowany w językach MATLAB oraz C++. Implementacja w MATABIE została przetestowana na 30-tu obrazach, na których znajdowało się łącznie 312 par linii Kikuchiego. Uzyskane wyniki zostały porównane z dwoma algorytmami referencyjnymi. Algorytm pierwszy, zaimplementowany i przetestowany przez autora, był oparty o standardową transformację Hougha bez dodatkowych procedur przetwarzania obrazów. Algorytm drugi pochodzi z Instytutu Metalurgii i Inżynierii Materiałowej Polskiej Akademii Nauk w Krakowie. Zmierzoną całkowitą liczbę poprawnie i niepoprawnie znalezionych par linii Kikuchiego przez każdy z algorytmów. Uzyskane wyniki zostały zaprezentowane w tabeli (5.5). Z przeprowadzonych badań wynika, iż zaproponowany obecnie algorytm charakteryzuje się większą skutecznością detekcji par linii oraz mniejszą ilością błędnie znajdowanych par linii niż algorytm z IMIM-PAN. Jest on co prawda trochę mniej dokładny niż metoda opracowana w rozdziale 4., ale za to zdecydowanie, prawie 10-krotnie od niej szybszy.

Tabela 5.5. Wyniki badań skuteczności wykrywania par linii Kikuchiego

Algorytm	Liczba poprawnie znalezionych par linii	Liczba niepoprawnie znalezionych par linii	Czas wykonania [s]
1. Standardowa transformacja Hougha	212 (67.9%)	28 (9.0%)	0.4
2. IMIM-PAN	224 (71.8%)	36 (11.5%)	1.0
3. Zaproponowany w rozdz. 4	275 (88.1%)	2 (0.6%)	11.2
4. Zaproponowany obecnie	261 (83.7%)	2 (0.6%)	1.45

W celu zbadania wpływu modyfikacji wprowadzonej do transformacji Hougha, zbadano zachowanie całej metody detekcji linii w dwóch przypadkach: z zastosowaniem standardowej transformacji Hougha oraz jej zmodyfikowanej wersji. Uzyskane wyniki zaprezentowano w tabeli (5.6). Jednoznacznie z nich wynika, że zastosowanie modyfikacji przyczyniło się do skrócenia całkowitego czasu pracy algorytmu prawie czterokrotnie. Jest to spowodowane tym, że w przestrzeni argumentów zmodyfikowanej transformacji Hougha występuje znacznie mniejsza liczba maksimumów nieodpowiadających rzeczywistym liniom na obrazie. Dzięki temu do czasochłonnego etapu weryfikacji przechodzi mniej maksimumów, co znacznie skraca całkowity czas obliczeń.

Tabela 5.6. Porównanie dwóch odmian opracowanej metody dla różnych rodzajów transformacji Hougha

Algorytm transformacji Hougha	Liczba poprawnie znalezionych par linii	Liczba niepoprawnie znalezionych par linii	Czas wykonania [s]
1. Standardowy	261 (83.7 %)	2 (0.6 %)	5.23
2. Zmodyfikowany	261 (83.7 %)	2 (0.6 %)	1.45

Zbadano również, jaką ilość czasu zajmuje wykonanie poszczególnych etapów algorytmu. W tabeli (5.7) został przedstawiony czas średni wykonania. Uzyskane wyniki posłużą do dalszej optymalizacji pracy algorytmu pod kątem szybkości działania.

Tabela 5.7. Czas wykonania poszczególnych etapów algorytmu

Etap	Implementacja w języku MATLAB [s]	Implementacja w języku C++ [s]
Odszumianie	0.3 (2 %)	0.03 (2 %)
Filtracja kierunkowa	0.6 (4 %)	0.06 (4 %)
Binaryzacja	0.14 (1 %)	0.01 (1 %)
Przetwarzanie obrazu binarnego	1.5 (10 %)	0.15 (10 %)
Zmodyfikowana transformacja Hougha	3 (21 %)	0.3 (21 %)
Weryfikacja wyników	9.1 (62 %)	0.9 (62 %)
Razem	14.5 (100 %)	1.45 (100 %)

5.12. Podsumowanie

W rozdziale tym przedstawiono kolejną z opracowanych metod detekcji linii Kikuchiego. Metoda ta wykorzystuje akumulacyjną implementację transformacji Hougha, uzupełnioną o zaproponowaną przez autora modyfikację. Dzięki modyfikacji osiągnięto znaczne skrócenie całkowitego czasu pracy algorytmu detekcji linii (patrz tab. 5.6). W porównaniu do metody z rozdziału 4., metoda ze zmodyfikowaną transformacją Hougha charakteryzuje się co prawda niższą skutecznością detekcji (nie 88.1%, tylko 83.7% znalezionych linii – patrz tab. 5.5), ale

jest prawie dziesięciokrotnie od niej szybsza. Dalej jest jednak ona dokładniejsza niż algorytm referencyjny.

W następnym rozdziale zostanie przedstawiony ostatni, najlepszy z zaproponowanych algorytmów, któremu udało się połączyć dokładność z szybkością. Wykorzystuje on zaawansowane metody wstępnego przetwarzania sygnałów, szybką implementację transformacji Hougha za pomocą FFT oraz szybki klasyfikator neuronowy.

6. Algorytm 3. – zastosowanie zaawansowanych metod przetwarzania obrazów i klasyfikacji

6.1. Wstęp

W niniejszym rozdziale przedstawiono ostatni z opracowanych algorytmów detekcji linii Kikuchiego. Stanowi on udoskonalenie metody zaproponowanej w rozdziale 5. Wprowadzone modyfikacje obejmują:

- dodanie falkowej metody *Neighshrink* odszumiania obrazu źródłowego (rozd. 2.4.2 i 2.5.5),
- zwiększenie kontrastu za pomocą krzywej tonalnej i transformacji *curvelet* (rozd. 2.4.4),
- użycie szybkiej transformacji Radona z algorytmem FFT do wyznaczenia transformacji Hougha (rozd. 2.2 i 2.3.2);
- zastosowanie zaawansowanych metod do klasyfikacji maksimów współczynników transformacji Hougha w celu wyodrębnienia należących lub nienależących do linii;
- dodanie etapu końcowego, polegającego na poszukiwaniu ewentualnych brakujących do par linii metodą przestrajanego filtra kierunkowego (rozd. 2.6).

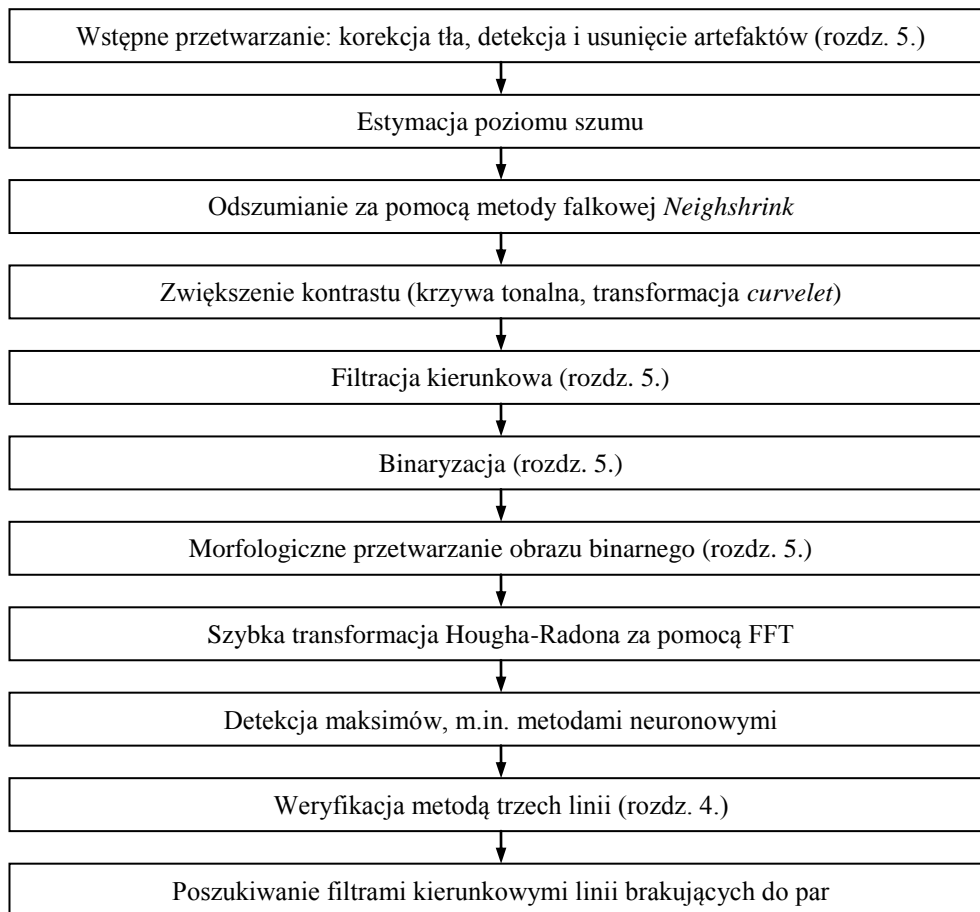
6.2. Struktura algorytmu

Na rysunku (6.1) przedstawiono schemat blokowy zaproponowanego algorytmu. Zaznaczono na nim, które z operacji występowały w metodach opracowanych w rozdziale 4 i 5. W dalszej części rozdziału omówiono szczegółowo każdy z modułów.

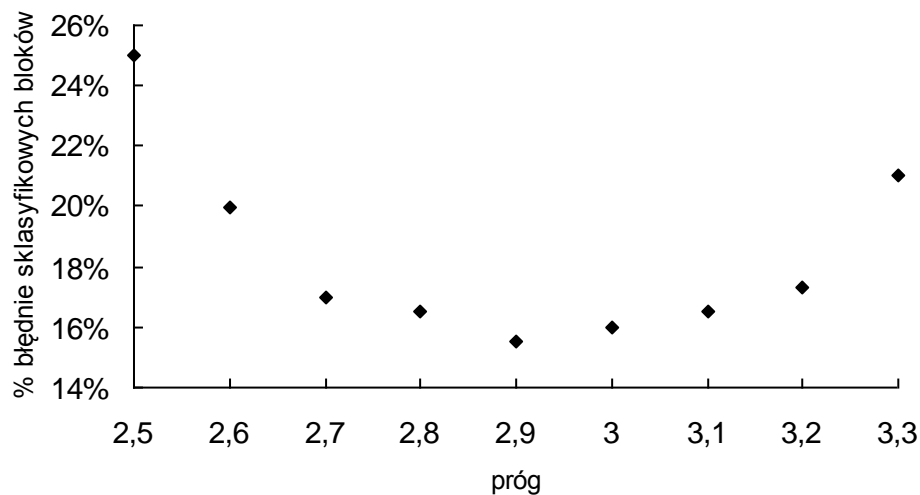
6.3. Estymacja poziomu szumu

Po dokonaniu korekcji nierównomiernego oświetlenia, przeprowadza się estymację odchylenia standardowego szumu obecnego na obrazie źródłowym. W tym celu należy wyodrębnić z obrazu te rejony, które zawierają szum, a jednocześnie nie zawierają linii. W tym celu najpierw odejmuje się od obrazu globalną wartość średnią, a następnie obraz jest dzielony na kwadratowe bloki o wymiarze 4x4 pikseli. Każdy z tych bloków jest następnie poddawany analizie za pomocą 2D FFT. Możliwe są dwa przypadki: pojedynczy blok zawiera linię Kikuchiego lub nie. W celu odróżnienia tych dwóch przypadków wprowadza się następujący parametr:

$$v = F(0,0)^2 + F(0,1)^2 + F(1,0)^2 + F(1,1)^2 \quad (6.1)$$



Rys. 6.1. Schemat algorytmu



Rys. 6.2. Ilość błędnych decyzji w zależności od wartości progów

gdzie $F(i,j)$ oznacza wartość współczynnika 2D FFT. Jeśli ν jest większy niż wartość progowa, wtedy odpowiedni blok zawiera linię Kikuchiego. Wartość tego progów jest wyznaczona na podstawie poprawnie i niepoprawnie sklasyfikowanych bloków. Z rysunku (6.2) wynika, że optymalna wartość progów wynosi 2.9.

Ostatecznie cały obraz zostaje podzielony na dwa podzbiory: pierwszy zawiera bloki z szumem nie zawierające linii, a drugi – bloki z liniami. Podzbiór zawierający tylko szum jest

użyty do estymacji odchylenia standardowego (σ) szumu. Znajomość tej wartości jest potrzebna podczas odszumiania obrazu w dziedzinie współczynników transformacji falkowej.

6.4. Odszumianie

Obecność szumu na obrazach przyczynia się do zwiększenia prawdopodobieństwa wykazywania istnienia linii, których w rzeczywistości nie ma na obrazie. Dlatego konieczny jest bardzo staranny dobór odpowiedniej metody odszumiania obrazu. Dobra metoda powinna charakteryzować się wysoką skutecznością redukcji szumu, a jednocześnie nie powinna powodować nadmiernego rozmycia słabych linii oraz nie powinna wprowadzać do obrazu dodatkowych artefaktów.

W celu znalezienia najlepszej metody odszumiania obrazów mikroskopowych, przetestowano kilka alternatywnych, nowoczesnych podejść. Zastosowano następującą metodologię badań: najpierw utworzono sztuczny, nie zaszumiany obraz *referencyjny*, zawierający kilkadziesiąt linii o różnej długości, szerokości oraz kontraście w stosunku do tła. Następnie do tego obrazu dodawano szum wydzielony z rzeczywistych obrazów mikroskopowych. Tak zaszumiony obraz poddano następnie operacji odszumiania za pomocą różnych metod. Jego skuteczność zmierzono za pomocą współczynnika PSNR (*Peak Signal to Noise Ratio*) w odniesieniu do obrazu referencyjnego. Początkowa wartość PSNR dla obrazu referencyjnego wynosiła $PSNR_{in}=21$ dB.

Zbadano następujące metody: uśrednianie pikseli w oknie 3x3, filtrację medianową, adaptacyjną filtrację Wienera, liniową i nieliniową zespoloną dyfuzję [Gilb04], automatyczną redukcję szumu z wykorzystaniem pochodnych czwartego rzędu (PDE) [You00], odszumianie za pomocą transformacji *curvelet* [Star02], odszumianie za pomocą standardowej transformacji falkowej [Dono95a] oraz odszumianie falkowe z użyciem sąsiadujących współczynników (*Neighshrink*) [Chen04]. Otrzymane wyniki są zaprezentowane w tabeli (6.1).

W przypadku standardowej transformacji falkowej oraz metody *Neighshrink* zbadano skuteczność odszumiania dla następujących falek: Haar, Beylkin, Coiflet, Daubechies, Symmlet, Vaidyanathan oraz Battle. W tabeli (6.2) przedstawiono maksymalne wartości PSNR obrazu, otrzymane dla poszczególnych typów falek. Ponieważ metoda *Neighshrink* z falką Haara dała najlepsze rezultaty, została ona zastosowana w dalszej części badań w tworzonym algorytmie detekcji linii Kikuchiego.

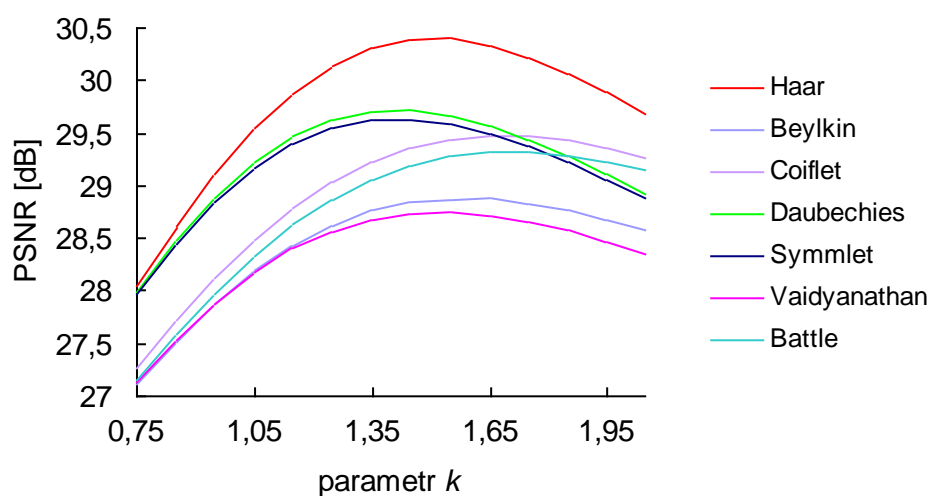
Tabela 6.1. Skuteczność odszumiania dla poszczególnych metod

Metoda	PSNR _{out} [dB]
PDE [You00]	24.81
Liniowa zespolona dyfuzja [Gilb04]	25.23
Uśrednianie pikseli	25.63
Filtracja medianowa	25.78
Nieliniowa zespolona dyfuzja [Gilb04]	26.12
Odszumianie za pomocą transformacji <i>curvelet</i> [Star02]	28.03
Adaptacyjna filtracja Wienera [Lim90]	28.60
Standardowa transformacja falkowa (Haar) [Dono95a]	29.39
Odszumianie metodą <i>Neighshrink</i> (Haar) [Chen04]	30.41

Tabela 6.2. Skuteczność odszumiania dla metod falkowych

Rodzaj falki	PSNR _{out} [dB]	
	Standard	<i>Neighshrink</i>
Haar	29.39	30.41
Beylkin	27.87	28.88
Coiflet	28.48	29.48
Daubechies	28.72	29.70
Symmlet	28.63	29.64
Vaidyanathan	27.74	28.73
Battle	28.32	29.33

Na rysunku (6.3) przedstawiono szczegółowe wyniki badania skuteczności odszumiania w metodzie *Neighshrink* dla różnych wartości parametru k (patrz wzór 2.56).



Rys.6.3. Skuteczność odszumiania dla różnych falek

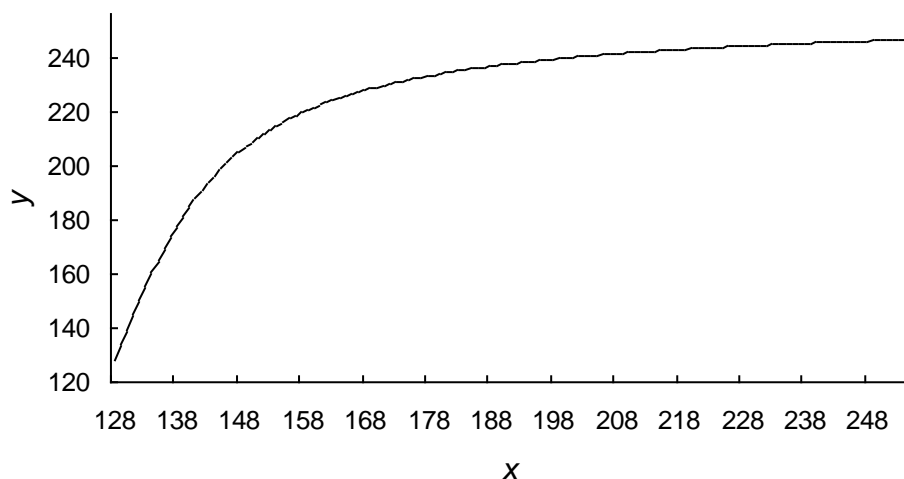
6.5. Zwiększenie kontrastu

Niektóre linie Kikuchiego charakteryzują się niewielkim kontrastem w stosunku do tła obrazu. Dlatego, aby ułatwić ich detekcję, należy zastosować operację selektywnego zwiększenia kontrastu.

6.5.1. Zwiększenie kontrastu za pomocą modyfikacji krzywej tonalnej

Modyfikacja krzywej tonalnej jest szybkim sposobem na zmianę rozkładu wartości pikseli. Aby uniknąć zwiększania szumu, dokonuje się modyfikacji tylko tych pikseli, które leżą wewnątrz bloków 4x4, poprzednio zaklasyfikowanych jako zawierające linie. Wartości pikseli są odwzorowywane za pomocą następującego przekształcenia (rys. 6.4):

$$y = 128 + 127 \cdot \frac{a \tan(m \cdot (x - 128))}{a \tan(128)} \quad (6.2)$$



Rys. 6.4. Przebieg funkcji przejścia dla $x > 128$

Jeśli wartość parametru m jest zbyt duża, to następuje zbyt duże wzmocnienie możliwych artefaktów powstałych w wyniku odsumowania za pomocą transformacji falkowej. Jeśli z kolei wartość parametru m jest zbyt mała, to ma miejsce zbyt słabe zwiększenie kontrastu. Dlatego przyjęto optymalną wartość $m = 0.07$. Wartości powyższego przekształcenia są obliczane i zaokrąglane z góry, a następnie zapamiętane w tablicy liczb całkowitych. Następnie, dla kolejnych pikseli ich nowa wartość jest odczytywana z tej tablicy.

Na podstawie rysunku (6.4) można zauważyć, że zastosowane przekształcenie daje największy wzrost kontrastu dla pikseli położonych w pobliżu wartości 128. Jest to ten obszar, gdzie znajdują się słabo widoczne linie Kikuchiego.

6.5.2. Zwiększenie kontrastu za pomocą transformacji *curvelet*

Z uwagi na ograniczone możliwości zwiększenia kontrastu za pomocą modyfikacji krzywej tonalnej, stosowane jest dodatkowo zwiększenie kontrastu za pomocą transformacji *curvelet* [Star03]. Transformacja ta jest dobrze dopasowana do reprezentowania obrazów zawierających linie oraz krawędzie (odpowiednie współczynniki transformacji mają relatywnie duże wartości). Dlatego jest ona z powodzeniem stosowana do przetwarzania zwłaszcza zaszumionych obrazów. Jej użycie w kontekście detekcji linii Kikuchiego pozwala na zwiększenie kontrastu obrazu, przy jednocześnie nieznacznym zwiększeniu poziomu pozostałego szumu.

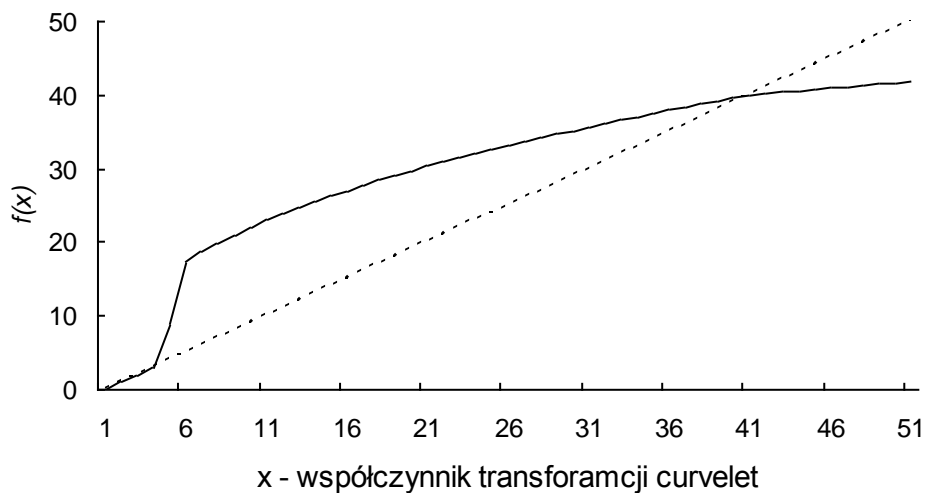
Zastosowanie transformacji *curvelet* do zwiększenia kontrastu przebiega według następującego scenariusza:

- wyznaczenie transformacji *curvelet* obrazu;
- modyfikacja współczynników transformacji za pomocą nieliniowej funkcji $f(x)$;
- wyznaczenie transformacji odwrotnej na podstawie zmodyfikowanych współczynników.

Do modyfikacji współczynników transformacji użyto następującej nieliniowej funkcji $f(x)$ [Star03]:

$$f(x) = \begin{cases} 1 & \text{jeśli } x < c \\ \frac{x-c}{c} \left(\frac{m}{c}\right)^p + \frac{2c-x}{c} & \text{jeśli } c \leq x < 2c \\ \left(\frac{m}{x}\right)^p & \text{jeśli } 2c \leq x < m \\ \left(\frac{m}{x}\right)^s & \text{jeśli } x \geq m \end{cases} \quad (6.3)$$

W równaniu (6.3) parametr p określa stopień nieliniowości, a parametr s – stopień kompresji zakresu. Wartość $s > 0$ powoduje wzmacnianie najsłabszych linii. Z kolei c jest współczynnikiem normalizującym, zaś parametr m – stopniem, w jakim będą wzmacniane współczynniki transformacji. Parametr c jest powiązany z uprzednio estymowaną wartością odchylenia standardowego. Jeśli c jest kilkakrotnie większe σ od (np. $c = 3\sigma$), to szum nie będzie wzmacniany. Wartość m jest wyznaczana na podstawie wartości maksymalnej M współczynników transformacji w danym podpaśmie. Przyjęto, że $m = k \cdot M$, gdzie k jest dobierane eksperymentalnie.



Rys. 6.5. Przebieg funkcji $f(x)$

Optymalne wartości parametrów (p , s , k), dobrano eksperymentalnie. Ponieważ zwiększenie kontrastu jest szczególnie istotne w przypadku słabo widocznych linii, dlatego do badania wybrano zbiór obrazów, na których jest stosunkowo duża liczba takich linii. Jako kryterium oceny jakości zwiększenia kontrastu przyjęto osiągnięcie przez jak największą liczbę linii ustalonego minimalnego kontrastu względem tła obrazu. Kontrast ten jest zdefiniowany następująco:

$$q = \frac{A_{lini}}{128} \quad (6.4)$$

gdzie A_{lini} jest wartością średnią pikseli położonych wzdłuż linii, zaś wartość 128 odpowiada poziomowi tła obrazu. Jego wartość jest gwarantowana przez operację wyrównania nierównomiernego tła obrazu. Przyjęto, że kontrast (q) każdej linii na obrazie powinien

wynosić co najmniej 1.15. Zmieniając iteracyjnie wartości parametrów (p, s, k), zmierzono na badanych obrazach ilość linii spełniających przyjęte kryterium kontrastu. Jako optymalny przyjęto ten zestaw parametrów, dla którego ilość linii spełniających to kryterium jest największa. Optymalne wartości wynoszą: $p=0.5, s=0.6, k=0.45$. Przykładowy kształt funkcji $f(x)$ został przedstawiony na rysunku (6.5) ($c = 3, m = 30, p = 0.5, s = 0.6$).

6.6. Filtracja kierunkowa

Wstępnie przetworzony obraz jest następnie dwukrotnie filtrowany za pomocą zbioru masek, nachylonych pod kątami $0^\circ, 45^\circ, 90^\circ$ oraz 135° do osi pionowej. W rezultacie powstają cztery obrazy, które są dalej niezależnie przetwarzane. Operację tę opisano już w algorytmie przedstawionym w rozdziale 5.

6.7. Binaryzacja

Na tym etapie każdy z czterech obrazów jest poddawany operacji binaryzacji. Zastosowane podejście polega na użyciu progu, a następnie przyporządkowaniu każdemu pikselowi obrazu wartości 0 lub 1 w zależności od tego, czy wejściowa wartość piksela jest większa od progu, czy nie. Wartość progu jest wyznaczana globalnie dla całego obrazu na podstawie jego entropii [Sahh97].

6.8. Przetwarzanie obrazu binarnego

W celu lepszego dostosowania obrazów binarnych do analizy za pomocą transformacji Hougha, zastosowano, w sposób zbliżony jak w rozdziale 5., sekwencję następujących operacji morfologicznych:

- dylatacja – wypełnienie możliwych przerw w liniach. Jako element strukturalny użyto kwadratowej maski o wymiarach 5 na 5 pikseli.
- operacja ścieniania – ma na celu uzyskanie linii o grubości jednego piksela;
- usunięcie pojedynczych izolowanych pikseli – te piksele praktycznie nie przenoszą informacji o położeniu linii na obrazie.

6.9. Transformacja Hougha

Uzdatnione obrazy binarne są następnie przetwarzane za pomocą szybkiej implementacji transformacji Hougha-Radona, wykorzystującej algorytm FFT. Podejście takie było opisane w rozdziale 2. (rozd. 2.2 i 2.3.2).

6.10. Detekcja maksimów w przestrzeni transformacji Hougha

Kolejnym krokiem jest detekcja lokalnych maksimów występujących w przestrzeni transformacji Hougha. Maksima te, na ogół, reprezentują linie Kikuchiego na obrazie. Jednak niektóre maksima nie zawsze odpowiadają istniejącym liniom. Są one wynikiem przypadkowego ułożenia się pewnej liczby pikseli na obrazie binarnym wzdłuż linii prostej. Dodatkowo, prawdopodobieństwo detekcji linii w oparciu o maksima w przestrzeni transformacji Hougha nie jest jednakowe na całym obszarze obrazu [Hans97]. Aby to wyjaśnić, wprowadzono pojęcie *wsparcia* dla linii lub ściślej – dla danej komórki transformacji Hougha. Wsparciem jest zbiór tych wszystkich pikseli, które w czasie działania transformacji Hougha wnoszą swój wkład do danej komórki. Obszar wsparcia może być w przybliżeniu wyznaczony za pomocą odwrotnej transformacji Hougha [Kesi00] i w ogólności przyjmuje on kształt zgiętego krawata [Prin92, Ratl81]. Rozmiarem wsparcia jest liczba stanowiących go pikseli. Innymi słowy, rozmiar wsparcia odpowiada maksymalnej możliwej wartości danej komórki transformacji Hougha. Z uwagi na fakt, iż obszar obrazu jest ograniczony, rozmiar wsparcia nie jest jednakowy dla wszystkich komórek transformacji Hougha. Dla linii równoległych do krawędzi obrazu jest on zawsze równie duży, jeżeli jednak nie są one tak usytuowane, to znaczenie ma ich położenie względem środka obrazu: im dalej od środka, tym rozmiar wsparcia jest mniejszy. W efekcie prawdopodobieństwo detekcji linii położonych dalej od środka obrazu jest mniejsze niż linii przebiegających w pobliżu jego środka. Sama filtracja w przestrzeni Hougha nie powoduje ograniczenia tego zjawiska [Leav92, Veen81]. W celu osłabienia tego niekorzystnego efektu, wszystkie komórki transformacji podlegają normalizacji. W tym celu wyznacza się transformację Hougha obrazu, którego wszystkie piksele są ustawione na 1 ($HTW(\rho, \theta)$). Następnie każda komórka transformacji Hougha przetwarzanego obrazu ($HT(\rho, \theta)$) jest dzielona przez odpowiadającą jej komórkę transformacji $HTW(\rho, \theta)$:

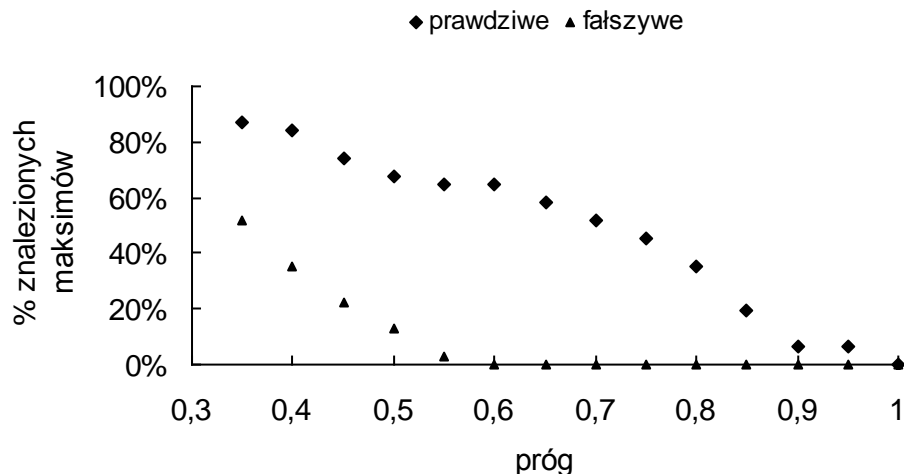
$$HT(\rho, \theta) = \frac{HT(\rho, \theta)}{HTW(\rho, \theta)} \quad (6.5)$$

W dalszej kolejności dokonuje się ekstrakcji maksimów z transformacji Hougha.

Zbadano następujące zaproponowane przez autora metody detekcji lokalnych maksimów: *proste progowanie* (ST), *analizę lokalnego kontrastu* (LCA), *klasyfikację cech* (FC) oraz *analizę profilu* (PPA).

6.10.1. Proste progowanie (ST)

Najprostszą metodą ekstrakcji lokalnych maksimów w przestrzeni transformacji Hougha jest progowanie. Wszystkie komórki, których wartość jest większa od pewnego progu, są uznawane za maksima odpowiadające liniom prostym na obrazie. Jednak nie wszystkie maksima odpowiadają istniejącym liniom. Optymalna wartość progu jest wyznaczana empirycznie na podstawie liczby znalezionych maksimów odpowiadających istniejącym liniom (maksima „prawdziwe”) oraz liczby maksimów, które nie odpowiadają liniom (maksima „fałszywe”).



Rys. 6.5. Procent znalezionych maksimów

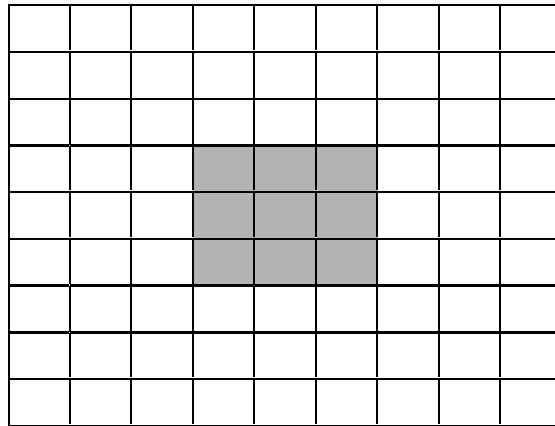
Przyjęto, iż współczynnik błędów nie powinien przekroczyć 30%. Jest to wartość względnie duża, jednak na tym etapie korzystniej jest znaleźć większą liczbę „fałszywych” maksimów i jednocześnie większą liczbę istniejących linii. „Fałszywe” maksima, związane z nieistniejącymi liniami zostają odfiltrowane na dalszym etapie pracy algorytmu. Dla przyjętego współczynnika błędów optymalna wartość progu wynosi 0.45.

Problemem związanym ze stosowaniem tej metody jest to, iż w idealnym przypadku maksima linii powinny mieć kształt wysokich szpilek, a pozostałe komórki transformacji Hougha powinny mieć niewielką wartość. Jednak w rzeczywistym przypadku maksima są rozmyte oraz otoczone obszarami o relatywnie dużych wartościach. Spowodowane jest to faktem, że większość linii na obrazie binarnym jest złożona z pionowych i poziomych odcinków. W dodatku, niektóre linie, jakie wizualnie wydają się być ciągłe, w rzeczywistości składają się z kilku odcinków. Całkowita długość linii może być rzędu 100 pikseli, podczas gdy długość każdego z odcinków składających się na całą linię może być rzędu 30 pikseli. Powoduje to rozmycie maksimum odpowiadającego takiej linii oraz obniżenie jego wysokości. Prowadzi to do konieczności stosowania niższych wartości progu, a to z kolei wiąże się z detekcją większej liczby „fałszywych” maksimów. Pomimo tych wad, największą zaletą metody detekcji linii w dziedzinie współczynników transformacji Hougha-Radona, wykorzystującej FFT jest jej szybkość.

Jedną z możliwych odmian tej metody jest progowanie z lokalnym uśrednianiem (LAT). W tym przypadku przeprowadza się uśrednianie kilku sąsiednich komórek transformacji Hougha i dopiero ta średnia podlega progowaniu. Otrzymane wyniki są ilościowo i jakościowo zbliżone do tych uzyskanych za pomocą prostego progowania. Różnica polega na tym, iż zastosowanie lokalnego uśredniania daje nieznacznie mniejszą liczbę „fałszywych” detekcji.

6.10.2. Analiza lokalnego kontrastu (LCA)

Metoda *analizy lokalnego kontrastu* polega na porównaniu kontrastu pomiędzy dwoma przyległymi do siebie obszarami (patrz rys. 6.6).

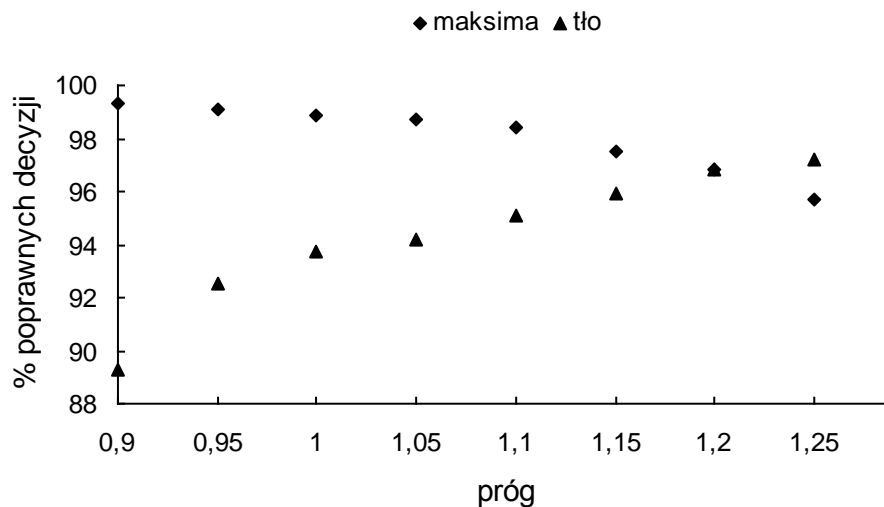


Rys.6.6. Maska użyta w metodzie LCA

Istota metody jest następująca: Kwadratowa maska jest przesuwana nad wszystkimi komórkami transformacji Hougha. Jej wysokość i szerokość wynosi 9 pikseli. Maska jest podzielona na dwa obszary: wewnętrzny, składający się z 9 pikseli (szary obszar na rysunku 6.6) oraz zewnętrzny, składający się ze wszystkich pozostałych pikseli (białe komórki na rysunku 6.6). Dla każdego z obszarów jest wyznaczana wartość średnia ze wszystkich pikseli należących do danego obszaru. Zatem p_1 oznacza wartość średnią dla wewnętrznego obszaru, zaś p_2 oznacza wartość średnią obliczoną ze wszystkich pikseli zewnętrznego obszaru. Następnie wyznacza się współczynnik p :

$$p = p_1 / p_2 \quad (6.6)$$

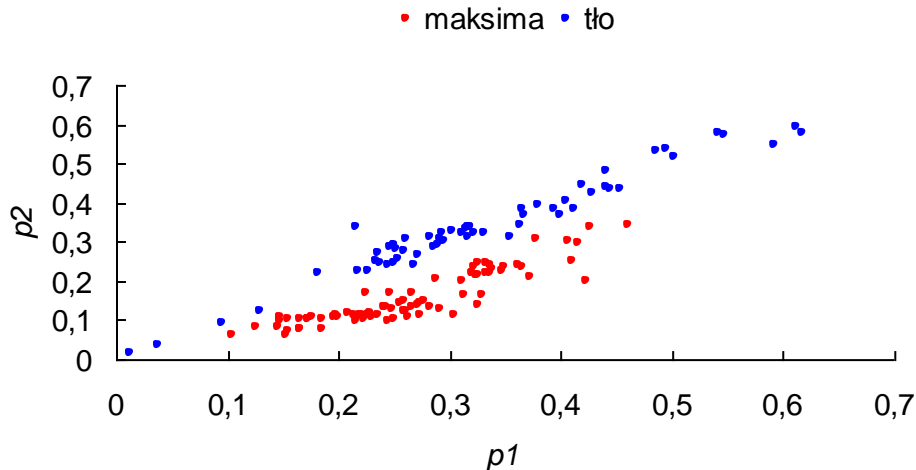
Gdy maska pokrywa się z lokalnym maksimum, wtedy wartość p powinna znacząco przekraczać 1. W pozostałych przypadkach, wartość p powinna być zbliżona do 1. Dokładna wartość progu oddzielającego te dwa przypadki została wyznaczona empirycznie. Przyjęte kryterium zakłada maksymalizację liczby znalezionych maksimum odpowiadających istniejącym liniom („prawdziwe”), przy jednoczesnym nie przekraczaniu zadanego progu liczby detekcji nieodpowiadających istniejącym liniom („fałszywe”). Wartość progową dla ilości fałszywych detekcji przyjęto podobnie jak poprzednio – na poziomie 30%. Zbadano ilość poprawnie klasyfikowanych maksimum oraz ilość poprawnie klasyfikowanych pozostałych przypadków (tło). Otrzymane wyniki zostały przedstawione na rysunku (6.7).



Rys. 6.7. Procent poprawnych decyzji dla metody LCA

6.10.3. Klasyfikacja cech (FC)

Metoda *klasyfikacji cech* wykorzystuje tę samą maskę, co metoda LCA. Różnica polega na tym, że w tym przypadku nie jest liczony parametr p , lecz jest analizowany rozkład parametrów p_1 oraz p_2 . Na rysunku (6.8) przedstawiono otrzymany rozkład parametrów na płaszczyźnie.



Rys. 6.8. Rozkład parametrów p_1 oraz p_2

Możliwe są dwa przypadki: maska pokrywa się z lokalnym maksimum lub nie. W celu ich odróżnienia, zastosowano następujące metody klasyfikacji, opisane w rozdziale 3.: k -średnich (KM), rozmytych k -średnich (FKM), algorytm Gustafsona-Kessela (GK), k -najbliższych sąsiadów (KMean), rozmytych k -najbliższych sąsiadów (FKMean), MMD, *Support Vector Machine* (SVM) oraz sieci neuronowe (NN).

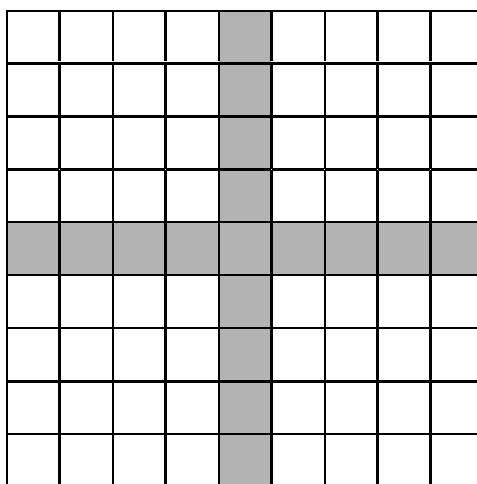
Niektóre z powyższych metod zależą od pewnych parametrów, jak np. parametr określający stopień rozmycia poszczególnych klastrów. Parametry te zostały zoptymalizowane w celu uzyskania najlepszych wyników dla danej metody. W przypadku sieci neuronowej, zastosowano standardową sieć składającą się z dwóch warstw. Warstwa wejściowa składała się z dwóch nieliniowych neuronów, zaś warstwa wyjściowa składała się z jednego liniowego neuronu. Uczenie sieci przeprowadzono na zbiorze uczącym składającym się z 600 punktów przestrzeni (p_1, p_2) odpowiadających maksimum oraz 600 punktów odpowiadających tłu. Proces uczenia powtórzono 50-ciokrotnie. Sieć przetestowano na odrębnym zbiorze testującym złożonym, podobnie jak zbiór uczący, z 1200 punktów.

6.10.4. Analiza profilu (PPA)

Kolejna metoda polega na analizie profili sygnałów jednowymiarowych. W celu wyodrębnienia jednowymiarowych profili w przestrzeni współczynników transformacji Hougha, zastosowano maskę przedstawioną na rysunku (6.9).

Za pomocą tej maski dwa jednowymiarowe sygnały są wydobywane dla każdego kolejnego położenia maski względem współczynników transformacji Hougha. Pierwszy z nich odpowiada pionowemu szaremu paskowi na masce, a drugi – poziomemu. Kształt wydobytych profili zależy od tego, czy centralny punkt maski pokrywa się w danej chwili z maksimum, czy nie. Każdy z dwóch profili ma długość 9 próbek. Następnie sygnał odpowiadający pionowemu paskowi jest dołączany na koniec sygnału odpowiadającego

poziomemu paskowi. W ten sposób powstaje jeden sygnał składający się z 18 próbek. Sygnał ten jest poddawany analizie za pomocą sieci neuronowej. Sieć neuronowa jest dobrym narzędziem do klasyfikacji sygnałów, gdyż na ogół daje lepsze wyniki niż metody oparte na numerycznej analizie rzeczywistych i teoretycznych kształtów maksimum. Sieć neuronowa może nauczyć się złożonych relacji pomiędzy próbkami sygnału wejściowego, a przez to, za pomocą ćwiczenia, może być dostosowana do różnych kształtów maksimum.



Rys. 6.9. Maska do analizy profili

W tym przypadku zastosowano sieć składającą się z trzech warstw. Warstwa wejściowa składa się z 18 neuronów z nieliniową funkcją przejścia typu log-sigmoid. Warstwa ukryta składa się z nieliniowych neuronów (funkcja przejścia jest również log-sigmoidalna). Warstwa wyjściowa składa się z 2 liniowych neuronów. Jeden z nich jest skojarzony z sygnałem odpowiadającym obecności maksimum, a drugi z sygnałem odpowiadającym pozostałym przypadkom. Ilość neuronów w warstwie ukrytej została dobrana eksperymentalnie. W tym celu zbadano ilość poprawnych i niepoprawnych klasyfikacji w zależności od ilości neuronów w warstwie ukrytej. Do badania użyto 1200 wzorców tła oraz 1200 wzorców maksimum. Oba te zbiory podzielono na: składający się z 600 wzorców tła i 600 wzorców maksimum zbiór uczący oraz zbiór testowy składający się z pozostałych wzorców. Przykłady wzorców użytych do uczenia sieci zostały zaprezentowane w tabeli (6.4), zaś przykładowe profile zostały zaprezentowane w tabeli (6.5). Dla każdej liczby neuronów w warstwie ukrytej badanie zostało przeprowadzone 50 razy. Uśrednione wyniki zaprezentowano w tabeli (6.3).

Tabela 6.3. Uśrednione wyniki badania sieci dla metody PPA

ilość neuronów w warstwie ukrytej	poprawne klasyfikacje	
	wzorce maksimum	Wzorce tła
9	99.3%	98.8%
8	99.3%	98.8%
7	99.3%	98.8%
6	99.1%	98.6%
5	98.8%	98.1%
4	98%	97.2%

Tabela 6.4. Przykłady wzorców użytych do uczenia sieci neuronowej

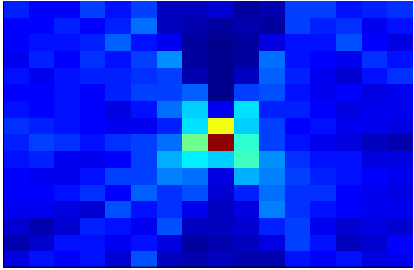
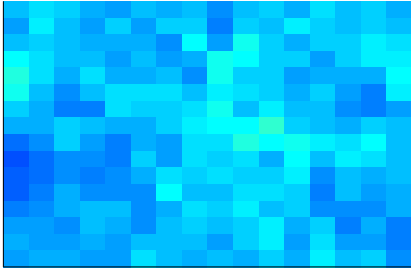
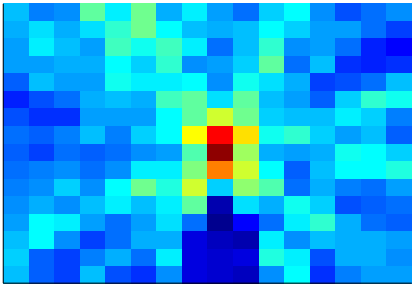
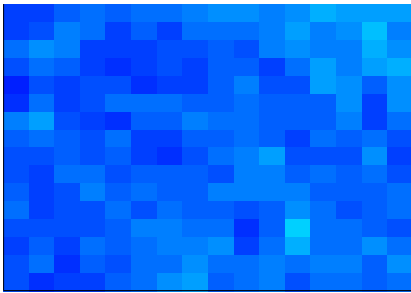
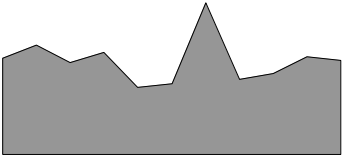
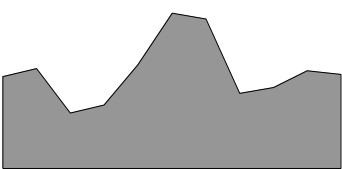
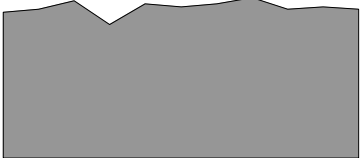
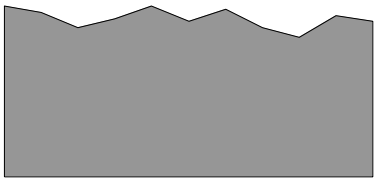
Wzorce maksimów	Wzorce tła
	
	

Tabela 6.5. Przykładowe profile

Przykłady profili odpowiadających maksimom	
	
Przykłady profili odpowiadających tłu	
	

Jak widać, najbardziej optymalna ilość neuronów w warstwie ukrytej wynosi w tym przypadku 7. Zwiększanie liczby neuronów ponad tę wartość nie przyniosło już poprawy uzyskiwanych wyników, zaś zmniejszanie powodowało uzyskiwanie coraz większej liczby błędnych klasyfikacji.

6.11. Weryfikacja

W celu zminimalizowania prawdopodobieństwa wskazywania przez lokalne maksimum transformacji Hougha na istnienie linii, której w rzeczywistości nie ma, zastosowano dodatkową weryfikację hipotez metodą *trzech linii*, zaproponowaną w rozdziale 4.4.2.

6.12. Poszukiwanie brakujących linii

Ostatecznym celem pracy algorytmu jest detekcja par, a nie pojedynczych linii. Parę stanowią dwie równoległe linie oddalone od siebie o pewną odległość, mieszczącą się w zadanych granicach. Dlatego ostatnim etapem pracy algorytmu jest poszukiwanie ewentualnej brakującej linii dla każdej pojedynczej. W tym celu, obraz źródłowy jest modyfikowany za pomocą przestrajanego filtra kierunkowego (rozdz. 2.6.), którego orientacja jest zgodna z orientacją znalezionej pojedynczej linii. Dzięki temu uzyskuje się maksymalne wzmocnienie linii o danej orientacji. Przefiltrowany obraz jest powtórnie analizowany za pomocą transformacji Hougha, jednak w tym przypadku poszukuje się już tylko jednego maksimum.

6.13. Wyniki

Skuteczność poszczególnych metod detekcji maksimów została sprawdzona eksperymentalnie. Test przeprowadzono na zbiorze składającym się z 500 przykładowych wzorców zawierających maksima oraz 500 przykładowych wzorców nie zawierających maksimów. Metody wykorzystujące sieci neuronowe oraz zaawansowane klasyfikatory zostały wcześniej poddane procesowi uczenia na oddzielnym zbiorze uczącym. Otrzymane wyniki zostały zaprezentowane w tabeli (6.6).

Tabela 6.6. Porównanie metod klasyfikacji (detekcji) maksimów

Metoda	Liczba poprawnie sklasyfikowanych maksimów linii	Liczba poprawnie sklasyfikowanych wzorców tła	Czas wykonania [ms]
ST (próg = 0,45)	74.2%	72.2%	<1
LAT (próg = 0,45)	74,2%	77.3%	<1
LCA (próg = 0,95)	98.2%	83.2%	25
FC-KM	84.7%	31.3%	980
FC-FKM	84.5%	31.7%	1050
FC-GK	95.2%	99.2%	1200
FC-KMean	91.1%	92.3%	870
FC-FKMean	97.4%	97.8%	1080
MMD	91.2%	92.0%	880
FC-SVM	98.4%	99.4%	1815
FC-NN	98.8%	99.0%	30
PPA-NN	99.1%	98.6%	110

Jak można zauważyć, najlepsze wyniki poprawnej klasyfikacji maksimów linii osiągnięto dla metody analizy profilu (PPA), nieznacznie gorsza jest metoda klasyfikacji cech FC-NN. Dlatego metoda PPA-NN została ostatecznie wykorzystana w całym algorytmie detekcji linii Kikuchiego. Cały algorytm został zaimplementowany w języku MATLAB oraz przetestowany pod kątem efektywności oraz wiarygodności detekcji linii Kikuchiego. Test przeprowadzono na zbiorze 30-tu obrazów zawierających 312 par linii Kikuchiego. Otrzymane wyniki zostały zaprezentowane w tabeli (6.7). Algorytm standardowy oparty jest jedynie na transformacji Hougha bez zaawansowanych technik przetwarzania wstępnego obrazu oraz wyrafinowanej detekcji maksimów.

Otrzymane wyniki potwierdzają, iż zastosowanie zaawansowanych metod wstępnego przetwarzania obrazu źródłowego, jak i skutecznych metod detekcji maksimów w przestrzeni transformacji Hougha, pozwala na uzyskanie znaczącej poprawy skuteczności detekcji linii Kikuchiego.

Tabela 6.7. Skuteczność opracowanej metody detekcji linii Kikuchiego

Algorytm	Poprawnie znalezione pary linii	Niepoprawnie znalezione pary linii
Standardowy IMIM-PAN	224 (71.8%)	36 (11.5%)
Opracowany z metodą PPA-NN	288 (92.3%)	2 (0.6%)

Tabela 6.8. Szersze porównanie opracowanej metody detekcji z innymi metodami

Algorytm	Poprawnie znalezione pary linii	Niepoprawnie znalezione pary linii	Czas wykonania [ms]
1. Standardowa transf. Hougha	212 (67.9%)	28 (9.0%)	400
2. Używany w IMIM-PAN	224 (71.8%)	36 (11.5%)	1100
3. Zaproponowany w rozdz. 5.	261 (83.7%)	2 (0.6%)	1450
4. Zaproponowany w rozdz. 4.	275 (88.1%)	2 (0.6%)	11200
5. Zaproponowany LAT	287 (91.9%)	2 (0.6%)	3900
6. Zaproponowany PPA-NN	288 (92.3%)	2 (0.6%)	2700
7. Zaproponowany FC-NN	288 (92.3%)	2 (0.6%)	2600

Szerszego porównania opracowanych metod detekcji linii z alternatywnymi podejściami dokonano w tabeli (6.8). Z tabeli (6.7) wynika, że sama metoda LAT klasyfikacji maksimów nie jest dokładna (74.2%), ale jest za to bardzo szybka. Jednak po zintegrowaniu jej z czasochłonnymi, ale za to dokładnymi metodami dodatkowej weryfikacji wyników, uzyskuje się łączny algorytm, który jest bardzo dokładny (91.9%) w detekcji linii, ale już nie tak szybki. Z kolei metoda FC-NN jest bardzo skuteczna (98.8%) w klasyfikacji maksimów, chociaż 30 razy wolniejsza. Dzięki temu nie wymaga ona bardzo częstego stosowania procedury weryfikacji i poszukiwania możliwych brakujących par, przez co końcowy algorytm detekcji linii jest szybszy niż dla metody LAT. Z kolei metoda PPA-NN pozwala na uzyskanie podobnej skuteczności detekcji, jak metoda FC-NN, przy nieco dłuższym czasie działania.

Zbadano również jakość pracy algorytmu dla różnych odmian transformacji Hougha. Przetestowano działanie następujących transformacji: *Combinatorial Hough Transform* (CHT), *Probabilistic Hough Transform* (PHT), *Adaptive Hough Transform* (AHT), *Randomized Hough Transform* (RHT) oraz *Gray-scale Hough Transform* (GHT). Jako metodę detekcji maksimów zastosowano metodę FC-NN. Badanie przeprowadzono z pominięciem etapu weryfikacji oraz poszukiwania brakujących linii w celu lepszej oceny wpływu poszczególnych odmian transformacji Hougha na skuteczność oraz wiarygodność detekcji linii Kikuchiego. Zmierzono również czas wykonania poszczególnych transformacji, przyjmując za punkt odniesienia czas wykonania dla standardowej transformacji Hougha. Badanie przeprowadzono na 30 obrazach testowych. Uzyskane wyniki zaprezentowano w tabeli (6.9).

Tabela 6.9. Porównanie różnych odmian transformacji Hougha

Transformacja Hougha	Poprawnie znalezione pary linii	Niepoprawnie znalezione pary linii	Czas obliczeń
SHT	283	12	100 %
CHT	281	20	58 %
PHT	265	13	72 %
AHT	264	8	75 %
RHT	270	10	20 %
GHT	282	11	210 %

6.14. Optymalizacja metody – dobór opcji i wartości parametrów

6.14.1. Czas wykonania poszczególnych etapów algorytmu

Przeprowadzono pomiar czasu wykonania poszczególnych etapów algorytmu. Uzyskane wyniki zaprezentowano w tabeli (6.10). Mają one charakter orientacyjny, gdyż czas wykonania niektórych etapów (weryfikacja, poszukiwanie możliwych brakujących par) nie jest stały, ale zależy od różnych czynników, np. od liczby znalezionych pojedynczych linii, czy też liczby znalezionych par linii. Jak widać, najbardziej czasochłonne jest (w kolejności): zwiększenie kontrastu (40%), poszukiwanie brakujących linii do pary (22%), odsumianie metodą *Neighshrink* (15%), weryfikacja (10%), ekstrakcja maksimów (7%) oraz transformacja Hougha-Radona (4%).

6.14.2. Ocena wpływu poszczególnych etapów na wyniki końcowe

Zbadano także, jaki wpływ mają poszczególne moduły algorytmu na uzyskiwane wyniki końcowe. Badania przeprowadzono na 30-tu obrazach testowych zawierających w sumie 312 par linii Kikuchiego.

Tabela 6.10. Pomiar czasu wykonania poszczególnych etapów algorytmu

Etap algorytmu		Względny czas wykonania	
Detekcja i usunięcie artefaktów		<1%	
Estymacja poziomu szumu		1%	
Odszumianie za pomocą metody <i>Neighshrink</i> [Chen04]		15%	
Zwiększenie kontrastu	Modyfikacja krzywej tonalnej	<1%	40%
	Transformacja <i>curvelet</i>	40%	
Filtracja kierunkowa		1%	
Segmentacja (binaryzacja)		<1%	
Przetwarzanie obrazu binarnego		<1%	
Transformacja Hougha-Radona		4%	
Ekstrakcja maksimów		7%	
Weryfikacja		10%	
Poszukiwanie brakujących linii do pary		22%	

Tabela 6.11. Porównanie wpływu poszczególnych etapów algorytmu na końcowe wyniki

Algorytm	Poprawnie znalezione pary linii	Niepoprawnie znalezione pary linii
Pełny algorytm	288 (92%)	2
Algorytm bez odsumiania	288 (92%)	4
Algorytm bez zwiększania kontrastu	249 (79%)	2
Algorytm bez weryfikacji	288 (92%)	12
Algorytm bez poszukiwania brakujących par	283 (91%)	2

Z przeprowadzonych badań wynika, iż kluczowym elementem algorytmu mającym wpływ na jego skuteczność jest zwiększenie kontrastu obrazu. Pomimo tego, że jest on bardzo czasochłonny (40%), to jego zastosowanie pozwala na wyraźne zwiększenie liczby znalezionych linii (z 249 do 288). Jest to zrozumiałe, gdyż większy kontrast pozwala na detekcję słabiej widocznych linii Kikuchiego.

Kolejnym ważnym modułem z punktu widzenia jakości uzyskiwanych wyników jest etap weryfikacji. To dzięki niemu możliwe jest uzyskanie niskiego poziomu błędnych wyników pracy całego algorytmu.

Jak widać, odsumianie obrazu nie wnosi istotnych zmian jeśli chodzi o skuteczność detekcji, chociaż jego brak nieznacznie zwiększa prawdopodobieństwo uzyskania „liniuduchów”. Wiąże się to z tym, że nadmierny poziom szumu na obrazie powoduje powstawanie w przestrzeni transformacji Hougha maksimum nie odpowiadających rzeczywistym liniom na obrazie. Procedura poszukiwania brakujących par linii wnosi niewielki wkład do całkowitej skuteczności metody, aczkolwiek w niektórych przypadkach pozwala na odnalezienie brakującej linii do pary.

6.15. Podsumowanie

Rozdział ten jest najważniejszym rozdziałem rozprawy pod względem jakości uzyskanych wyników. Zaproponowany w nim algorytm (algorytmy) w wyraźny sposób przewyższa rozwiązania, które są obecnie stosowane w krajowych i światowych laboratoriach naukowych zajmujących się detekcją linii Kikuchiego.

7. Wnioski końcowe

W pracy zaproponowano nowe, o wiele lepsze niż obecnie stosowane metody detekcji par dyfrakcyjnych linii Kikuchiego, które występują na obrazach mikroskopowych struktur polikrystalicznych. Do prawidłowego wnioskowania o orientacjach krystalograficznych wymaga się znalezienia możliwie jak największej liczby par linii na pojedynczym obrazie mikroskopowym. Metoda referencyjna z Instytutu Metalurgii i Inżynierii Materiałowej PAN w Krakowie oferuje średnio 7.5 takich par, a metoda opracowana – aż 9.6 par, dodatkowo z o wiele mniejszym prawdopodobieństwem stwierdzenia obecności par linii, które w rzeczywistości nie istnieją. Daje to podstawy do o wiele dokładniejszych badań materiałów polikrystalicznych oraz panowania nad procesem ich produkcji.

W wyniku przeprowadzonych, bardzo rozległych prac eksperymentalnych opracowano bardzo szybki i dokładny algorytm do detekcji par linii Kikuchiego, opisany w rozdziale 6. (patrz FC-NN w tabeli 6.8), który podnosi skuteczność detekcji z 71.8 % do 92.3 %, redukując przy tym znacznie niebezpieczeństwo detekcji nieistniejących par linii z 11.5% do 0,6%. Algorytm ten wykorzystuje zaawansowane, nowoczesne, jak widać skuteczne, metody analizy i przetwarzania obrazów. Co istotne, jest tylko około 2.5 razy wolniejszy niż obecnie stosowany, co nie jest jego dużą wadą uwzględniając bardzo dużą skuteczność detekcji, którą oferuje.

Biorąc powyższe pod uwagę można stwierdzić, że cele pracy zostały osiągnięte, a jej teza wykazana.

W porównaniu z dotychczasowymi zastosowaniami transformacji Hougha, przedstawione w niniejszej pracy rozwiązania charakteryzują się zastosowaniem znacznie bardziej złożonych i zaawansowanych metod wstępnego przetwarzania obrazu źródłowego oraz autorskich metod przetwarzania wyniku działania transformacji Hougha. Zastosowane w rozprawie przetwarzanie wstępne obejmuje: korekcję tła obrazu, detekcję artefaktów, filtrację filtrami kierunkowymi, selektywne odszumianie falkowe, zwiększenie kontrastu za pomocą transformacji *curvelet*, binaryzację w oparciu o właściwości statystyczne obrazu oraz operacje morfologiczne na obrazie binarnym. Z kolei przetwarzanie wyniku transformacji Hougha obejmuje zarówno niestosowane gdzie indziej metody ekstrakcji maksimum w przestrzeni transformacji, jak również nowe metody weryfikacji uzyskanych wyników, np. szybki klasyfikator neuronowy.

Zdaniem autora do oryginalnych osiągnięć badań i rozprawy można zaliczyć:

- 1) opracowanie powolnego ale za to bardzo dokładnego, algorytmu detekcji linii techniką maski składającej się z trzech linii, zaproponowanego w rozdziale 4. (podrozdział 4.4.2); algorytm ten może być stosowany jako dodatkowe, końcowe narzędzie weryfikacyjne w innych, szybszych, ale za to mniej dokładnych algorytmach detekcji (w pracy, algorytmy 2. i 3.);

- 2) zaproponowanie modyfikacji (tabela 5.4) klasycznej, zliczeniowej realizacji transformacji Hougha (tabela 2.1), sprowadzającej się do innego sposobu inkrementowania akumulatorów; dzięki temu zmniejszono poziom szumów w przestrzeni transformacji Hougha, co przekłada się na mniejszą liczbę błędnych detekcji linii.
- 3) przebadanie celowości użycia różnych metod analizy i przetwarzania obrazów do zadania detekcji par linii Kikuchiego (np. tabele 6.9 i 6.10), w tym także ostatnio zaproponowanych (np. uzdatnianie obrazów metodą przestrajanych filtrów kierunkowych, poprawa kontrastu metodą transformacji *curvelet*, odsumianie metodą transformacji falkowej *Neighshrink*);
- 4) opracowanie trzech nowych algorytmów przeznaczonych do detekcji par dyfrakcyjnych linii Kikuchiego na obrazach mikroskopowych, opisanych w rozdziałach 4., 5. i 6., charakteryzujących się różnymi własnościami użytkowymi i porównanych w tabeli (6.8);
- 5) opracowanie rozwiązania (algorytm 3., tabela 6.7 oraz FC-NN w tabeli 6.8), które pod względem skuteczności detekcji zdecydowanie przewyższa metodę stosowaną aktualnie w specjalistycznych laboratoriach badawczych oraz oferuje zupełnie nowy poziom jakości oceny struktur polikrystalicznych.
- 6) opracowanie metod detekcji maksimumów w przestrzeni transformacji Hougha (rozdział 6.10), a zwłaszcza metod: LAT, LCA, FC oraz PPA.

Algorytm nr 2 z rozdziału 5., opracowany w ramach niniejszej pracy, już został zaimplementowany w języku C. Obecnie trwają prace nad podobną implementacją zwycięskiego algorytmu nr 3, zaproponowanego w rozdziale 6. Autor ma nadzieję, że w niedalekiej przyszłości stworzone przez niego metody i programy będą rutynowo stosowane przez specjalistów do badań materiałów polikrystalicznych.

Wyniki przedstawione w rozprawie były wcześniej opublikowane w pracach [Frac04a, Frac04b, Frac05a, Frac05b, Frac06a, Frac06b, Frac06c].

8. Literatura

- [Alam54] M. N. Alam, M. Blackman, D. W. Pashley, *High-angle Kikuchi Patterns*, Proceedings of Royal Society, vol. 222 pp. 224-242, 1954.
- [Archi04] W. Archibald, *Microstructural Characterization of Aluminum Thin Films and Foils: Grain Boundary Topology, Properties and Statistics*, Phd Thesis, Carnegie Mellon University, 2004.
- [Asto89] J. Astola, P. Haavisto, Y. Neuvo, *Detail Preserving Monochrome and Color Image Enhancement Algorithms*, J. C. Simon (Ed.), Elsevier, Amsterdam, 1989.
- [Babus02] R. Babuska, U. Kaymak, P. J. van der Veen, *Improved Covariance Estimation for GustafsonKessel Clustering*, In Proceedings of 2002 IEEE International Conference on Fuzzy Systems, pp. 1081-1085, Honolulu, Hawaii, May 2002.
- [Bakh99] A. Bakhtazad, A. Palazoglu, J. A. Romagnoli, *Process Data De-noising Using Wavelet Transform*, Intelligent Data Analysis, vol. 3, p. 267-285, 1999.
- [Ball81] D. H. Ballard, *Generalizing the Hough Transform to Detect Arbitrary Shapes*, Pattern Recognition, vol. 13, no. 2, pp. 111-122, 1981.
- [Basc02] J. Bascou, A. Tommasi, D. Mainprice, *Plastic Deformation and Development of Clinopyroxene Lattice Preferred Orientations in Eclogites*, Journal of Structural Geology, 24, pp. 1357-1368, 2002.
- [Bent90] D. Ben-Tzvi, M. B. Sandler, *A Combinatorial Hough Transform* Pattern Recognition Letters, vol. 11, no. 3, pp. 167-174, 1990.
- [Berg91] J. R. Bergen, H. Shvaytser, *A Probabilistic Algorithm for Computing Hough Transforms*, Journal of Algorithms, vol. 12., no. 4, pp. 639-656, 1991.
- [Bez81] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, 1981.
- [Blas1] B. Balasko, J. Abonyi, B. Feil, *Fuzzy Clustering and Data Analysis Toolbox*, <http://www.fmt.vein.hu/softcomp/>
- [Brew02] L. N. Brewer, *Misorientation Mapping for Visualization of Plastic Strain via Electron Back-Scattered Diffraction*, Microscopy & Microanalysis, 2002.
- [Brow84] D. R. K. Brownrigg, *The Weighted Median Filter*, Communications of the ACM, vol. 27, Issue 8, pp. 807-818, 1984.
- [Bung00] H. J. Bunge, *Industrial Applications of X-ray diffraction*, New York-Basel, pp. 919, 2000.
- [Burg98] C. J. C. Burges, *A Tutorial on Support Vector Machines for Pattern Recognition*, Data Mining and Knowledge Discovery, vol. 2, pp. 121-167, 1998.
- [Burr98] C. S. Burrus, R. A. Gopinath, H. Guo, *Introduction to wavelets and wavelet transforms*, Saddle pper River, NJ (USA): Prentice Hall, 1998.
- [Cabu04] C. Cabus, H. Regle, B. Bacroix, *Phases Transformation Textures in Steels*, International Conference on Advanced High Strength Sheet Steels for Automotive Applications, Winter Park, Colorado, USA, 2004.

- [Cand98] E. J. Candès, *Ridgelets: Theory and Applications*, Ph.D. thesis, Department of Statistics, Stanford University, 1998.
- [Chan00a] S. G. Chang, M. Vetterli, B. Yu, *Adaptive Wavelet Thresholding for Image Denoising and Compression*, IEEE Transactions on Image Processing, vol. 9(9), pp. 1532-1546, Sep. 2000.
- [Chan00b] T. Chan, A. Marquina, P. Mulet, *High-order Total Variation-based Image Restoration*, SIAM Journal on Scientific Computing, vol. 22, no. 2, pp. 503-516, 2000.
- [Chen04] G. Y. Chen, T. D. Bui, A. Krzyzak, *Image Denoising Using Neighbouring Wavelet Coefficients*, IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 2, pp. 917-920, 2004.
- [Chen94] W. Chen, M. C. Chaturvedi, *The Influence of Grain Boundary Properties on Creep Fracture*, Superalloys, vol. 718, 1994.
- [Choi98] H. Choi, R. G. Baraniuk, *Analysis of Wavelet Domain Wiener Filters*, IEEE International Symposium on Time-Frequency and Time-Scale Analysis, (Pittsburgh), Oct. 1998.
- [Cris00] N. Cristianini, J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, Cambridge University Press, ISBN 0-521-78019-5, 2000.
- [Dark03] C. Dark, S. Speller, H. Wu, A. J. Wilkinson and C. R. M. Grovenor, *Grain Boundary Properties of Ti-2212 Thin Films*, proceedings of European Conference on Applied Superconductivity, Sorrento 2003.
- [Daub92] I. Daubechies, *Ten Lectures on Wavelets*, CBMS-NSF Regional Conference Series in Applied Mathematics, Society for Industrial & Applied Math, 1992.
- [Dixo06] S. Dixon, S. Essex, *Texture Measurement of Aluminum Sheet Using Electron Backscattered Diffraction and Electromagnetic Acoustic Transducers*, The 45th Annual British Conference on NDT, Stratford, 2006.
- [Do03] M. N. Do, M. Vetterli, *The Finite Ridgelet Transform for Image Representation*, IEEE Transactions on Image Processing, vol. 12, pp. 16-28, Jan. 2003.
- [Dono94] D. L. Donoho, I. M. Johnstone, *Ideal Spatial Adaptation by Wavelet Shrinkage*, Biometrika, vol. 81(3), pp. 425-455, Aug. 1994.
- [Dono95a] D. L. Donoho, *De-noising by Soft Thresholding*, IEEE Transactions on Information Theory, vol. 41(3), pp. 613-627, May 1995.
- [Dono95b] D. L. Donoho, I. M. Johnstone, *Adapting to Smoothness via Wavelet Shrinkage*, Journal of the Statistical Association, vol. 90(432), pp. 1200-1224, Dec. 1995.
- [Drie05] I. V. Driessche, B. Schoofs, G. Penneman, E. Bruneel, S. Hoste, *Review of the Application of High Temperature Superconductors in Coated Conductor Development and the Measurement of Their Properties*, Measurement science review, vol. 5, Section 3, 2005.
- [Driv96] J. H. Driver, M. C. Theyssier, C. Maurice, *Electron Backscattered Diffraction Microtexture Studies on Hot Deformed Aluminium Crystals*, Material Science Technology, vol. 12, pp. 851-858, 1996.
- [Duda72] R. O. Duda, P. E. Hart, *Use of the Hough Transformation to Detect Lines and Curves in Pictures*, Communications ACM, vol. 15, pp. 11-15, 1972.
- [Ecab04] O. Ecabert, J. P. Thiran, *Adaptive Hough Transform for the Detection of Natural Shapes under Weak Affine Transformations*, Pattern Recognition Letters, vol. 25, pp. 1411-1419, 2004.
- [Fitt98] N. C. Fitton, S. J. D. Cox, *Optimizing the Application of the Hough Transform for Automatic Feature Extraction from Geoscientific Images*, Computers and Geosciences, vol. 24, no. 10, pp. 933-951, 1998.

- [Floer02] W. Floer, Y. M. Hu, U. Krupp, H. J. Christ, *Application of the EBSD Technique to Study the Initiation and Propagation of Short Cracks*, Practical Metallography, vol. 7, 2002.
- [Frac04a] R. Frączek, T. Zieliński, *Praktyczne aspekty nowego algorytmu detekcji linii Kikuchiego*, Materiały XIV Sympozjum „Modelowanie i Symulacja Systemów Pomiarowych, Krynica Górská 2004. <http://best-programs.net>
- [Frac04b] R. Frączek, T. Zieliński, *Algorytm detekcji linii Kikuchiego w mikroskopii elektronowej*, Kongres Metrologii, Wrocław 2004.
- [Frac05a] R. Frączek, T. Zieliński, *Algorytm detekcji linii Kikuchiego z użyciem zmodyfikowanej transformacji Hougha*, XV Sympozjum „Modelowanie i Symulacja Systemów Pomiarowych”, str. 209-218, Krynica Górská 2005.
- [Frac05b] R. Frączek, T. Zieliński, *New Algorithm for Kikuchi Lines Detection in Electron Microscopy Images*, European Signal Processing Conference EUSIPCO-2005, Antalya, 2005.
- [Frac06a] R. Frączek, *Ulepszony algorytm detekcji linii Kikuchiego*, XVI Sympozjum „Modelowanie i Symulacja Systemów Pomiarowych”, Krynica 2006.
- [Frac06b] R. Frączek, T. Zieliński, *Peak Detection Methods in Hough-Transform Based Kikuchi Bands Extraction*, International Conference of Signals and Electronic Systems ICSES-206, Łódź 2006.
- [Frac06c] R. Frączek, T. Zieliński, *Application of Advanced Image Processing Techniques to Automatic Kikuchi Lines Detection*, European Signal Processing Conference EUSIPCO-2006, Florencja 2006.
- [Free91] W. T. Freeman, E. H. Adelson, *The Design and Use of Steerable Filters*, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 13, no. 9, pp. 891-906, 1991.
- [Fund03] J. J. Fundenberger, A. Morawiec, E. Bouzy, J. S. Lecomte, *Polycrystal Orientation Maps from TEM*, Ultramicroscopy, vol. 96, pp. 127-137, 2003.
- [Gala99] C. Galambos, J. Matas, J. Kittler, *Progressive Probabilistic Hough Transform for Line Detection*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, p.1554, 1999.
- [Gey02] N. Gey, E. Gautier, M. Humbert, A. Cerqueira, J.L. Bechade, P. Archambault, *Study of the α/α' Phase Transformation of Zy-4 in Presence of Applied Stresses at Heating: Analysis of the Inherited Microstructures and Textures*, Journal of Nuclear Materials, vol. 302, pp.175-184, 2002.
- [Gilb04] G. Gilboa, N. Sochen, Y. Y. Zeevi, *Image Enhancement and Denoising by Complex Diffusion Processes*, IEEE Trans. On Pattern Analysis and Machine Intelligence, vol. 26, no. 8, Aug. 2004.
- [Gonz93] R. C. Gonzalez, R. E. Woods, *Digital Image Processing*, Prentice Hall, 1993.
- [Gopi97] D. Gopikrishna, S. N. Jha, L. N. Dash, *Influence of Microstructure on Fatigue Properties of Alloy 718*, The Fourth International Special Emphasis Symposium on Superalloy, Pittsburgh, Pennsylvania, 1997.
- [Groe81] T. M. van Veen, F. C. A. Groen, *Discretization Errors in the Hough Transform*, Pattern Recognition, vol. 14, no. 1, 1981.
- [Gunn98] S. Gunn, *Support Vector Machines for Classification and Regression*, Technical Report ISIS-1-98, Department of Electronics and Computer Science, University of Southampton, 1998.
- [Hans97] K. Hansen, J. D. Andersen, *Understanding the Hough Transform: Hough Cell Support and its Utilization*, Image and vision computing, vol. 15, no. 3, pp. 205-218, 1997.

- [Hanz99] A. B. Hamza, P. L. Luque-Escamilla, J. M. Aroza, R. R. Roldan, *Removing Noise and Preserving Details with Relaxed Median Filters*, Journal of Mathematical Imaging and Vision, vol. 11, pp. 161-177, 1999.
- [Heid00] F. Heidelbach, K. Kunze, H. R. Wenk, *Texture Analysis of a Recrystallized Quartzite Using Electron Diffraction in the Scanning Electron Microscope*, Journal Of Structural Geology, vol. 22, pp. 91-104, 2000.
- [Houg62] P. V. C. Hough, *Method and Means for Recognizing Complex Patterns* U. S. Patent 3,069,654, 1962.
- [Hump01] F. J. Humphreys, *Grain and Subgrain Characterisation by Electron Backscatter Diffraction*, Journal of Material Science, vol. 36, pp. 3833-3854, 2001.
- [Hump99] F. J. Humphreys, *Quantitative Metallography by Electron Backscattered Diffraction*, Journal of Microscopy, vol. 195, pp.170-185, 1999.
- [Hunt02] A. Hunter, M. Ferry, *Comparative Study of Texture Development in Strip-cast Ferritic and Austenitic Stainless Steels*, Scripta Materialia, Sep. 2002.
- [Hurl03] P. J. Hurley, F. J. Humphreys, *The Application of EBSD to the Study of Substructural Development in a Cold Rolled Single-phase Aluminium Alloy*, Acta Materialia, Feb. 2003.
- [Illi87] J. Illingworth, J. Kittler, *The Adaptive Hough Transform*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 9, no. 5, pp. 690-698, 1987.
- [Illi88] J. Illingworth, J. Kittler, *A Survey of the Hough Transform*, CVGIP, vol. 44, pp. 87-116, 1988.
- [Inte1] *Internetowy podręcznik statystyki*, <http://www.statsoft.pl/textbook/stathome.html>
- [Jaco04] M. Jacob, M. Unser, *Design of Steerable Filters for Feature Detection Using Canny-Like Criteria*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 8, Aug. 2004.
- [Jain89] A. K. Jain, *Fundamentals of Digital Image Processing*, Prentice-Hall International, ISBN 0-13-332578-4, 1989.
- [Jans01] M. Jansen, *Noise Reduction by Wavelet Thresholding*, vol. 161. Springer Verlag, United States of America, 1st edition, 2001.
- [Jans99] M. Jansen, G. Uytterhoeven, A. Bultheel, *De-noising by Integer Wavelet Transforms and Generalized Cross Validation*, Medical Physics, vol. 26, pp. 622-630, 1999.
- [Joun1] P. H. Joneau, *Introduction to EBSD*, <http://cime.epfl.ch>
- [Käl94] H. Kälviäinen, P. Hirvonen, E. Oja, L. Xu, *Comparisons of Probabilistic and Non-probabilistic Hough Transforms*, Proceedings of the 3rd European Conference on Computer Vision, pp. 351-360, 1994.
- [Kesi00] A. L. Kesidis, N. Papamarkos, *On the Gray-scale Inverse Hough transform*, Image and Vision Computing, vol. 18, pp. 607-618, 2000.
- [Kimm75] C. Kimme, D. H. Ballard, J. Sklansky, *Finding Circles by an Array of Accumulators*, Communications of the Association for Computing Machinery, vol. 18, pp. 120-122, 1975.
- [Kiry91] N. Kiryati, Y. Eldar, A. M. Bruckstein, *A Probabilistic Hough Transform*, Pattern Recognition, vol. 24, no. 4., pp. 303-316, 1991.
- [Koon76] L. G. Koontz, W. P. M. Narendra, K. A. Fukunaga, *Graph-Theoretic Approach to Nonparametric Cluster Analysis*, IEEE Transactions on Computers, no. 9, pp. 936-944, 1976.
- [Lam92] L. Lam, S-W. Lee, C. Y. Wuen, *Thinning Methodologies – A Comprehensive Survey*, IEEE Transactions on PAMI, vol. 14, no. 9, pp. 869-885, 1992.

- [Lass98] K. Lasse, *Automatic High-precision Measurements of the Location and Width of Kikuchi Bands in Electron Backscatter Diffraction Patterns*, Journal of Microscopy, vol.190, pt.3, pp.375-391, 1998.
- [Leav92] V. F. Leavers, *Shape Detection in Computer Vision Using the Hough Transform*, Springer-Verlag, 1992.
- [Leav93] V. F. Leavers, *Which Hough transform?*, Image Understanding, vol. 58, no. 2, pp. 250-264, 1993.
- [Lim90] J. S. Lim, *Two-Dimensional Signal and Image Processing*, Englewood Cliffs, NJ, Prentice Hall, pp. 536-540, 1990.
- [Lo95] Rong-Chin Lo, Wen-Hsiang Tsai, *Gray-scale Hough Transform for Thick Line Detection in Gray-scale Images*, Pattern Recognition, vol. 28, no. 5, pp. 647-661, 1995.
- [Lysa03] M. Lysaker, A. Lundervold, X. C. Tai, *Noise Removal Using Fourth-Order Partial Differential Equation With Applications to Medical Magnetic Resonance Images in Space and Time*, IEEE Transactions on Image Processing, vol. 12, no. 12, pp. 1579-1590, 2003.
- [Mali02] W. Malina, S. Ablameyko, W. Pawlak, *Podstawy cyfrowego przetwarzania obrazów*, Akademicka Oficyna Wydawnicza Exit, Warszawa 2002.
- [Mall99] S. G. Mallat, *A Wavelet Tour of Signal Processing*, Elsevier, ISBN: 012466606X, 1999.
- [Mari91] A. Marion, *An Introduction to Image Processing*, Chapman and Hall, p. 274, 1991.
- [Math1] *Mathworks*, www.mathworks.com
- [Mirp04] K. Mirpuri, H. Wendrock, et al., *High Temperature Behavior of Cu Films Studied in-situ by Electron Backscatter Diffraction*, European Workshop on Materials for Advanced Metallization, Brussels, Belgium, Elsevier, 2004.
- [Mora02] A. Morawiec, J. J. Fundenberger, E. Bouzy and J. S. Lecomte, *EP – a Program for Determination of Crystallite Orientations from TEM Kikuchi and CBED Diffraction Patterns*, Journal of Applied Crystallography, vol. 35, pp. 287, 2002.
- [Motw04] M. Motwani, M. Gadiya, R. Motwani, *A Survey of Image Denoising Techniques*, in Proceedings of GSPx, 2004.
- [Naka97] S. Nakamura, G. Fasol, S. J. Pearton, *The Blue Laser Diode*, Springer, Berlin, p. 23, 1997.
- [Naso96] G. P. Nason, *Wavelet Shrinkage by Cross-validation*, Journal of Royal Statistics Society, vol. 58, pp. 463-479, 1996.
- [Oie1] *Oxford Instruments Electron Backscatter Diffraction site*, www.ebsd.com
- [Osow94] S. Osowski, *Sieci neuronowe*, Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa 1994.
- [Otho02] M. A Othon, *Electron Back-Scattered Diffraction Misorientation Mapping Applied to Stress Corrosion Cracking of Stainless Steels*, Microscopy & Microanalysis, 2002.
- [Piaz04] S. Piazolo, M. W. Jessell, D. J. Priori, P. D. Bons, *The Integration of Experimental in-situ EBSD Observations and Numerical Simulations: a Novel Technique of Microstructural Process Analysis*, Journal of Microscopy, vol. 213, pp. 273-284, 2004.
- [Piaz04] S. Piazolo, G. Steward, N. Seaton, D. J. Priori, *Recrystallization and Phase Transformation in Polycrystalline Materials: New Insights in Processes Using Combined in-situ Heating Experiments and Detailed EBSD Analysis*, Materials Science Forum, pp. 1407-1412, 2004.

- [Piaz05] S. Piazoło, D. J. Priori, M. D. Holness, *The Use of Combined Cathodoluminescence and EBSD Analysis: A Case Study Investigating Grain Boundary Migration Mechanisms in Quartz*, *Journal of Microscopy*, vol. 217, pp. 152-161, 2005.
- [Pita01] I. Pitas, *Digital Image Processing Algorithms and Applications*, John Wiley & Sons; 1 edition, ISBN: 0471377392, 2001.
- [Plat00] K. N. Plataniotis, A. N. Venetsanopoulos, *Color Image Processing and Applications*, Berlin, Springer, 2000.
- [Poli1] *Politechnika Śląska, Katedra Podstaw Konstrukcji Maszyn* <https://kpk.m.polsl.pl/>
- [Port01] J. Portilla, V. Strela, M. J. Wainwright, E. P. Simoncelli, *Adaptive Wiener denoising Using a Gaussian Scale Mixture Model in the Wavelet Domain*, *Proceedings of the 8th International Conference on Image Processing*, Thessaloniki, Greece, 2001.
- [Prin92] J. Princen, J. Illingworth, J. Kittler, *A Formal Definition of the Hough Transform: Properties and Relationships*, *Journal of Mathematical Imaging and Vision*, vol. 1, 1992.
- [Prio99] D. Prior, A. Boyle, F. Brenker, M. Cheadle, A. Day, G. Lopez, L. Peruzzo, G. Potts, S. Reddy, R. Spiess, N. Timms, P. Trimby, J. Wheeler, L. Zetterström, *The Application of Electron Backscatter Diffraction and Orientation Contrast Imaging in the SEM to Textural Problems in Rocks*, *American Mineralogist*, vol. 84, pp. 1741-1759, 1999.
- [Prio99] D. J. Prior, J. Wheeler, *Feldspar fabrics in a Greenschist Facies Albite-rich Mylonite from Electron Backscatter Diffraction*, *Tectonophysics*, vol. 303, pp. 29-49, 1999.
- [Rand00] V. Randle, O. Engler, *Introduction to Texture Analysis Macrotecture, Microtexture and Orientation Mapping*, ISBN 90-5699-224-4, Harwood Academic, Feb. 2000.
- [Rand92] V. Randle, *Microtexture Determination and Its Applications*, Institute of Materials, 1992.
- [Rat181] P. A. Ratley, A. G. Lundgren, *Sampling the 2-D Radon Transformation*, *IEEE Acoustic Speech Signal Processing*, vol. 29(5), 1981.
- [Riss89] T. Risse, *Hough Transform for Line Recognition*, *Computer Vision and Image Processing*, vol. 46, pp. 327-345, 1989.
- [Rosc1] <http://pawelrosczak.republika.pl/mlp/mlp.html#classification>
- [Rudi92] L. I. Rudin, S. Osher, E. Fatemi, *Nonlinear Total Variation Based Noise Removal Algorithms*, *Physica D*, vol. 60, pp. 259-268, 1992.
- [Sahh97] P. Sahho, C. Wilkins, J. Yeager, *Threshold Selection Using Renyi's Entropy*, *Pattern Recognition*, vol. 30, pp. 71-84, 1997.
- [Schu04] J. W. Schultze, B. Davepon, F. Karman, C. Rosenkranz, A. Schreiber, O. Voigt, *Corrosion and Passivation in Nanoscopic and Microscopic Dimensions: the Influence of Grains and Grain Boundaries*, *Corrosion Engineering, Science and Technology*, vol. 39, no. 1, pp. 45-52, 2004.
- [Schw00] A. J. Schwarz, M. Kumar, B. L. Adams, *Electron Backscatter Diffraction in Materials Science*, ISBN 0-306-46487-X, Kluwer Academic/Plenum Publishers, 2000.
- [Schw98] R. A. Schwarzer, *Crystallography and Microstructure of Thin Films Studied by X-ray and Electron Diffraction*, *Materials Science Forum*, vol. 287-288, pp. 23-60, 1998.
- [Skar93] W. Skarbek, *Metody reprezentacji obrazów cyfrowych*, Akademicka Oficyna Wydawnicza PLJ, Warszawa, 1993.

- [Star02] J. L. Starck, J. L. SE. Candès, D. L. Donoho, *Curvelet Transform for Image Denoising*, IEEE Transactions on Image Processing, vol. 11, no. 6, pp. 670-684, 2002.
- [Star03] J. L. Starck, F. Murtagh, E. Candès, D. L. Donoho, *Gray and Color Image Contrast Enhancement by the Curvelet Transform*, IEEE Transactions on Image Processing, vol. 12, no. 6, pp. 706-717, 2003.
- [Star94] J. L. Starck, A. Bijaoui, *Filtering and Deconvolution by the Wavelet Transform*, Signal Processing, vol. 35, pp. 195-211, 1994.
- [Star99] J. L. Starck, F. Murtagh, *Multiscale Entropy Filtering*, Signal Processing, vol. 76, pp. 147-165, 1999.
- [Stre00] V. Strela, *Denoising via Block Wiener filtering in Wavelet Domain*, in 3rd European Congress of Mathematics, Barcelona, Jul. 2000.
- [Szab90] J. Szabatin, *Podstawy teorii sygnałów*, Wydawnictwa Komunikacji i Łączności, Warszawa, 1990.
- [Sztw06a] K. Sztwiertnia, M. Bieda, G. Sawina, *Determination of Crystallite Orientations Using TEM. Examples of Measurements*, Archives of Metallurgy, vol. 51, no. 1, pp. 55-62, 2006.
- [Sztw06b] K. Sztwiertnia, M. Faryna, G. Sawina, *Misorientation Characteristics of Interphase Boundaries in Particulate Al₂O₃-based Composites*, Journal of the European Ceramic Society, vol. 26, pp. 2973-2978, 2006.
- [Tade93] R. Tadeusiewicz, *Sieci neuronowe*, Akademicka Oficyna Wydawnicza RM., Warszawa 1993.
- [Toft96] P. Toft, *The Radon Transform – Theory and Implementation*, Ph. D. Thesis, Department of Mathematical Modeling, Technical University of Denmark, 1996.
- [Trag02] C. Trager-Cowan, F. Sweeney, J. Hastie, S. K. Manson-Smith, D. A. Cowan, C. T. Foxon, I. Harrison, S. D. Hersee, D. McColl, A. Mohammed, S. V. Novikov, K. P. O'Donnell, D. Zubia, *Characterization of Nitride Thin Films by Electron Backscatter Diffraction*, Journal of Microscopy, vol. 205, pp. 226-230, Mar. 2002.
- [Trim00] P. W. Trimby, M. R. Drury, C. J. Spiers, *Recognizing the Crystallographic Signature of Recrystallisation Processes in Deformed Rocks: a Study of Experimentally Deformed Rocksalt*, Journal of Structural Geology, vol. 22, pp.1609-1620, 2000.
- [Vand02] L. Vandeveld, J. Melkebeek, *Modeling of Magnetoelastic Material*, IEEE Transactions on Magnetics, vol. 38, no. 2, pp. 993-996, 2002.
- [Vena73] J. A. Venables, C. J. Harland, *Electron Back-Scattering Patterns – A New Technique for Obtaining Crystallographic Information in the Scanning Electron Microscope*, Philosophical Magazine vol. 27, pp. 1193-1200, 1973.
- [Wiki1] *Encyklopedia internetowa*, <http://pl.wikipedia.org/>
- [Wrig91] S. I. Wright, B. L. Adams, *Automated Lattice Orientation Determination from Electron Backscatter Kikuchi Diffraction Patterns*, Textures & Microstructures, vol. 14-18, pp. 273-278, 1991.
- [Yang95] R. Yang, L. Yin, M. Gabouj, Y. Neuvo, *Optimal Weighted Median Filtering under Structural Constraints*, IEEE Trans. Signal Processing, vol. 43, pp. 591-603, 1995.
- [Yip92] R. K. K. Yip, P. K. S. Tam, D. N. K. Leung, *Modification of the Hough Transform for Circles and Ellipses Detection Using a 2-dimensional Array*, Pattern Recognition, vol. 25, pp. 1007-1022, 1992.

- [You00] Yu-Li You, M. Kaveh, *Fourth Order Partial Differential Equations for Noise Removal*, IEEE Transactions on Image Processing, vol. 9, no. 10, pp. 1723-1730, Oct. 2000.
- [Yuan03] Y. Yuan, A. J. Davenport, M. Strangwood, R. Ambat, *The Effect of Crystallographic Misorientation on Intergranular Corrosion of Aluminum Alloy 5182*, 44th Corrosion Science Symposium, Electrochem 2003, Southampton, UK, Sep. 2003.
- [Yuen89] H. K. Yuen, J. Illingworth, J. Kittler, *Detecting Partially Occluded Ellipses Using the Hough Transform*, Image and Vision Computing, vol. 7, pp. 31-37, 1989.
- [Ziel02] T. P. Zieliński, *Od teorii do cyfrowego przetwarzania sygnałów*, Wydział EAIiE AGH, Kraków, 2002.
- [Ziel05] T. P. Zieliński, *Cyfrowe przetwarzanie sygnałów – od teorii do zastosowań*, Wydawnictwa Komunikacji i Łączności, Warszawa, 2005.