AKADEMIA GÓRNICZO – HUTNICZA

IM. STANISŁAWA STASZICA W KRAKOWIE

WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI, INFORMATYKI I ELEKTRONIKI

KATEDRA AUTOMATYKI

Praca magisterska

Generowanie modeli symulacyjnych układów logicznych na podstawie schematów drukowanych i odręcznych

Imię i nazwisko:

Tomasz Mędrala

Marcin Motyka

Kierunek studiów:

Automatyka i robotyka (SUM)

Promotor:

dr inż. Paweł Rotter

Kraków 2009

Spis treści

Cel i zakres pracy

Celem pracy było napisanie oprogramowania do automatycznego generowania modelu umożliwiającego symulację układu logicznego na podstawie obrazu schematu logicznego, uzyskanego z wykorzystaniem kamery CCD lub aparatu cyfrowego.

Aby zrealizować powyższe cele należało poszerzyć zakres wiadomości z dziedziny komputerowego przetwarzania i akwizycji obrazów oraz opanować podstawowe funkcje modułu Matlab (Image Processing Toolbox), w którym była pisana aplikacja.

Praca składa się z części teoretycznej i praktycznej. Część teoretyczna została opracowana pod kątem podstawowych informacji na temat przetwarzania i rozpoznawania obrazów. Zdefiniowano wybrane pojęcia związane z reprezentacją obrazu, jego akwizycją, przetwarzaniem i rozpoznawaniem. Opis części praktycznej zawiera podstawowe informacje praktyczne do realizacji programu, obrazy wejściowe i wyniki badań.

Niniejszą pracę wykonaliśmy wspólnie i nasz wkład w nią był następujący:

- ✓ Podstawowe definicje, akwizycja obrazów, przekształcenia opisane w podrozdziałach **1.1-1.5** zostały przygotowane przez Marcina Motykę,
- ✓ Rozpoznawanie obrazów, techniki segmentacji, analizę i opis cech geometrycznych opisane w podrozdziałach **1.6-1.8** przygotował Tomasz Mędrala,
- ✓ Analiza dotychczasowych rozwiązań z zakresu rozpoznawania obrazu umieszczona w rozdziale **2** została opracowana przez Marcina Motykę,
- ✓ Technikę rozpoznawania schematów logicznych za pomocą naszej aplikacji zamieszczoną w podrozdziale **3.1** opisał Tomasz Mędrala,
- ✓ Zrzuty ekranowe oraz przygotowanie rysunków w podrozdziale **3.2** wykonał Marcin Motyka natomiast treść uzupełnił Tomasz Mędrala,
- ✓ Wykonanie odręcznych schematów logicznych, ich akwizycję za pomocą aparatu cyfrowego Sony α350 wykonał Tomasz Mędrala,
- ✓ Przygotowanie obrazów do obróbki oraz dalszej analizy (binaryzacja, filtracja, przekształcenia morfologiczne oraz inne operacje w celu segmentacji obrazu) wykonał Marcin Motyka,

- ✓ Opracowanie algorytmów wydzielania cech z obrazu, obliczenia współczynników kształtu, niezmienników momentowych, trzy odrębne algorytmy dla rozpoznawania bramek AND, OR, NOT oraz ich negacji, rozpoznawanie bramki XOR oraz synteza tych parametrów w celu identyfikacji bramek wykonał Tomasz Mędrala,
- ✓ Rozpoznawanie połączeń, ich indeksacja oraz generowanie kodu w Prologu wykonał Marcin Motyka,
- ✓ Wnioski zostały opracowane wspólnie na bazie doświadczeń podczas pisania programu oraz analizy uzyskiwanych wyników,
- ✓ Opis zmiennych i funkcji zawartych w dodatkach **A**, **B** wykonał Tomasz Mędrala,
- ✓ Opracowanie Tabeli z wynikami współczynników kształtu i niezmienników momentowych na bazie serii narysowanych bramek zamieszczonej w dodatku **C** wykonał Marcin Motyka.

Dodatek A zawiera opis funkcji oraz zmiennych globalnych używanych w programie *bramki***.** W dodatku **B** umieszczono kod wynikowy realizowanego programu do generowania schematu i kodu wynikowego w Prologu. **Dodatek C** zawiera wykaz uzyskanych wartości współczynników kształtu i wartości używanych w programie współczynników momentowych.

1 Podstawy przetwarzania i rozpoznawania obrazów

1.1 Wprowadzenie

Rozpoznawanie obrazów i systemy wizyjne znajdują zastosowanie w coraz większej ilości dziedzin naukowych i technicznych. Wiąże się to z dużym rozwojem technologicznym procesów przemysłowych, badań naukowych, rozwoju medycyny, eksploracji kosmosu, badan wojskowych jak również podniesieniem standardu życia. Obecnie istnieje wiele systemów wizyjnych, które „czuwają" nad bezpieczeństwem, jakością nie tylko procesów przemysłowych, ale i również innych dziedzin.

Analizując zastosowania technik komputerowego przetwarzania obrazów [19] można wyróżnić szereg cech przewyższających ludzki wzrok:

- automatyka (polepszenie autonomiczności robotów wyręczających człowieka w wielu mechanicznych pracach, polepszenie jakości działania systemów regulacji automatycznej);
- medycyna (automatyczna analiza i rozpoznawanie obrazów preparatów tkankowych, rentgenogramów, ultrasonografów, obrazów z tomografii rentgenowskiej i NMR, itp.);
- kryminalistyka (wyszukiwanie w obrazie cech i szczegółów nie rozróżnialnych ludzkim wzrokiem);
- geodezja i kartografia (automatyczne przetwarzanie wielkich ilości danych obrazowych - zwłaszcza zdjęcia lotnicze i satelitarne);
- komunikacja (wykrywanie obecności, kierunku i natężenia ruchu pojazdów, określanie wielkości kolejki pojazdów czekających na skrzyżowaniach, automatyczne wykrywanie kolizji i wypadków);
- laboratoria badawcze (kontrola materiałów, kontrola jakości wyrobów);
- zabezpieczanie obiektów pod szczególnym nadzorem (kontrola bagaży, nadzór nad ruchem w pewnym obszarze, wykrywanie intruzów);
- wojskowość (kierowanie naprowadzaniem inteligentnej broni na współczesnym polu walki);
- astronomia i astrofizyka (analiza obrazów o zakresie widmowym przekraczającym możliwości ludzkiego wzroku).

1.2 Cele i zastosowania przetwarzania obrazów

Celem sztucznego przetwarzania lub analizy obrazu, jest takie automatyczne przetworzenie i przeanalizowanie obrazu wybranych obiektów lub całego otoczenia systemu zautomatyzowanego, aby uzyskać użyteczną informację na temat interesujących obiektów (na przykład będących przedmiotem manipulacji ze strony robota przemysłowego) lub na temat otoczenia, które może wpływać (i zwykle znacząco wpływa) na sterowany automatycznie proces [19]. Zatem do głównych celów przetwarzania obrazów należy:

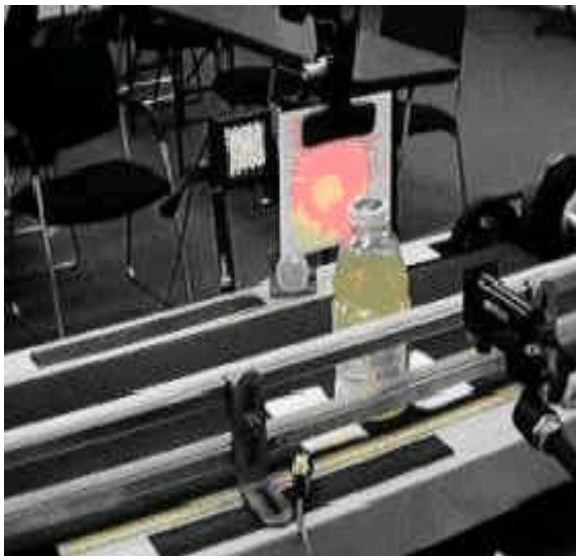
- poprawa subiektywnej jakości obrazu postrzeganej przez oko ludzkie;
- przetworzenie obrazu do postaci umożliwiającej pomiar wybranych jego cech, automatycznej analizy lub transmisji.

Praktycznymi dziedzinami, w których istnieje obecnie pilna potrzeba „sztucznego widzenia", w tym głównie przetwarzania, analizy i rozpoznawania obrazu, są między innymi:

- ***Automatyka- systemy kontroli jakości***

Systemy kontroli jakości opierają się na pomiarach i wyglądzie przedmiotu po obróbce. Duża seryjność produkcji powoduje, że potrzebne są urządzenia o dużej szybkości analizy obrazu, wielu narzędziach do pomiaru i rozpoznawania wad oraz współpracujące z urządzeniami automatyki na linii produkcyjnej. W systemach kontroli jakości występuje często konieczność bezdotykowej kontroli parametrów przedmiotu. Powyższe kryteria zmuszają producentów do stosowania zaawansowanych systemów wizyjnych zapewniających wykonanie w bardzo krótkim czasie wszystkich pomiarów oraz wymianę informacji z systemami automatyki. Systemy kontroli jakości są wykorzystywane w wielu dziedzinach przemysłu (motoryzacja, opakowania, farmacja, kosmetyki, elektronika, tworzywa sztuczne) do kontroli jakości produkowanych elementów. Szczególnie przydatne są w produkcji wielkoseryjnej, taśmowej. Zautomatyzowanie procesu kontroli jakości pozwala na wytwarzanie produktów najwyższej klasy będących kluczem do sukcesu nowoczesnej firmy. Dzięki szybkości działania i niezawodności pozwalają na maksymalne wykorzystanie możliwości przerobowych nowoczesnych urządzeń produkcyjnych przyczyniając się do zwiększenia zysków firm.

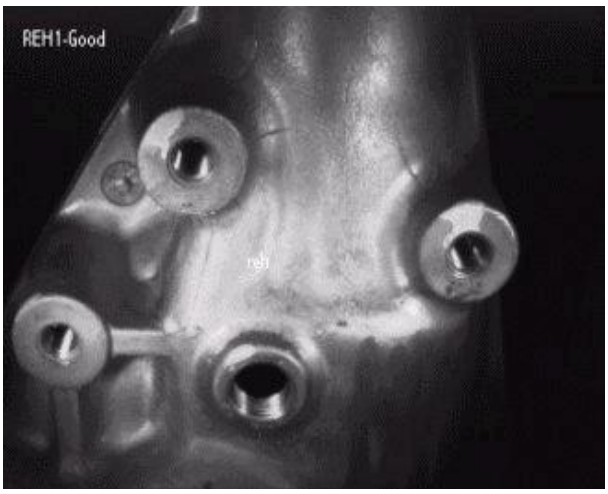
Rys. 1. Kontrola jakości szczelności butelek.



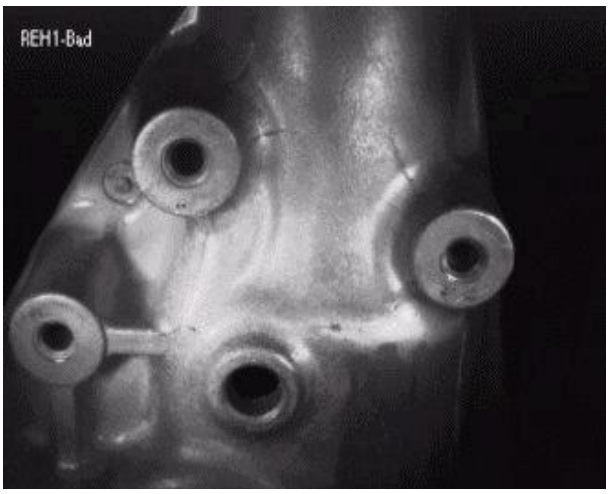
Rys. 2. Kontrola produktów butelkowych. ABIS http://www.abis.krakow.pl/index.asp

- ***Automatyka- diagnostyka w przemyśle motoryzacyjnym***

Na obudowie skrzyni biegów kontrolowane są gwinty w otworach do mocowania skrzyni. Zadanie diagnostyczne polega na końcowej weryfikacji otworów pod względem wykonania w nich gwintu. Zadanie wykonywane jest przy wykorzystaniu jednej kamery i oświetleniu z kilku źródeł korpusu skrzyni. Wykorzystuje się narzędzia do pomiaru intensywności światła w otworach. Powierzchnia wewnętrzna otworu bez gwintu widoczna na rysunku 4 nie odbija światła i widoczna jest jako ciemny punkt. Ocena ilości światła odbitego przez powierzchnię gwintową jest podstawą do oceny prawidłowego wykonania korpusu.

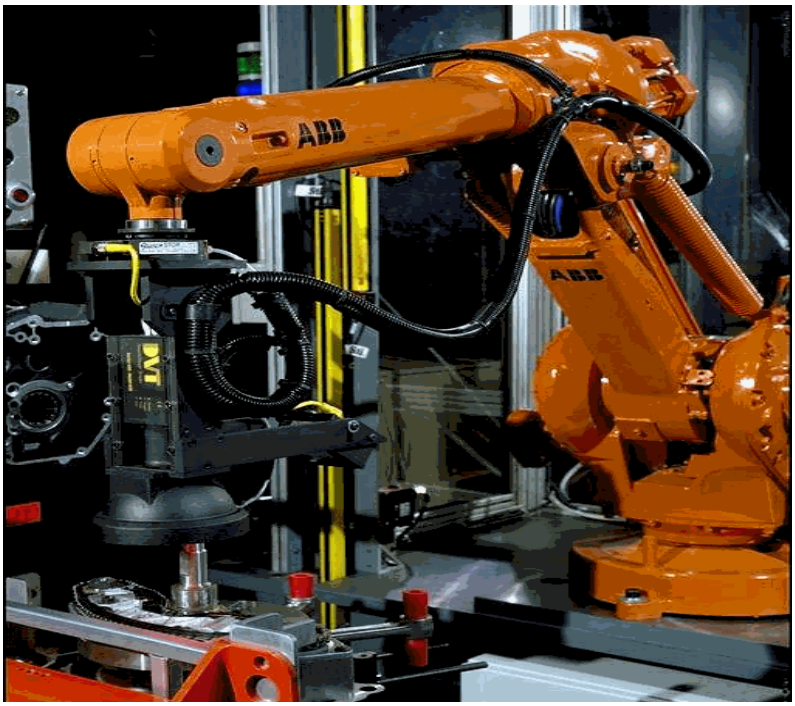


Rys. 3. Element korpusu z gwintem



Rys. 4. Element korpusu bez gwintu.

Zadanie diagnostyczne robota przedstawionego na rysunku 5 polega na zbadaniu kilkunastu parametrów silnika produkowanego na tej linii. Zadaniem robota jest pozycjonowanie kamery nad kolejnymi elementami silnika. Kamera na podstawie przygotowanego programu diagnostycznego przeprowadza inspekcję kolejnych elementów. Po wykonaniu zadania diagnostycznego przesyła informację do robota o gotowości do wykonania następnego zadania. Informacje i wyniki pomiarów kontrolowanych kolejno elementów przesyłane są do bazy danych w celu tworzenia historii produktu.



Rys. 5. Robot ABB na linii produkcyjnej diagnozujący parametry silnika.

1.3 Proces przetwarzania obrazu w systemie wizyjnym

Systemy wizyjne zyskują w ostatnich latach coraz większą popularność. Znajdują zastosowanie w coraz większej ilości dziedzin. Stosowane są w wyposażeniu robotów, przy nadzorowaniu procesów przemysłowych, sterowaniu ruchem drogowym, itp. Operacje przetwarzania obrazu pozwalają na łatwą analizę zdjęć satelitarnych powierzchni Ziemi. Przez zastosowanie odpowiedniej operacji przetwarzania obrazów można uzyskać informacje, które normalnie nie są rozróżniane przez system wzrokowy człowieka [19]. Do głównych zadań systemu wizyjnego należy:

- Uzyskanie cyfrowej reprezentacji obrazu (recepcja, akwizycja);
- Przetworzenie obrazu cyfrowego z wykorzystaniem technik komputerowych;
- Analiza i przetworzenie rezultatów w celu sterowania robotami, kontroli automatycznych procesów, kontroli jakości, itp.

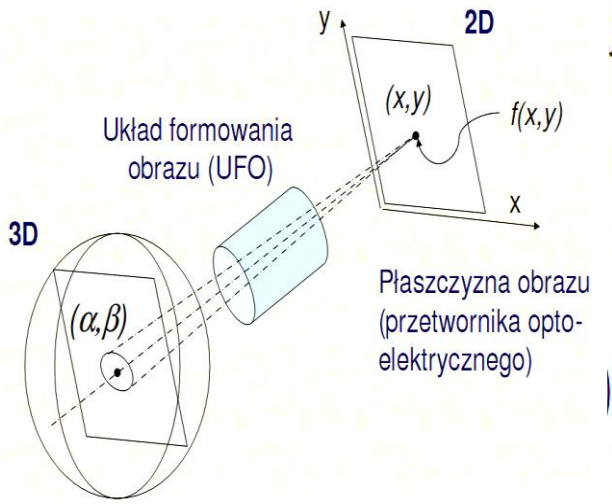
Opisane w niniejszej pracy metody klasyfikacji obiektów znajdują praktyczne zastosowanie w systemach widzenia maszynowego. Systemy te są obecne w wielu gałęziach szeroko pojętego przemysłu od branży spożywczej po zbrojeniową oraz w różnorodnych zastosowaniach multimedialnych (tworzenie filmów animowanych, śledzenie obiektów i ich zamiana na inne.). Wykorzystanie urządzeń optycznych wraz z odpowiednim oprogramowaniem rozpoznawania obrazu w znacznym stopniu usprawnia proces kontroli końcowej czy pozwala na bieżąco kontrolować stan zestawianych na taśmie produkcyjnej urządzeń.

1.4 Reprezentacja i analiza obrazu

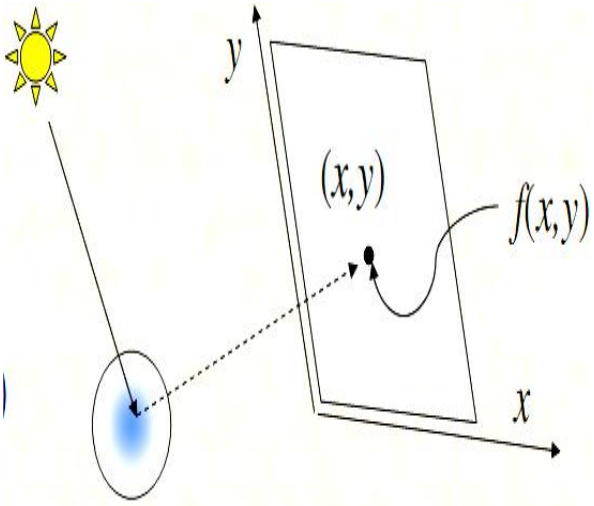
1.4.1 Model formowania obrazu

Obraz - dwuwymiarowa funkcja intensywności światła $f(x,y)$, gdzie wartość lub amplituda ***f*** w przestrzennych współrzędnych ***(x,y)*** określa intensywność (jasność) obrazu w tym punkcie. Ponieważ światło jest formą energii, ***f(x,y)*** musi być niezerowa i skończona:

$$0 \leq f(x,y) < \infty$$



Rys. 6. Układ formowania obrazu.



Rys. 7. Rzutowanie współrzędnych świata trójwymiarowego na współrzędne dwuwymiarowe detektora.

Na wartość $f(x,y)$ mają wpływ dwa elementy. Jednym jest wielkość promieniowania świetlnego, które pada na oświetloną scenę. Drugim jest ilość światła odbitego przez obiekty w scenie. Obiekty te są zwane odpowiednio iluminacją i odbiciem (ilumination, reflectance) i są opisane jako $i(x,y)$ i $r(x,y)$

zatem

$$f(x,y) = i(x,y)r(x,y)$$

gdzie:

$0 < i(x,y) < \infty$ - *iluminacja (x,y)*

$0 < r(x,y) < 1$ -*współczynnik odbicia*

Dla liniowego procesu akumulacji energii w płaszczyźnie obrazu:

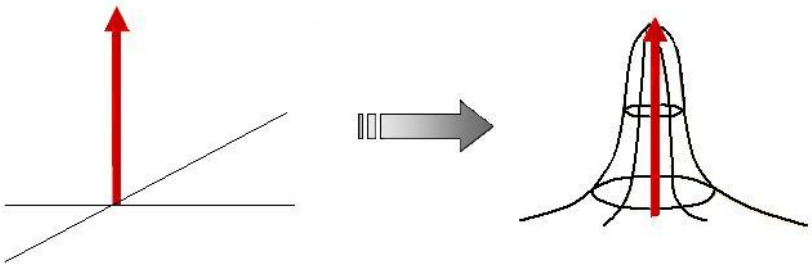
$$f(x,y) = \iint_{-\infty}^{\infty} f(\alpha,\beta)h(x,y,\alpha,\beta)d\alpha d\beta$$

gdzie:

$h(x,y,\alpha,\beta)$- tzw. funkcja rozmycia punktu (ang. *point spread function*) opisująca wpływ promieniowania otoczenia wybranego punktu obiektu (α,β) na energię odpowiadającego mu punktowi obrazu (x,y).

Jeżeli funkcja rozmycia punktu jest niezmienna względem przesunięcia to układ formowania obrazu opisuje całka splotowa:

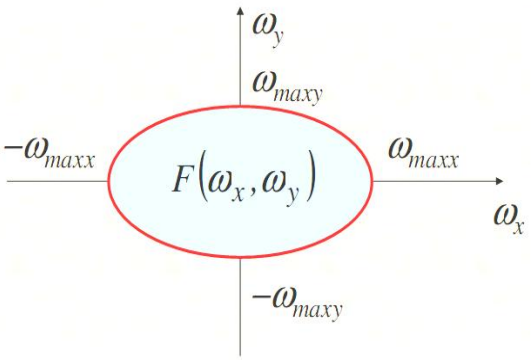
$$f(x,y) = \iint_{-\infty}^{\infty} f(\alpha,\beta)h(x-\alpha,y-\beta)d\alpha d\beta$$



Rys. 8. Funkcja rozmycia punktu.

$$h(x,y) = exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

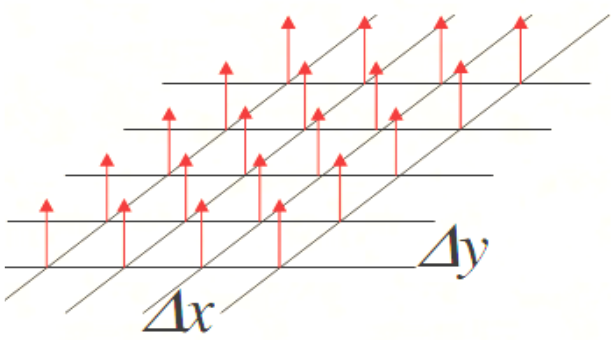
zakładamy, że obraz analogowy charakteryzuje się ograniczoną szerokością widma Fourierowskiego.



Rys. 9. Szerokość widma Fourierowskiego.

W komputerowym systemie przetwarzania obrazów, obraz analogowy (o ograniczonym widmie) jest próbkowany i kwantowany za pomocą dwuwymiarowej funkcji próbkującej:

$$S(x,y) = \sum_{i=0}^{M-1} \sum_{k=0}^{N-1} \delta(x - i\Delta x, y - k\Delta y)$$



Rys. 10. Dwuwymiarowa funkcja próbkująca.

Obraz spróbkowany wyraża się wzorem:

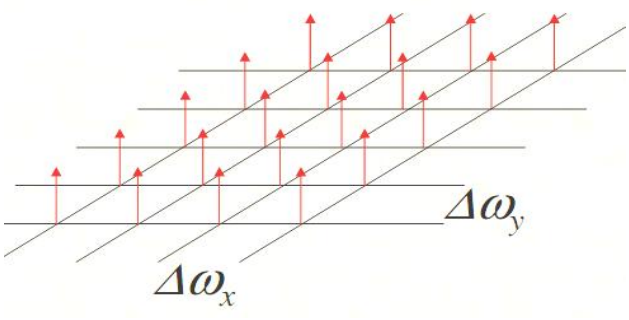
$$f_s(x,y) = f(x,y)S(x,y) = \sum_{i=0}^{M-1} \sum_{k=0}^{N-1} f(i\Delta x, k\Delta y)\delta(x - i\Delta x, y - k\Delta y)$$

W dziedzinie widma Fouriera, próbkowanie przestrzenne obrazu odpowiada splotowi widma obrazu oraz widma funkcji próbkującej. Widmo funkcji spróbkowanej przedstawia się wzorem:

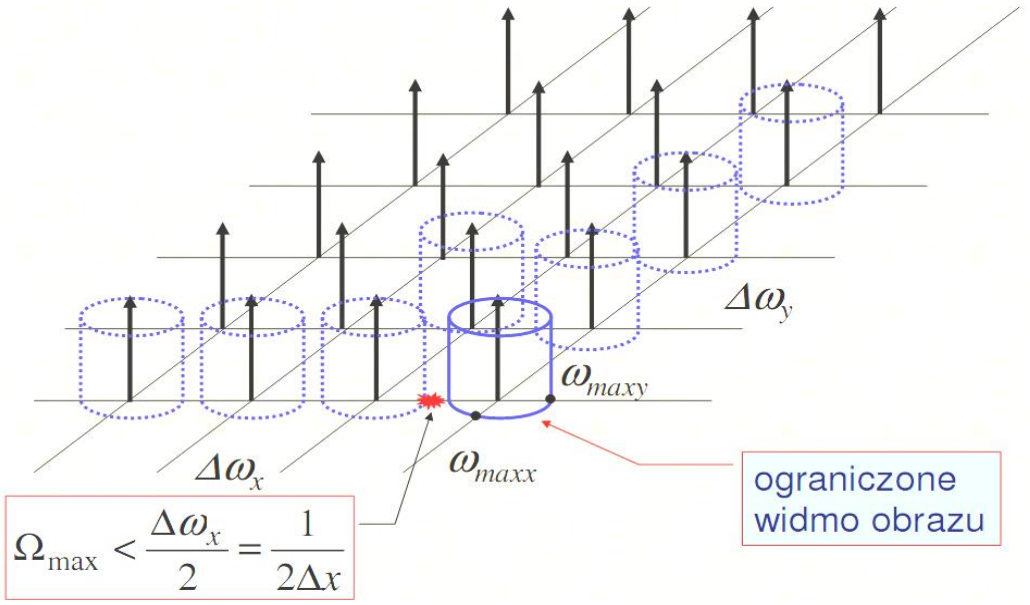
$$F_s(\omega_x, \omega_y) = \frac{1}{\Delta x \Delta y} \sum_{i=0}^{M-1} \sum_{k=0}^{N-1} F(\omega_x - i\Delta\omega_x, \omega_y - k\Delta\omega_y)$$

gdzie:

$$\Delta\omega_x = \frac{1}{\Delta x}, \Delta\omega_y = \frac{1}{\Delta y}$$



Rys. 11. Próbkowanie przestrzenne widma obrazu.



Rys. 12. Powielenie okresowe widma obrazu z próbkowanego.

Treść oraz rysunki powyższych rozważań zostały zaczerpnięte z materiałów dydaktycznych [16] pana P. Strumiłło z Instytutu Elektroniki Politechniki Łódzkiej.

Jedną z podstawowych cech obrazu jest format, w jakim jest zapisany, obejmujący rozdzielczość przestrzenna oraz ilość poziomów szarości (poziomów barw), tzw. rozdzielczość barwną. Biorąc pod uwagę ilość poziomów barw (szarości) wyróżnia się obrazy czarno-białe, obrazy monochromatyczne (tzw. wielo-odcieniowe) oraz obrazy kolorowe [22].

Częstym pojęciem występującym przy analizie obrazów jest poziom szarości (gray level) ***l.*** Jest on zdefiniowany jako intensywność obrazu monochromatycznego *f* w punkcie *(x,y)*

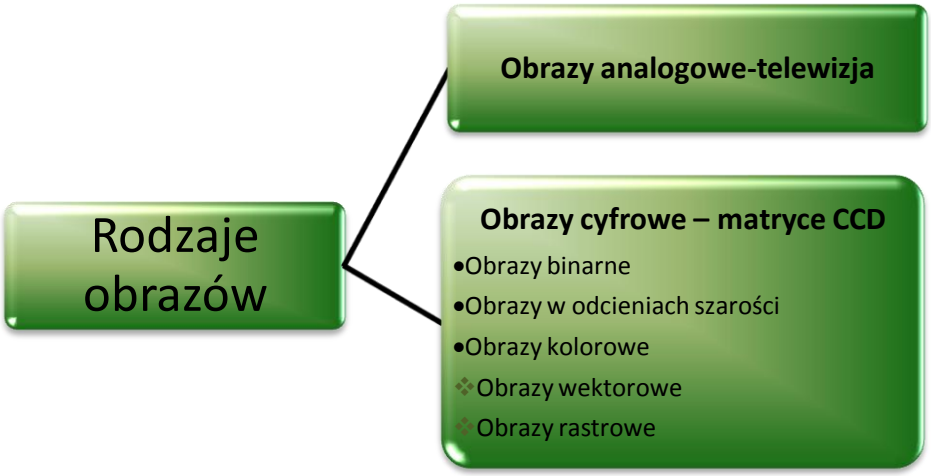
$$L\_min < l < L\_max$$

gdzie:

$$L_{min} = i_{min}\, r_{min}$$
$$L_{max} = i_{max}\, r_{max}$$

a przedział $[L_{min}, L_{max}]$ jest zwany ***skalą szarości*** *(gray scale).* Często przesuwa się ten przedział do zakresu $[0, L]$, gdzie $l = 0$ oznacza czerń, a $l = L$ oznacza biel w rozważanej skali szarości [18].

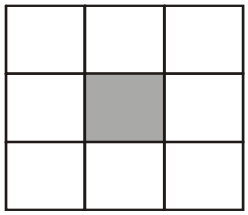
1.4.2 Rodzaje obrazów cyfrowych



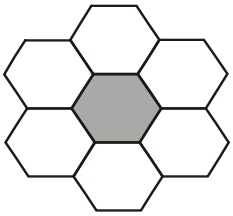
Rys. 13. Rodzaje obrazów cyfrowych.

Istnieją dwie podstawowe formy reprezentacji cyfrowej obrazu:

- Siatka heksagonalna;
- Siatka kwadratowa;

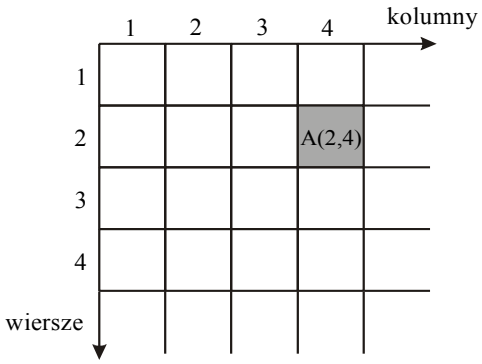


Rys. 14. Siatka kwadratowa.



Rys. 15. Siatka heksagonalna.

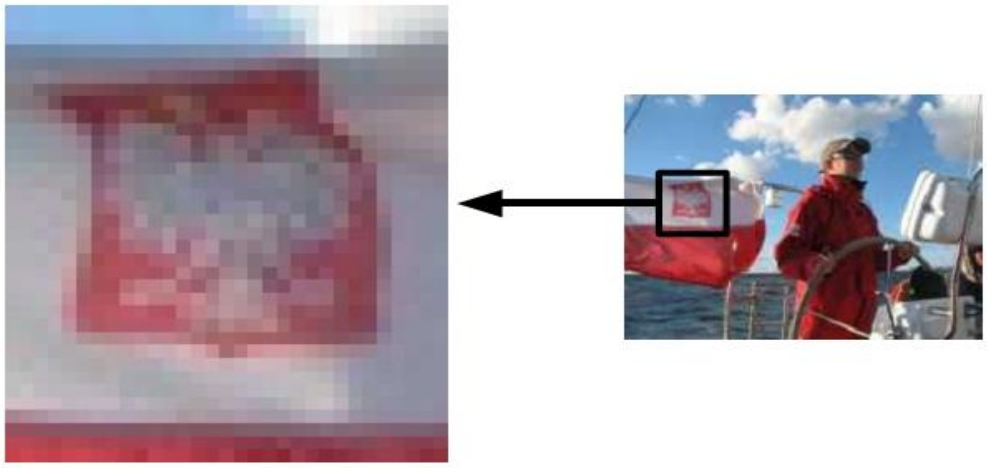
Reprezentacja za pomocą siatki heksagonalnej jest zbliżona do naturalnego układu receptorów w oku ludzkim, lecz rzadko wykorzystywana w analizie cyfrowej obrazu.
Najbardziej rozpowszechniona i szeroko stosowana w komputerowej analizie obrazu jest siatka kwadratowa ze względu na prostotę i wygodniejszą obsługę.



Rys. 16. Obraz jako tablica dwuwymiarowa

Obraz cyfrowy, rozumiemy jako dwuwymiarową funkcję dyskretną ***f(x,y)*** taką, że jej argumentami są dyskretne piksele, a wartościami kolory w przestrzeni RGB, czyli trójki liczb z przedziału [0,255]. Początek układu współrzędnych przyjmujemy w lewym górnym rogu obrazu [8].

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(N-1,0) & f(N-1,1) & \cdots & f(N-1,N-1) \end{bmatrix} \quad [19]$$

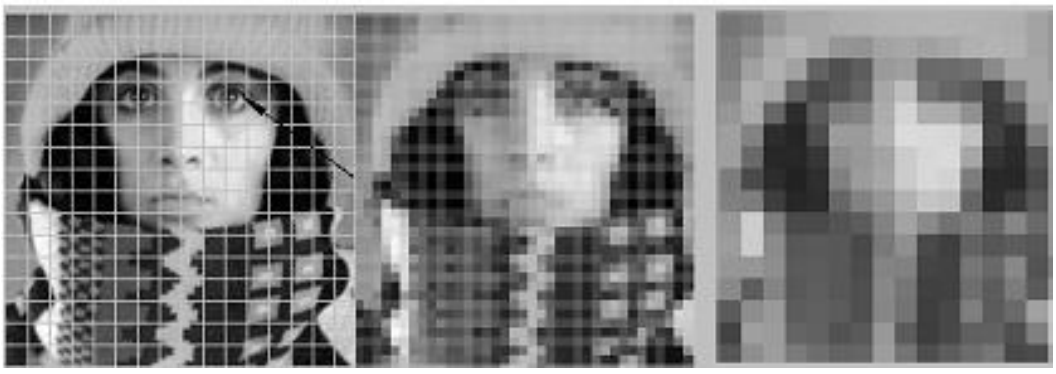


Rys. 17. Obraz cyfrowy rastrowy w powiększeniu.

Każdy ***piksel*** (element tablicy obrazu cyfrowego) zawiera wektor (kanałów) opisujący kolor:

- 1 - bitowy dla obrazów binarnych;
- 1 - elementowy dla obrazów monochromatycznych;
- 3 - elementowy dla obrazów barwnych ([R G B], [H S V], [L a b],…);
- wieloelementowy dla obrazów multimodalnych [1].

Obraz jest zawsze reprezentowany w formie próbkowanej oraz skwantowanej.



Rys. 18. Etapy próbkowania i kwantyzacji obrazu cyfrowego.

Jako stopień rozróżnialności detali definiuje się pojęcie ***rozdzielczości.*** O rozdzielczości przestrzennej i rozdzielczości poziomów jasności obrazu rastrowego, decyduje typ próbkowania i kwantowania sygnału wizyjnego.

Wybór właściwej rozdzielczości obrazu jest sprawą bardzo ważną, gdyż rozdzielczość jest miarą zdolności rozpoznawania szczegółów obrazu [19].

Rozdzielczość przestrzenna jest to liczba punktów zwanych pikselami obrazowymi przypadających na jednostkę długości i szerokości skanowanego obrazu. Pod terminem rozdzielczości należy rozróżnić:

- Rozdzielczość wejściową lub rozdzielczość skanowania. Jest to gęstość punktów próbkowania informacji na danej powierzchni (zwykle pikseli na cal ppi lub pikseli na centymetr), które urządzenie może przechwycić;
- Rozdzielczość wyjściowa, czyli miara punktów na cal definiuje gęstość znaków drukowanych w poziomie przez naświetlarki i drukarki postscriptowe;
- Standard obrazu - jest to całkowita liczba punktów obrazu rastrowego;
- Standard ekranu - która jest definiowana, jako liczba próbek dyskretnych obrazu mierzonych w pionie i poziomie, które mogą być wyświetlone przez monitor.

Rozdzielczość poziomów jasności – zwana też rozdzielczością bitową lub głębią barw określa zdolność urządzenia np. skanera do rozróżniania stopni jasności skanowanego obrazu. Związana jest z kwantowaniem sygnału, czyli z procesem, w którym każdemu punktowi w obrazie przyporządkowany jest jeden poziom jasności ze skończonego zbioru tych poziomów.

W komputerowym przetwarzaniu obrazów wyróżniamy dwa podstawowe typy obrazów: **rastrowe i wektorowe** [9]**.**

Obrazy wektorowe - tworzone są za pomocą wyrażeń matematycznych opisujących linie, elipsy, krzywe. Możemy je generować za pomocą programów graficznych, aplikacji CAD, 3D, CorelDraw itp. [9].

Obrazy rastrowe - są wprowadzane do komputera za pomocą takich urządzeń jak skanery, aparaty cyfrowe, stacje robocze PhotoCD, karty do przechwytywania obrazów wideo itp. [9].

Obrazy cyfrowe rastrowe powstają w wyniku próbkowania i kwantowania sygnału wizyjnego.

Rys. 19. Porównanie obrazu rastrowego z wektorowym.

Obraz cyfrowy rastrowy może być rozpatrywany w dziedzinie przestrzennej lub w dziedzinie częstotliwości.

W dziedzinie przestrzennej obraz reprezentowany jest jako macierz ***W*** która posiada swój wymiar ***W[MxN],*** elementami macierzy są piksele. Piksele te są reprezentowane przez liczby całkowite z przedziału ***<0,L-1>,*** gdzie ***L*** reprezentuje liczbę poziomów jasności danego punktu [9].

W dziedzinie częstotliwości obraz, a właściwie jego widmo uzyskane np. za pomocą transformaty Fouriera, jest reprezentowane przez dyskretną funkcję określaną za pomocą macierzy ***W***. Elementy macierzy są adresowane za pomocą współrzędnych ***1/x, 1/y*** lub ***u,v***, które określają tzw. Częstotliwości przestrzenne. Element widma o współrzędnych ***0,0*** nazywany jest składową stałą i jest proporcjonalny do średniej jasności obrazu [9].

W komputerowej technice przetwarzania obrazów najczęściej korzysta się z następujących formatów zapisu poziomów szarości lub nasycenia barw w obrazie:

- **Obraz binarny** (1-bit per pixel (bpp)) - dwupoziomowe (białe-czarne);

Rys. 20. Obraz binarny.

- **Obraz pseudobinarny** (dwupoziomowe, czarno-białe (0,255));
- **Obraz monochromatyczny** (8 bpp) - szaro odcieniowe. Za pomocą tego formatu można zakodować 256 poziomów szarości;

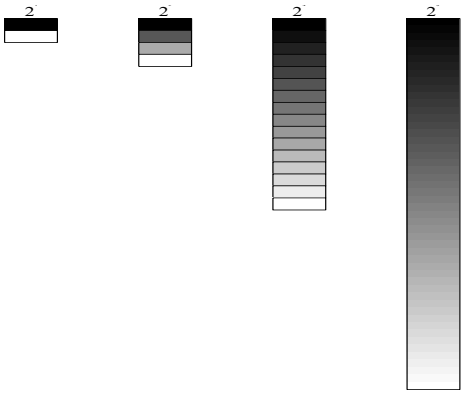
Rys. 21. Obraz monochromatyczny.

- **Obraz kolorowy**
  - **24 bpp** często po 8 kolejnych bitów opisują jasność jednej z trzech podstawowych barw RGB (red-green-blue). Za pomocą tego formatu można zakodować 16,7 mln barw.
  - **32 bpp**, zwane też CMY
  - **48 bpp**, 281 miliardów barw

Rys. 22. Obraz kolorowy.

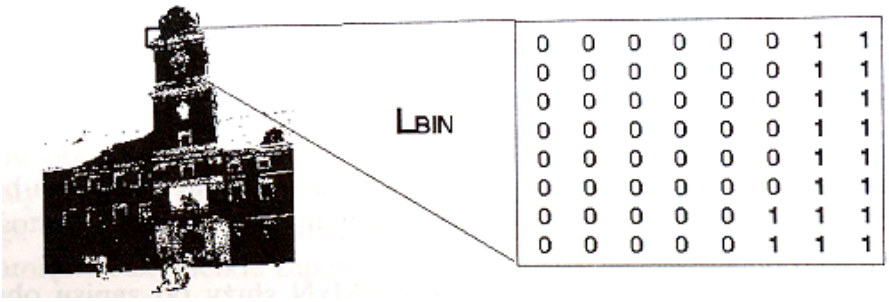
Dobór właściwych, z punktu widzenia przetwarzania obrazu, poziomów szarości dla obrazów monochromatycznych lub nasycenia poszczególnych kolorów jest zawsze kompromisem, pomiędzy jakością obrazu, i - co a za tym idzie - ilością ukrytej w nim informacji, a zajętością pamięci [19].

Poziom szarości – intensywność obrazu czarno – białego *f* w punkcie *(x,y)*

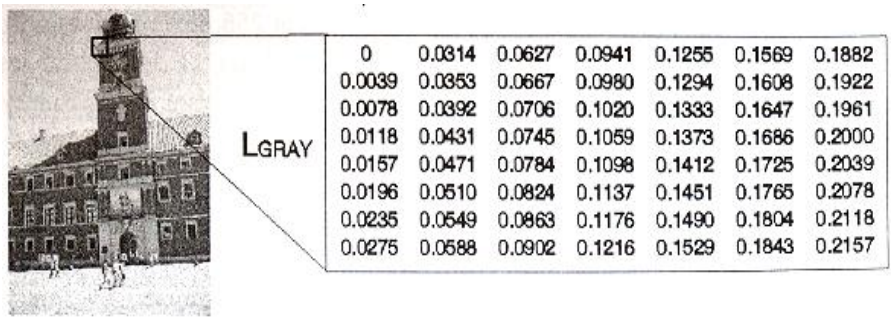


Skala szarości		Zakres skali
2^1	2 wartości	0,1
2^2	4 wartości	0,…,3
2^4	16 wartości	0,…,15
2^8	256 wartości	0,…,255

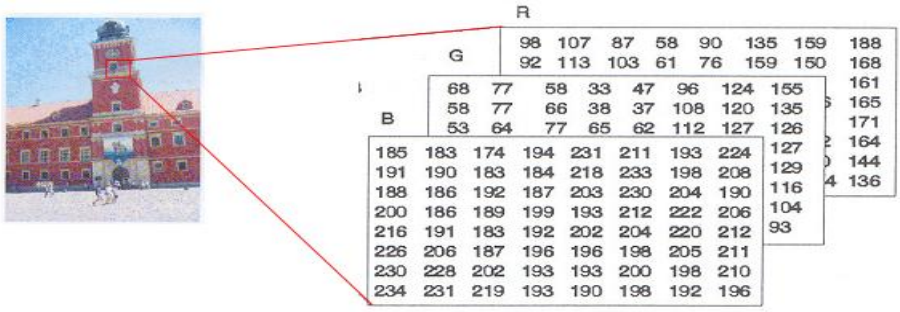
Rys. 23. Kwantyzacja obrazu cyfrowego.



Rys. 24. Cyfrowa reprezentacja obrazu - obraz binarny.



Rys. 25. Cyfrowa reprezentacja obrazu - obraz w skali szarości.

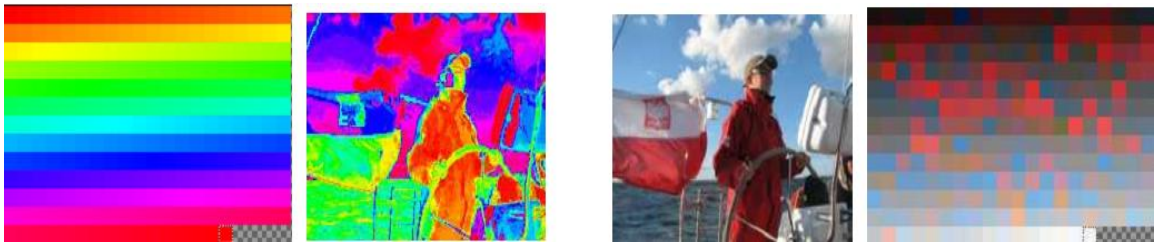


Rys. 26. Cyfrowa reprezentacja obrazu - obraz kolorowy.

Pikselizacja jest to przejście z obrazu o praktycznie nieskończonej rozdzielczości na obraz cyfrowy.

1.4.3 Kwantyzacja kolorów

Kwantyzacja barwy jest procesem transformacji obrazu barwnego opisanego przy pomocy 8 bitów (na każdą składową barwną RGB) na obraz barwny zawierający mniejszą liczbę specjalnie wybranych barw (paleta barw). Jest to, więc proces redukcji liczby barw w obrazie. Proces kwantyzacji barwy składa się z 2 etapów: projektowania palety barw i odwzorowania pikseli obrazu oryginalnego na barwy z palety. Paleta barw może być stała (uniwersalna) lub adaptacyjna (dostosowywana do każdego obrazu).



Rys. 27. Reprezentacja obrazu z różną paletą barw.

- ***Metoda kwantyzacji barwy ze stałą paletą*** jest najprostszą i najszybszą metodą i polega na równomiernej kwantyzacji barwy w przestrzeni RGB. Główną zasadą stosowaną w tej metodzie jest zasada kwantyzacji skalarnej, tzn. kwantyzacji każdej składowej RGB osobno.
- ***Metoda kwantyzacji w przestrzeni HSV*** wykorzystuje percepcyjną przestrzeń barw HSV.

Obok ogólnych miar teorio-sygnałowych takich jak błąd średniokwadratowy RMSE czy stosunek szczytowego sygnału do szumu PSNR stosowane są miary oparte na odległości pomiędzy barwami oryginału i skwantowanego obrazu w percepcyjnie równomiernej przestrzeni barw np. CIELAB. Najczęściej stosuje się poniższe wzory oparte na różnicy barw w przestrzeni RGB:

$$RMSE = \sqrt{\frac{1}{3MN}\sum_{j}^{M}\sum_{i}^{N}\left(R_{ij} - R_{ij}^{*}\right)^2 + \left(G_{ij} - G_{ij}^{*}\right)^2 + \left(B_{ij} - B_{ij}^{*}\right)^2}$$

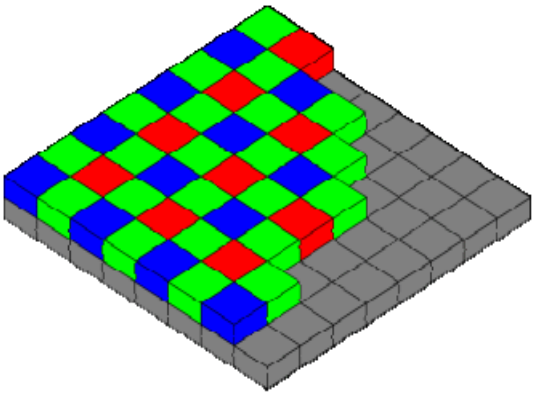
gdzie:

R_{ij}, G_{ij}, B_{ij} są składowymi barwy obrazu oryginalnego $R_{ij}^{*}, G_{ij}^{*}, B_{ij}^{*}$ są składowymi barwy obrazu powstałego w wyniku kwantyzacji.

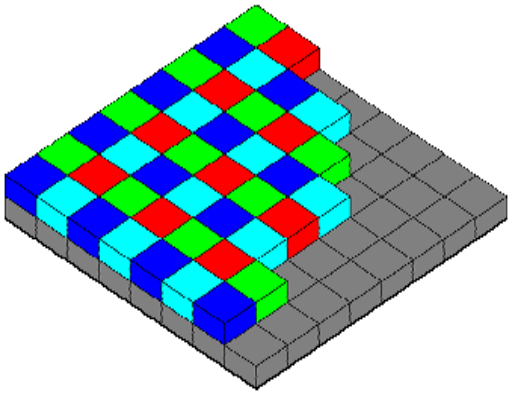
Wzór wyraża różnicę barw pomiędzy pikselami porównywanych obrazów odniesioną do liczby pikseli. Porównywanie obrazów na zasadzie piksel do piksela nie uwzględnia jednak wpływu sąsiedztwa piksela na postrzeganą barwę. Na bazie błędu RMSE często buduje się miarę logarytmiczną PSNR i wyraża ją w decybelach [17].

1.4.4 Akwizycja obrazu cyfrowego

Akwizycja obrazu jest procesem przetwarzania informacji o otaczającym świecie lub nie-elektronicznej reprezentacji obrazu (fotografie, plany, mapy dokumenty papierowe) na postać cyfrową, dogodną do obróbki. Do akwizycji obrazu mogą zostać wykorzystane różne urządzenia techniczne takie jak kamery CCD, cyfrowe aparaty fotograficzne, skanery, programy graficzne. W konkretnych przypadkach wyboru odpowiedniej metody akwizycji obrazu dokonuje się w zależności od założonych parametrów reprezentacji obrazu i przeznaczenia systemu.

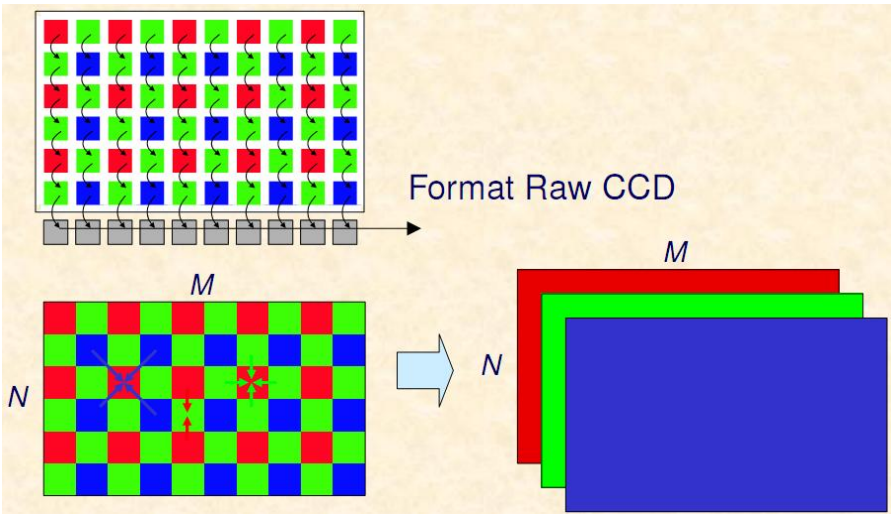


Rys. 28. Kamera CCD z maską Bayera.



Rys. 29. Kamera 3CCD.

Każdy układ akwizycji obrazów, zarówno ludzki układ wzrokowy jak również maszynowy układ wizyjny wykorzystujący kamery, dokonuje pewnej transformacji rzeczywistej przestrzeni trójwymiarowej do przestrzeni własnej, np. dwuwymiarowej płaszczyzny obrazowej. Poznanie oraz wyznaczenie parametrów tej transformacji jest fundamentalne dla dalszych prób jego modelowania czy też udoskonalania.



Rys. 30. Sposób wyznaczania obrazu RGB przy zastosowaniu interpolacji kolorów składowych.

Zadaniem przetwornika optoelektrycznego jest zamiana rozkładu energii ***f(x,y)*** na sygnał elektryczny, proces ten nazywany jest **wybieraniem obrazu** [16].

Podstawowe sposoby zamiany obrazu optycznego na sygnał elektryczny to:

- Bez akumulacji fotoładunków (np. skaner optyczny);
- Z akumulacją fotoładunków (np. matryca CCD).

Na jakość wyjściowego obrazu w rzeczywistych systemach akwizycji obrazu wpływ mają takie zjawiska jak:

- ✓ Ograniczona dynamika systemu
- ✓ Rozdzielczość elementu CCD i zjawisko aliasingu
- ✓ Szum
- ✓ Nasycenie sygnałów
- ✓ Rozmazywanie (ang. blooming).
- ✓ Sposób oświetlenia sceny

Wyżej opisane, niepożądane zjawiska powstające w rzeczywistych układach akwizycji obrazów są szczególnie dotkliwe podczas akwizycji obrazów kolorowych, gdyż mogą występować w różny sposób dla każdej z trzech składowych koloru sygnału wejściowego. W przypadku procesu znajdowania odpowiedników w obrazach stereoskopowych zjawiska te mogą prowadzić do znacznego zwiększenia tzw. fałszywych dopasowań, niemożliwych do dalszego wyeliminowania ze względu na złą jakość obrazu. Dlatego też dobranie właściwych parametrów toru do obserwowanej sceny podczas procesu akwizycji obrazu jest niezwykle istotne dla dalszego procesu przetwarzania [2].

1.5 Wstępne przetwarzanie obrazów

Głównym celem wstępnego przetwarzania obrazu jest polepszenie jego jakości, czyli filtracja [18]. W algorytmach obróbki obrazów można wyróżnić kilka poziomów przetwarzania i analizy. Najczęściej przyjmuje się trzy poziomy:

- ✓ Obróbka wstępna sygnału wizyjnego. Ma na celu eliminację zakłóceń, wydobywanie obiektu z tła, detekcję krawędzi, ustalanie poziomów szarości obiektu na podstawie histogramu, równoważenie histogramu, itp.
- ✓ Średni poziom analizy obrazów dokonuje segmentacji obrazu, lokalizacji obiektów, rozpoznania kształtu obiektu i wyróżnienia cech charakterystycznych tego kształtu.
- ✓ Analiza złożonej sceny w sensie analizy ruchu obiektu, bieżącego sterowania obiektem, zadawanie parametrów do obróbki i analizy obrazów na pozostałe poziomy.

Rys. 31. Podział algorytmów przetwarzania obrazów.

1.5.1 Przekształcenia geometryczne

Przekształcenia geometryczne pełnią rolę pomocniczą w trakcie przetwarzania obrazu [19]. Przekształcenia geometryczne najczęściej wykorzystywane są do korekcji błędów geometrii obrazu, zniekształcenia poduszkowe, beczkowe i trapezowe. Źródłem takich zniekształceń jest najczęściej niskiej jakości układy optyczne stosowane w kamerach video. Przekształcenia geometryczne można też wykorzystywać do niwelowania niedostatków systemu akwizycji obrazu [19].

Przekształcenia geometryczne mogą występować, jako samodzielne transformacje, ale także można je wykorzystywać do wspomagania innych rodzajów przekształceń i analiz. Przykładem takiej pomocniczej roli przekształceń geometrycznych jest obracanie obrazu w trakcie wyliczania średnic Fereta [19].

1.5.2 Przekształcenia punktowe

Przekształcanie punktowe polega na wykonywaniu operacji na poszczególnych pojedynczych punktach obrazu źródłowego, otrzymując w efekcie pojedyncze punkty obrazu wynikowego. Modyfikowany jest jeden punkt bez uwzględnienia elementów sąsiadujących. Operacje te mają za zadanie jedynie lepsze uwidocznienie pewnych treści już zawartych w obrazie pozostawiając niezmienione relacje geometryczne. Nie wprowadzają one żadnych nowych informacji do obrazu. Bezpośrednio widocznym efektem przekształceń punktowych jest więc zawsze zmiana skali jasności obrazu bez zmiany geometrii widocznych na obrazie obiektów.

Przekształcenia punktowe bardzo radykalnie modyfikują subiektywne wrażenia, jakie uzyskujemy oglądając obraz. Czasem prowadzi to do krańcowego zniekształcenia obrazu, czasem jednak pozwala wykryć lub uwypuklić pewne cechy obrazu praktycznie niewidoczne, gdy się ogląda obraz oryginalny.

1.5.2.1 Przekształcenia punktowe z użyciem LUT

Tablica LUT realizuje przekształcenia obrazu za pomocą wcześniej przygotowanych tabel przekodowań [9]. Możliwość przygotowania tabeli przekodowania wynika z faktu, że przy ograniczonej i dyskretnej skali jasności obrazu dla każdego piksela obrazu źródłowego ulokowanego w punkcie $(m,n)(m \in [0, M-1]),\ n \in [0, N-1]$ zachodzi warunek:

$$L(m,n) \in N$$

gdzie *N* oznacza zbiór liczb całkowitych z przedziału *[0, 2^B-1]*, a *B* jest przyjętą liczbą bitów dla reprezentacji jednego punktu obrazu [19].

Zbiór *N* zawiera skończoną i na ogół niewielką liczbę wartości, można więc dla każdej z tych wartości $x \in N$ z góry obliczyć wartość funkcji $\Psi(X)$ a następnie zbudować tabelę, w której zestawione będą wartości funkcji $\Psi(X)$ dla wszystkich wartości *X* należących do przedziału *[0, 2^B-1]* [19].

Tabela 1. Struktura tablicy przekodowania obrazu, wykorzystywanej w operacji LUT

Stara wartość	Nowa wartość
00000000	X1
00000001	X2
…	…
11111111	X256

1.5.2.2 Wyrównanie histogramu

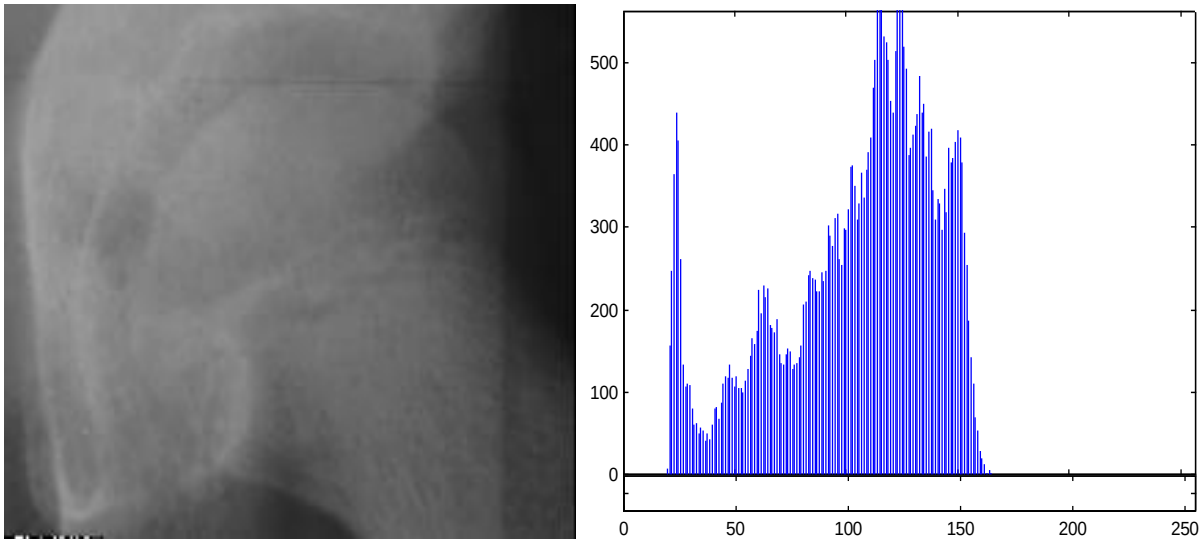
Histogram przedstawia rozkład częstości występowania w obrazie cyfrowym poszczególnych poziomów jasności. Wyznaczany jest często jako suma wszystkich pikseli o danej wartości znormalizowany przez liczbę wszystkich pikseli. Histogram ukazuje nam ile i jakie wartości pikseli występują w obrazie, daje nam jednocześnie informacje o liczbie kolorów, których można użyć do przedstawienia danego obrazu [9].

Histogram to wykres słupkowy, którym wysokość słupka określa sumaryczną liczbę punktów w obrazie, które mają zadany stopień szarości l_k , natomiast liczba słupków jest równa liczbie możliwych poziomów szarości w obrazie. Histogram jako funkcje można określić:

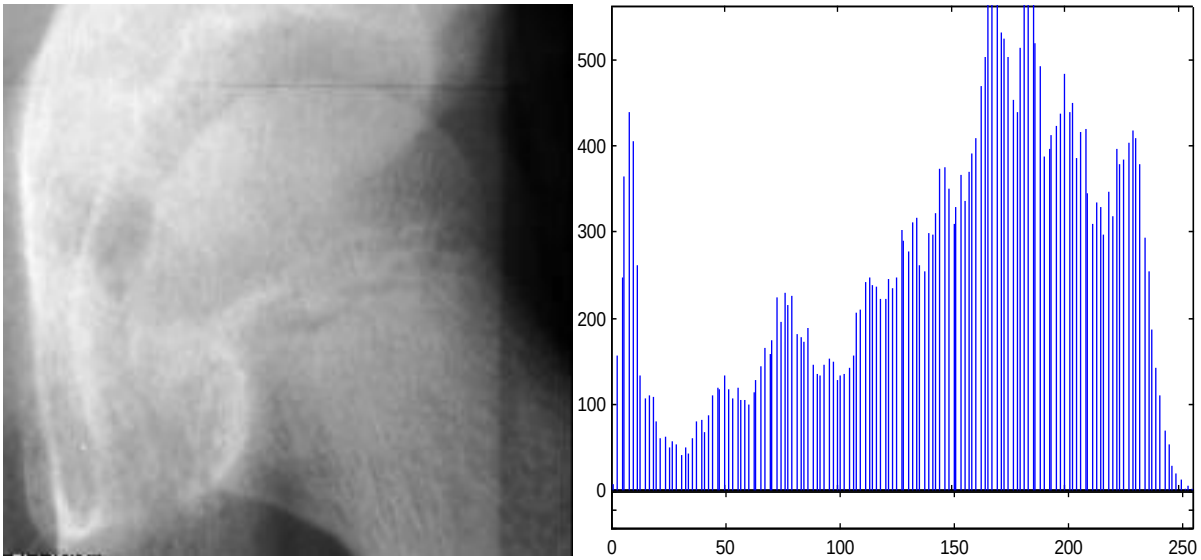
$$h(i) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} p\left(\frac{i}{m,n}\right) \quad i = 0,1,\dots,2^B - 1$$

gdzie:

$$p\left(\frac{i}{m,n}\right) = \begin{cases} 1; & gdy\ L(m,n) = i \\ 0; & w\ przeciwnym\ przypadku \end{cases} \quad [19]$$



Rys. 32. Obraz i jego histogram.



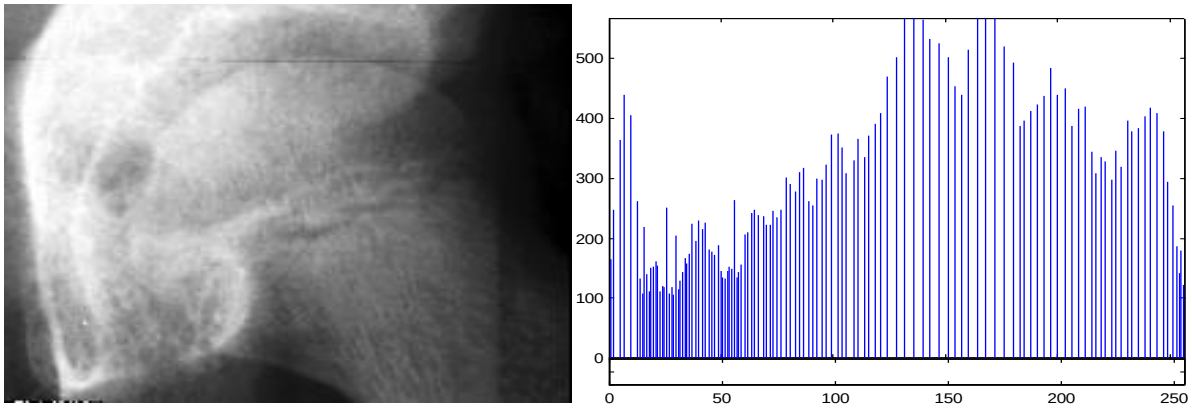
Rys. 33. Obraz po rozciągnięciu histogramu.

$$Histogram = \begin{bmatrix} His(0) \\ His(1) \\ . \\ . \\ His(I_{\max} - 1) \end{bmatrix}$$

gdzie: $His(i)$ - liczba punktów o jasnościach $i = 0...I_{\max} - 1$

$m(i) = His_{sk}(i) \cdot (I_{\max} - 1)$ oznacza nową jasność rozpatrywanego punktu obrazu po korekcji histogramu

$$His_{sk}(i) = \frac{\sum_{k=0}^{i} His(k)}{M \cdot N}$$



Rys. 34. Obraz i jego histogram po wyrównaniu.

Analizując histogram można uzyskać wiele użytecznych informacji na temat rozważanego obrazu. Na przykład można zauważyć, że wartość *h(i)* może być zerowa dla wielu wartości *i*, co oznacza, że dostępne poziomy kwantowania wykorzystane są nieefektywnie. Może to występować na obrazach, w których pewne poziomy szarości usunięto albo może to wystąpić w sytuacji, kiedy dynamika obrazu jest mała i zakres dozwolonych szarości nie jest poprawnie wykorzystany. Na ogół, jeżeli brakuje niektórych poziomów szarości w histogramie, to przyczyną są niedoskonałości przetwornika analogowo - cyfrowego i jest to zwykle sytuacja niekorzystna [19].

Operacja równoważenia histogramu polega na zmianie położenia (wzdłuż poziomej osi, odpowiadającej stopniom szarości poszczególnych pikseli) kolejnych słupków, zawierających zliczenia liczby pikseli o danej szarości. Kryterium, jakie jest stosowane przy tym przesuwaniu jest następujące: jeśli założymy, że dla pewnych liczb całkowitych m i n należących do dziedziny funkcji h(i) spełniony jest warunek $h(m) > 0$ i $h(n) > 0$ i równocześnie $h(i) = 0$ dla wszystkich $m < i < n$, to wówczas należy tak przemieszczać punkty *m* i *n*, by minimalizować wartość *Q* wyznaczaną ze wzoru:

$$Q = \frac{\sum_{i=0}^{2^B-1} h()}{2^B - 1} - \frac{h()}{n - m}$$

Przemieszczanie punktów *m* i *n* polega w istocie na tym, że zmienia się (za pomocą odpowiednich operacji typu LUT) stopnie szarości określonych punktów na obrazie.
Wyrównanie histogramu można wykonywać globalnie lub lokalnie. Lokalne wyrównanie histogramu polega na przeprowadzaniu takiej operacji w każdym z fragmentów obrazu niezależnie. Operacja ta niweluje na przykład, skutki nierównomierności oświetlenia obiektu [19].

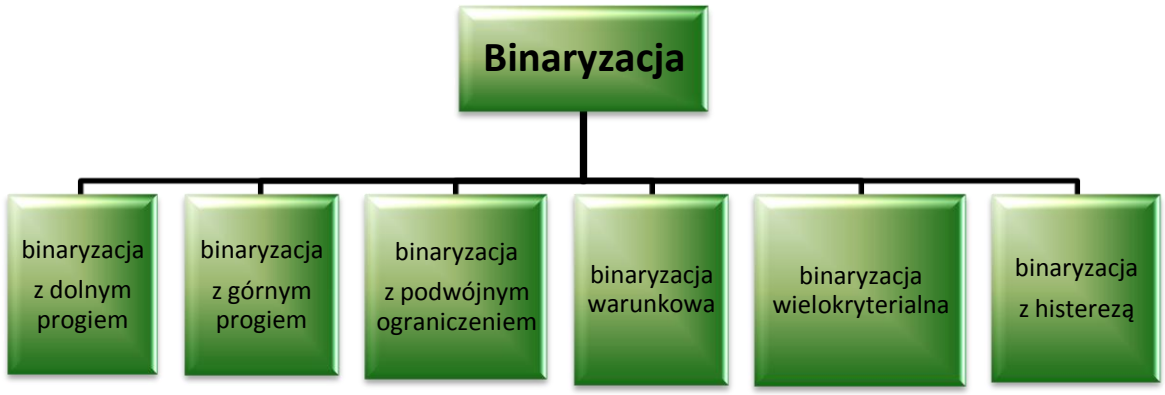
1.5.2.3 Binaryzacja obrazu cyfrowego

Binaryzacja jest jednym z podstawowych przekształceń stosowanych w analizie obrazów [22]. Jest czynnością, która praktycznie zawsze występuje w procesie obróbki obrazu w systemie wizyjnym [18]. Proces binaryzacji polega na przekształceniu obrazu mającego wiele poziomów szarości, czyli obrazów monochromatycznych i kolorowych, w obrazy czarno białe, zwane inaczej binarnymi [22]. Celem binaryzacji jest więc radykalna redukcja ilości informacji zawartej w obrazie [19]. Obrazy binarne można wykorzystać do takich działań jak:

- Wykonanie podstawowych pomiarów na obrazach
- Analizowanie i modyfikowanie kształtu obiektów monochromatycznych
- Definiowanie przekształceń obrazów monochromatycznych

Binaryzacja stanowi jedno z końcowych stadiów każdej ilościowej analizy obrazu, gdyż tylko na obrazach binarnych można przeprowadzić większość pomiarów oraz niektóre złożone przekształcenia [22].

W swojej książce [18] R. Tadeusiewicz definiuje binaryzacje jako:
Binaryzacja obrazu - zamiana obrazu ***f(x,y)***, którego piksele przyjmują wartości z przedziału $< L_{min}, L_{max} >$ na obraz ***b(x,y)***, którego piksele przyjmują wyłącznie wartości ***0*** lub ***1***.
Odmiany binaryzacji i sposoby realizacji przedstawia rysunek 35. Podczas wykonywania binaryzacji obrazu podstawowym problemem jest wybór progu binaryzacji. Najczęściej w celu znalezienia właściwej wartości progu tworzy sie histogram obrazu [22].



Rys. 35. Odmiany binaryzacji.

- ***Binaryzacja z dolnym progiem***

  Binaryzacja z dolnym progiem polega na tym, iż wszystko oprócz punktu o stopniu szarości poniżej wybranego progu staje sie czarne (0), a wszystko powyżej niego- białe.

$$L^{'}(m,n) = \begin{cases} 0; & L(m.n) \leq a \\ 1; & L(m.n) > a \end{cases}$$

  gdzie:

  $L(m,n)$-jasność w obrazie źródłowym $L(m,n) \in \{0, 2^B - 1\}$

  $L'(m,n)$-wartość odpowiedniego punktu w obrazie wynikowym

  $L'(m,n) \in \{0,1\}$

  a-próg binaryzacji

- ***Binaryzacja z górnym progiem***

  Obraz po binaryzacji z górnym progiem jest negatywem obrazu po binaryzacji z dolnym progiem (przy tej samej wartości progu)

$$L^{'}(m,n) = \begin{cases} 0; & L(m.n) \geq a \\ 1; & L(m.n) < a \end{cases}$$

- ***Binaryzacja z podwójnym ograniczeniem***

  Binaryzacja z podwójnym ograniczeniem polega na tym, iż punkty o stopniu szarości pomiędzy progami są białe, a wszystkie pozostałe punkty stają się czarne [22].

$$L^{'}(m,n) = \begin{cases} 0; & L(m,n) \leq a_1 \\ 1; & a_1 < L(m,n) \leq a_2 \\ 0; & L(m,n) > a_2 \end{cases}$$

  gdzie:

  a_1, a_2- progi binaryzacji; $a_1 < a_2$;

- ***Binaryzacja wielokryterialna***

  Binaryzacja przeprowadzana jest niezależnie na wielu obszarach obrazu, znacznie różniących się poziomem szarości [19]. Binaryzację wielokryterialną można traktować, jako kilkukrotne złożenie operacji z dwoma progami [22].

$$L^{'}(m,n) = \begin{cases} 0; & L(m,n) \leq a_1 \\ 1; & a_1 < L(m,n) \leq a_2 \\ 0; & a_2 < L(m,n) \leq a_3 \\ 1; & a_3 < L(m,n) \leq a_4 \\ 0; & L(m,n) > a_4 \end{cases}$$

gdzie:

$a_1,\ a_2,\ a_3, a_4$- progi binaryzacji; $a_1 < a_2 < a_3 < a_4$

Binaryzację wielokryterialną przeprowadza się w celu połączenia i wydzielenia obszarów obrazu znacznie różniących się poziomem jasności [22].

- ***Binaryzacja warunkowa***

Binaryzacja tego typu zwana jest często binaryzacją z histerezą. W celu jej przeprowadzenia musimy podać dwie wartości progowe ,$a_1\ i\ a_2$ przy czym $a_1 < a_2$. Każdy punkt obrazu o współrzędnych (m,n) ma swój stopień szarości lub intensywność L(m,n)

$$L'(m,n) = \begin{cases} 0; & L(m,n) \leq a_1 \\ s; & a_1 < L(m,n) \leq a_2 \\ 1; & L(m,n) > a_2 \end{cases}$$

gdzie:

s- wartość sąsiadujących punktów, $s \in \{0,1\}$;

Binaryzacja ta pozwala wydzielić z obrazu te cząstki, które mają jasne elementy. Jest to zatem narzędzie istotnie poprawiające selektywność procesu binaryzacji.

1.5.3 Przekształcenia kontekstowe

Operacje kontekstowe są to przekształcenia, w których na wartości poszczególnych pikseli obrazu wynikowego mają wpływ wartości odpowiadających im pikseli obrazu źródłowego oraz wartości punktów z ich otoczenia. Operacje te pozwalają wydobyć z obrazu wiele informacji, umożliwiają m.in. usuwanie lub ograniczenie poziomu zakłóceń nałożonych na obraz (redukcja szumu), wygładzają chropowate krawędzie, rozmywają i wygładzają obraz, ujawniają szczegóły obiektów [9].

Zależnie od zastosowanej operacji matematycznej zastosowanej na tzw. otoczeniu (oknie) grupy pikseli obrazowych filtry dzieli się na następujące rodzaje:

- Filtry liniowe
  - Filtry wygładzające,
  - Filtry wyostrzające,
- Filtry nieliniowe
  - Medianowy
  - Minimalny
  - Maksymalny

Filtracja to usuwanie szumu z obrazu. W procesie przetwarzania obrazów filtry wykorzystywane są między innymi do: poprawy złej jakości technicznej obrazu; korekcji określonych wad obrazu; wzmocnienia w obrazie pewnych elementów zgodnych z posiadanym wzorcem; stłumienia w obrazie niepożądanego szumu; rekonstrukcji poszczególnych fragmentów obrazu, które uległy częściowemu uszkodzeniu [22].

Do podstawowych operacji wykonywanych na fragmentach obrazu można zaliczyć: wygładzanie fragmentu obrazu, wyrównanie histogramu obrazu, filtracja fragmentu obrazu [22].

1.5.3.1 Filtry liniowe

Filtry liniowe obliczają nowa wartość jasności przetwarzanego piksela na podstawie określonej operacji liniowej, wykonanej na zdefiniowanym otoczeniu („oknie") pikseli. Każdy sąsiadujący punkt wnosi swój udział do obliczenia jasności nowego piksela proporcjonalny do wartości tzw. wagi. Wagi te zapisywane są w tablicy zwanej maską splotu, zaś elementy maski nazywane są współczynnikami maski [9].

- ***Filtry wykorzystujące maski***

Przy rozpatrywaniu funkcji realizujących filtry liniowe wygodnie jest się posłużyć pojęciem *konwolucji*, zwanej także *splotem funkcji* [19]. W przypadku dyskretnego splotu jądro (maska filtru) jest kwadratową tablicą zawierającą współczynniki filtra. Współczynniki mogą przyjmować zarówno dodatnie jak i ujemne wartości. Liczba wierszy i kolumn jest najczęściej nieparzysta, aby istniał element centralny. Na przykład dla prostego rozmycia maska filtru może mieć następującą postać:

1	1	1
1	**1**	1
1	1	1

Rys. 36. Przykład maski – pogrubiająca.

Dwuwymiarowa operacja splotu dla maski *h* oraz macierzy określającej obraz ma postać:

$$g(x,y) = \sum_{i=-M}^{M} \sum_{j=-M}^{M} f(x+i, y+j)h(i,j)$$

gdzie:

f(x,y)-najbliższe otoczenie analizowanego punktu, *h(i,j)-maska*

Maska (matryca) filtru to macierz o wymiarach *PxQ* (najczęściej kwadratowa o wymiarach 3x3), której element centralny „wskazuje" punkt obrazu źródłowego podlegający filtracji, a liczba wierszy i kolumn określa wielkość obszaru, jaki będzie brany pod uwagę w danym cyklu filtracji [22].

f(x-1,y-1)	***f(x,y-1)***	***f(x+1,y-1)***		***h(-1,-1)***	***h(0,-1)***	***h(1,-1)***					
f(x-1,y)	***f(x,y)***	***f(x+1,y)***	**x**	***h(-1,0)***	***h(0,0)***	***h(1,0)***	**=**		***g(x,y)***		
f(x-1,y+1)	***f(x,y+1)***	***f(x+1,y+1)***		***h(-1,1)***	***h(0,1)***	***h(1,1)***					

Rys. 37. Dwuwymiarowa operacja splotu maski i macierzy określającej obraz.

Korzystając z uproszczonego zapisu maski filtru, wartość stopnia jasności w obrazie wynikowym dla maski można obliczyć z zależności:

$$g(x,y) = f(x-1,y-1)\cdot h(-1,-1) + f(x,y-1)\cdot h(0,-1) + f(x+1,y-1)\cdot h(1,-1) + f(x-1,y)\cdot h(-1,0) +$$

$$f(x,y)\cdot h(0,0) + f(x+1,y)\cdot h(1,0) + f(x-1,y+1)\cdot h(-1,1) + f(x,y+1)\cdot h(0,1) +$$

$$f(x+1,y+1)\cdot h(1,1)$$

- ***Filtry dolno i górno przepustowy***

Filtry dolnoprzepustowe stosuje sie w celu "wygładzenia" obrazu i zredukowania szumu o wielkości piksela. Powoduje on łagodzenie gwałtownych przejść natomiast łagodne pozostawia prawie bez zmian. Obraz robi sie trochę "rozmyty". Filtry dolnoprzepustowe można wykorzystać paradoksalnie do wyostrzenia obrazu. Od obrazu oryginalnego odejmuje sie obraz rozmyty i w ten sposób można podkreślić szczegóły obrazu.

Filtry górnoprzepustowe stosuje się wtedy, gdy chcemy wzmocnić szczegóły w obrazie - nieco wyeksponować krawędzie, dokonać zmiany jasności. Obraz nabiera wyrazistości i zwiększa swoją "ostrość". Filtr ten jednak powoduje wzmocnienie szumu - czyli przypadkowych niedokładności wartości pikseli w zapisanym obrazie.

Poniżej przedstawiono tablice mnożników filtru dolnoprzepustowego i górnoprzepustowego oraz przykład ich zastosowania.

$$\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Rys. 38. Filtr dolnoprzepustowy.

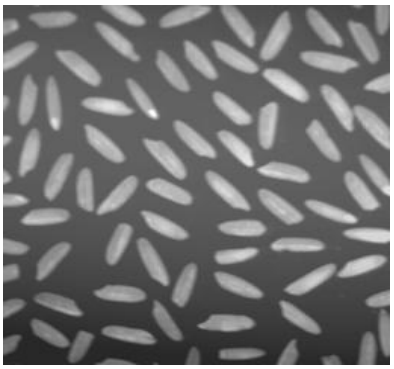
Rys. 39. Obraz oryginalny.

Rys. 40. Filtr górnoprzepustowy z użyciem pionowej maski Prewitta.

1.5.3.2 Filtry nieliniowe

- ***Filtry kombinowane***

Idea tych filtrów polega na kolejnym zastosowaniu dwóch gradientów w prostopadłych do siebie kierunkach, a następnie na dokonaniu nieliniowej kombinacji wyników tych gradientów. Dzięki nieliniowej kombinacji rezultatów liniowych transformacji obrazu tworzy się w ten sposób obraz wynikowy o wyjątkowo dobrze podkreślonych konturach [19].



Rys. 41. Obraz oryginalny.

Rys. 42. Gradient Sobela 0^o

Rys. 43. Gradient Sobela 45^o

- ***Filtry medianowe***

Filtry tego typu również pracują z użyciem kwadratowych masek, ale ich maski nie posiadają współczynników. Nowa wartość punktu jest obliczana na podstawie pewnej określonej wcześniej procedury.

Mediana jest wartością środkową w uporządkowanym rosnąco ciągu wartości jasności pikseli z całego rozważanego otoczenia przetwarzanego piksela.

Filtracja medianowa nie wprowadza do obrazu żadnego dodatkowego skalowania. Najważniejszym atutem filtracji medianowej jest to że nie powoduje ona pogorszenia ostrości krawędzi obecnych na filtrowanym obrazie poszczególnych obiektów [19].

Rys. 44. Obraz oryginalny.

Rys. 45. Filtr medianowy 5x5.

Rys. 46. Filtr medianowy 21x21.

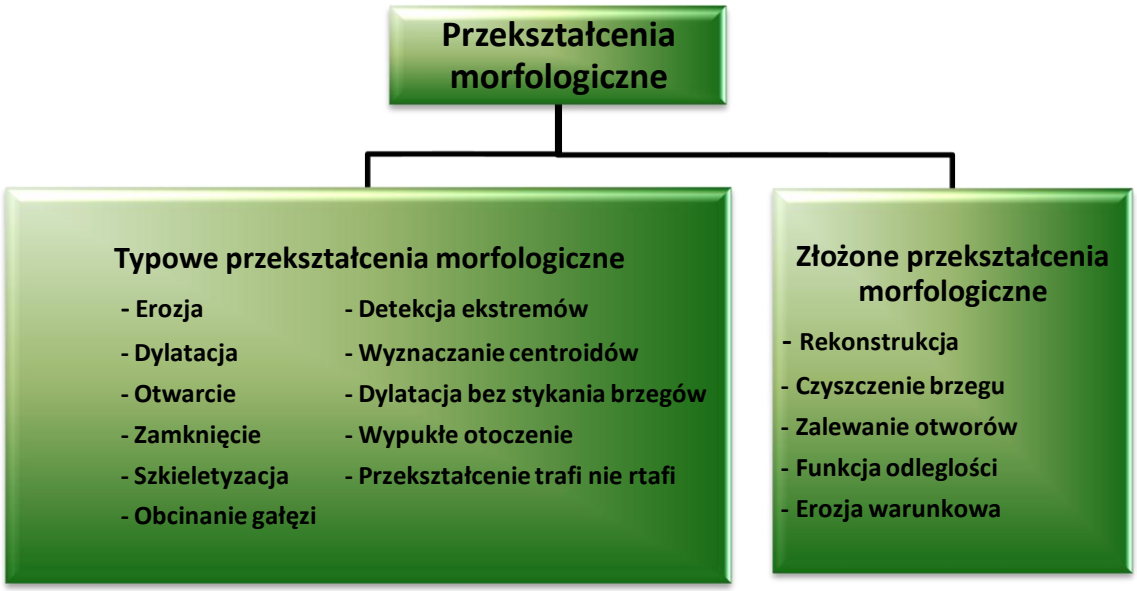
1.5.4 Przekształcenia morfologiczne

Przekształcenia morfologiczne są to przekształcenia, które powodują, że dany element obrazu jest modyfikowany tylko wtedy, gdy spełniony jest zadany warunek logiczny [9].

Fundamentalnym pojęciem przekształceń morfologicznych jest tzw. element strukturalny obrazu. Jest to pewien wycinek obrazu (przy dyskretnej reprezentacji obrazu – pewien podzbiór jego elementów) z wyróżnionym jednym punktem (tzw. punktem centralnym). Najczęściej stosowanym elementem jest koło o promieniu jednostkowym [19].

Ogólny algorytm przekształcenia morfologicznego polega na przyłożeniu centralnego punktu elementu strukturalnego kolejno do wszystkich punktów obrazu i sprawdzeniu, czy lokalna konfiguracja punktów odpowiada układowi, zapisanemu w tym elemencie, jeżeli tak, to na punkcie obrazu leżącym pod punktem centralnym elementu strukturalnego wykonuje się operację spełniającą określone warunki logiczne [9].

Istotną cechą przekształceń morfologicznych odróżniającą od innych przekształceń jest to, że przekształcenia te zmieniają tylko część punktów obrazu, których otoczenie jest zgodne z elementem strukturalnym [9].
Typowe przekształcenia morfologiczne przedstawia rysunek 47.



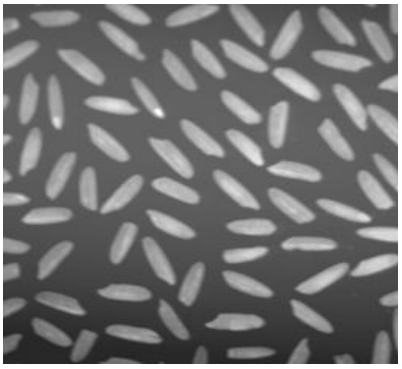
Rys. 47. Podział przekształceń morfologicznych.

Przekształcenia morfologiczne obrazu wiążą się ze zmianą stopnia szarości lub nasycenia barwy danego punktu w obrazie wynikowym w zależności zarówno od parametrów danego punktu w obrazie źródłowym jak i parametrów punktów z nim sąsiadujących [18].

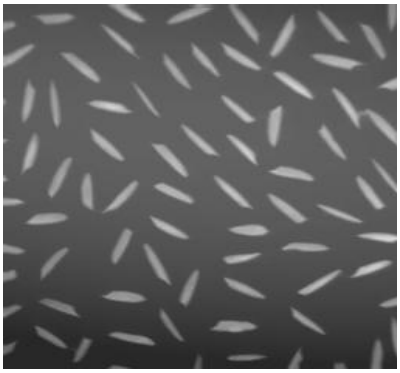
1.5.4.1 Erozja i dylatacja

Erozja jest podstawowym przekształceniem morfologicznym [22][19]. Operacja ta eliminuje drobne szczegóły w obrazie i wygładza brzeg figury. Zastosowanie liniowych elementów strukturalnych pozwala na wyszukiwanie fragmentów obrazu zorientowanych w tym samym kierunku co element strukturalny [9]. Erozja usuwa odizolowane drobne wyróżnione obszary, czyli może eliminować szumy w obrazie [9]. Bardzo często w praktyce wykorzystuje się erozje warunkową, czyli taką erozję, która przeprowadzana jest do momentu, gdy kolejna iteracja spowoduje całkowite zniknięcie obiektu. Wartość która pozostaje, jest najczęściej grupą pojedynczych pikseli będących pozostałością po obiekcie [22].

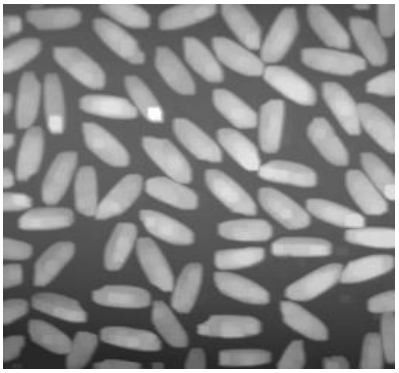
Dylatacja jest przekształceniem odwrotnym do erozji, jest addytywna. Operacja ta łączy w obiekty elementy, które są położone blisko siebie, usuwa drobne wklęsłości w wyróżnionych obszarach, wygładza ich brzegi. Długość brzegów zostaje zmniejszona, natomiast powierzchnia powiększona [9].



Rys. 48. Obraz oryginalny.



Rys. 49. Erozja.



Rys. 50. Dylatacja.

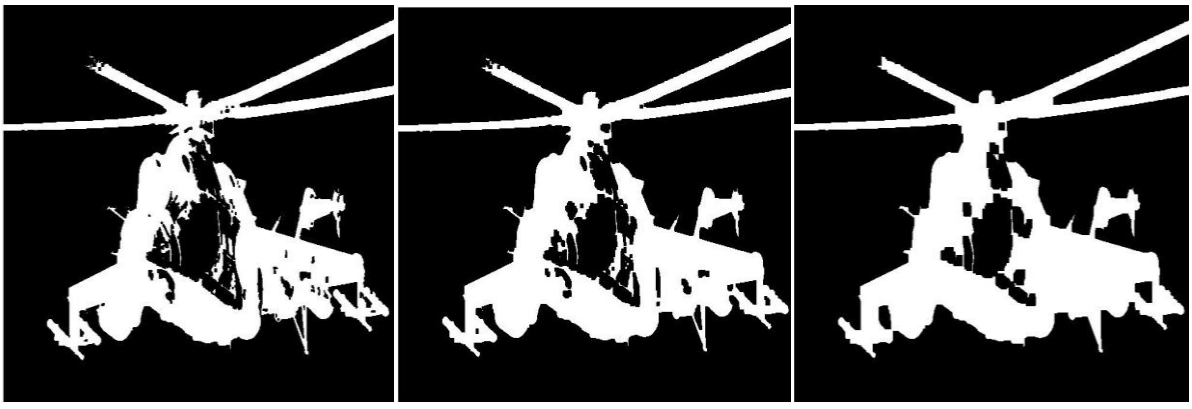
1.5.4.2 Otwarcie i zamknięcie

Przekształcenia **otwarcia** i **zamknięcia** wprowadzono w celu eliminacji zmiany pola powierzchni przekształcanych obszarów przez erozję i dylatację. Erozja zmniejsza pole powierzchni a dylatacja zwiększa.

Operacja otwarcia to kolejno wykonane operacje erozji i dylatacji [22]. Otwarcie nie zmienia wymiarów i kształtów dużych figur o wygładzonych brzegach. W wyniku tego przekształcenia usuwane są drobne obiekty i drobne szczegóły a obiekty z przewężeniami mogą być rozłączone. Wynik otwarcia jest ostateczny i powtórzenie tej operacji nie zmieni wyniku [9].

Rys. 51. Przykłady operacji otwarcia kolejno z progami 5/5, 15/15, 20/20.

Operacja zamknięcia to kolejno wykonane operacje dylatacji i erozji.
Zamknięcie wypełnia wąskie zatoki i wcięcia orz drobne otwory wewnątrz obiektu, nie zmieniając wielkości jego zasadniczej części, może też połączyć leżące blisko siebie obiekty [9].



Rys. 52. Przykłady operacji zamknięcia kolejno z progami 5/5, 15/15, 20/20.

1.5.4.3 Pogrubianie

Pogrubianie obiektu X przy użyciu elementu strukturalnego B polega na przyłożeniu tego elementu do każdego punktu obrazu w ten sposób, że punkt centralny pokrywa się z analizowanym punktem i podjęciu jego z dwóch decyzji:

- Nie zmieniać punktu, gdy element nie pokrywa się z jego sąsiedztwem,
- Zamienić wartość punktu na 1, jeżeli element strukturalny pasuje do sąsiedztwa analizowanego punktu obrazu.

Operacja ta jest przeprowadzana wielokrotnie, aż do braku zmian wprowadzanych przez operację, najczęściej element strukturalny jest przekręcany pomiędzy kolejnymi operacjami.

Pogrubienie jest przekształceniem odwrotnym do ścieniania. Dokładniej ścienianie zbiory X element B jest równoważne dopełnianiu zbioru powstałego w wyniku pogrubiania dopełnienia zbiory X elementem dopełniającym B.

Operację pogrubienia dla izolowanych punktów można wykonać posługując się elementem przedstawionym na rys.50 element strukturalny wykorzystywany przy pogrubieniu.

x	1	X
0	0	0
0	0	0

0	0	x
0	0	1
0	0	x

0	0	0
0	0	0
x	1	X

X	0	0
1	0	0
X	0	0

Jeżeli punkt centralny i otoczenie rotującego elementu strukturalnego zgadza się z lokalną konfiguracją punktów obrazu to nowa wartość punktu centralnego obrazu przyjmuje wartość 1. W przeciwnym przypadku punkt centralny nie ulega zmianie.

1.5.4.4 Detekcja ekstremów

Operacja ta służy wyodrębnianiu lokalnych ekstremów (minima i maksima) z obrazów. Do tego celu można wykorzystać operacje otwarcia i zamknięcia.

Aby wyszukać lokalne maksima należy od wyniku otwarcia danego obrazu odjąć obraz wyjściowy, a następnie dokonać binaryzacji z dolnym progiem otrzymanej różnicy:

$$M(f) = B(O(f) - f)$$

gdzie:

B(f)- binaryzacja z dolnym progiem obrazu *f;*

O(f)-operacja otwarcia;

C(f)-operacja zamknięcia.

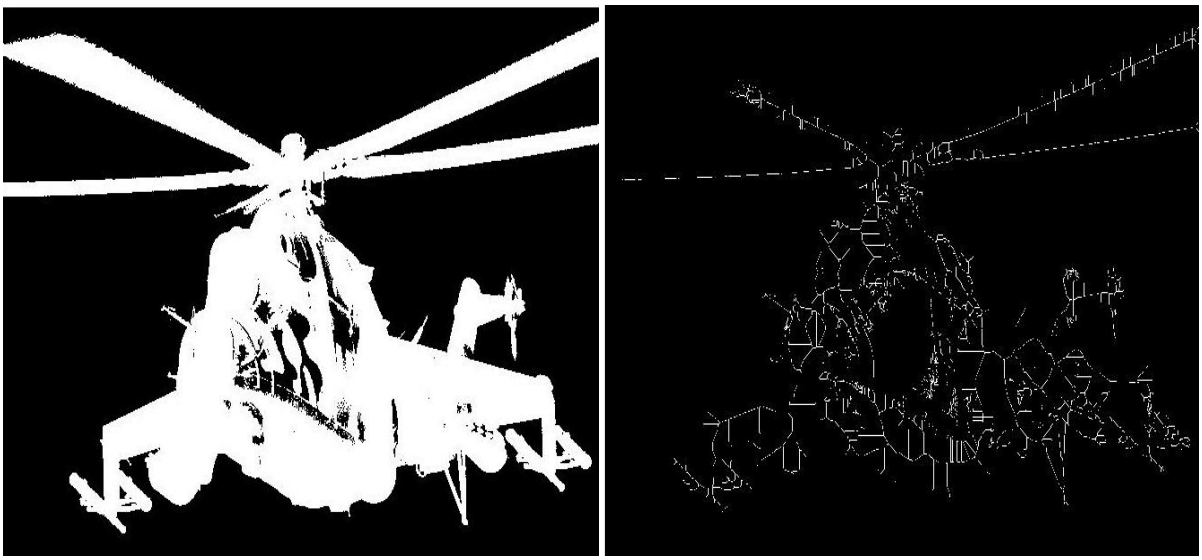
Aby wyszukać lokalne minima należy od wyniku zamknięcia danego obrazu odjąć obraz wyjściowy, a następnie dokonać binaryzacji z dolnym progiem otrzymanej różnicy:

$$m(f) = B(C(f) - f)$$

Efekt tych operacji zależy w dużym stopniu od przyjętych parametrów: wielkości otwarcia (lub odpowiednio zamknięcia) i progu binaryzacji [19].

1.5.4.5 Szkieletyzacja

Szkieletyzacja to zbiór punktów, które są w równych odległościach, od co najmniej dwóch punktów należących do brzegu rozpatrywanego obiektu. Proces ten jest wykorzystywany do wyznaczenia szkieletów (osiowych punktów) analizowanych figur. Szkielet figury jest zbiorem wszystkich punktów należących do brzegu [19]. Szkieletyzacja pozwala wyodrębnić szkielety figur (punkty osiowe) w danym obrazie [9].



Rys. 53. Przykład obrazu binarnego i jego szkieletu.

Dzięki wyodrębnianiu figur szkieletów można miedzy innymi:

- Przeprowadzić klasyfikację obiektów na podstawie ich kształtu;
- Określić orientację podłużnych obiektów;
- Rozdzielić posklejane figury;
- Wyznaczyć linie środkowe szerszych linii;
- Zasymulować proces rozrostu figur oraz utworzyć struktury ziarniste.

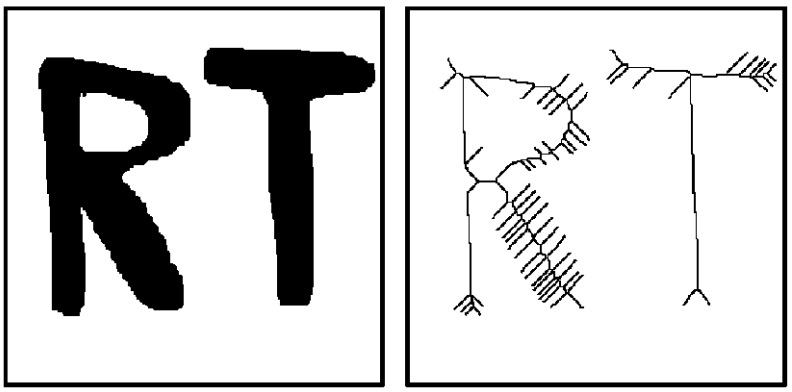
Szkielet figury jest znacznie mniejszy od niej, a w pełni odzwierciedla jej podstawowe topologiczne własności [19]. Szkieletyzacja może być realizowana jako ścienianie z elementem strukturalnym pokazanym poniżej:

$$\begin{bmatrix} X & 0 & X \\ X & 1 & X \\ 1 & 1 & 1 \end{bmatrix}$$

Najczęściej są jednak wykorzystywane inne elementy strukturalne stosowane, to znaczy:

$$\begin{bmatrix} 1 & 1 & 1 \\ X & 1 & X \\ 0 & 0 & 0 \end{bmatrix} i \begin{bmatrix} 1 & X & 1 \\ X & 1 & X \\ 0 & 0 & 0 \end{bmatrix} lub \begin{bmatrix} 1 & 1 & X \\ 1 & 1 & 0 \\ 1 & 1 & X \end{bmatrix}$$

Z definicji wynika, że szkieletem koła powinien być punkt, a szkieletem trójkąta jego dwusieczne, ale w praktyce trudno jest otrzymać takie „ teoretyczne” przypadki [22]. Niestety, proces szkieletyzacji może wprowadzać do obrazu pewne „artefakty” w postaci bocznego „gałązkowania” linii szkieletu. Efekt ten staje się szczególnie kłopotliwy, jeśli oryginalna figura posiada zakłócenia, które mogą mieć przemożny wpływ na kształt szkieletu [19]. Przykład zakłóceń spowodowanych procesem szkieletyzacji przedstawia rysunek 54.



Rys. 54. Efekt szkieletyzacji i widoczne zakłócenia w postaci licznych odgałęzień.

Operację szkieletyzacji wykorzystuje się głównie do analizy białych ciałek krwi, chromosomów, naczyń wieńcowych, klasyfikacji odcisków palców, ilościowej metalografii, pomiarów „zarysowania”(zanieczyszczenia obrazów), analizy obwodów drukowanych. Operacja ta jest wykorzystywana również w procesie wektoryzacji [9].

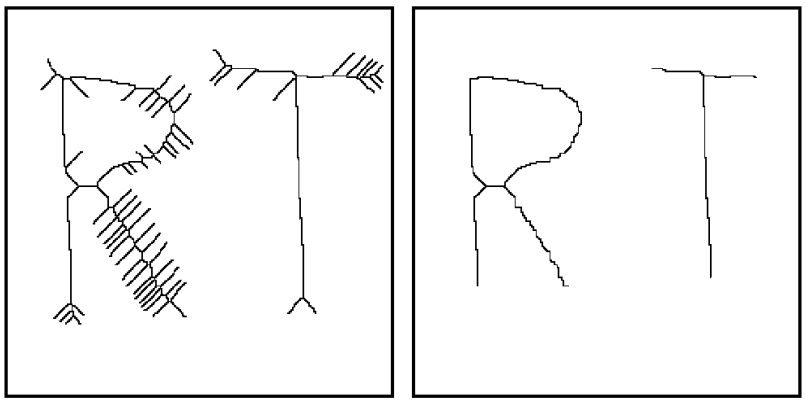
1.5.4.6 Obcinanie gałęzi

W praktycznym zastosowaniu szkielet figury w dużym stopniu zależy od regularności brzegu figury. Każda nawet niewielka, nieregularność, powoduje powstanie dodatkowego, niepotrzebnego odgałęzienia wynikowego szkieletu. Takie zbyteczne odnogi mogą powstać również w dość regularnych figurach, które nachylone są jednak pod niewłaściwym kątem do siatki.

Aby usunąć tego typu zniekształcenia można zastosować algorytm "obcinania gałęzi". Polega on na stopniowym redukowaniu odcinków posiadających wolne zakończenie. W ostateczności pozostają jedynie zamknięte pętle i odcinki przecinające brzeg obrazu. Obcinanie gałęzi po szkieletyzacji, można przeprowadzić kolejno za pomocą pary wzorców, z obracaniem o 90 stopni:

$$ES = \begin{bmatrix} X & 0 & X \\ 0 & \mathbf{(1)} & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad ES = \begin{bmatrix} 0 & X & X \\ 0 & \mathbf{(1)} & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

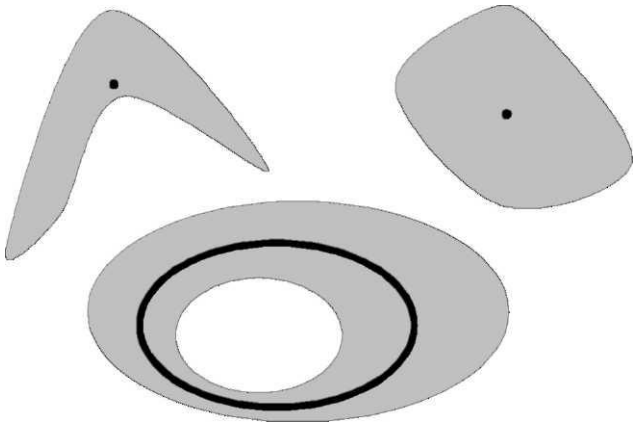
Zwykle iteracje przerywa się zanim algorytm dojdzie do punktu, w którym nie wnosiłby już żadnych zmian. Efekt algorytmu obcinania gałęzi można obejrzeć na rysunku 55.



Rys. 55. Efekt działania algorytmu obcinania gałęzi na szkielecie prostego obrazu binarnego.

1.5.4.7 Wyznaczanie centroidów

Centroid jest to szkielet figury ze szczególnie mocno obciętymi gałęziami. Operacja wyznaczania centroidu sprowadza figurę do punktu, w przypadku gdy figura nie ma „otworów", lub do pętli, gdy takowe posiada [19][22].



Rys. 56. Figury i ich centroidy.

Proces jest realizowany jako ścienianie z użyciem dwóch elementów strukturalnych:

$$\begin{bmatrix} 0 & X & X \\ 0 & 1 & 1 \\ 0 & 0 & X \end{bmatrix}, \text{ a następnie } \begin{bmatrix} X & 0 & X \\ 0 & 1 & 1 \\ X & 0 & X \end{bmatrix}$$

Kolejne iteracje wykonywane są dopóki wprowadzane są jakieś zmiany w obrazie [19].

1.5.4.8 Dylatacja bez stykania brzegów

Jednym z podstawowych problemów stojących przed komputerową analizą obrazu jest wydzielenie poszczególnych, regularnych obszarów obrazu. Zadanie to jest szczególnie trudne, gdy poszczególne obiekty stykają się nawet częściowo zachodząc na siebie. Jako pierwszy krok wykonać można wtedy erozję, która rozdzieli sklejone obszary. Powstałe jednak w ten sposób obszary mają dużo mniejszą powierzchnię, niż wynikowe. Aby powrócić do wyjściowej powierzchni nie można zastosować normalnej dylatacji, gdyż powiększane obszary połączą się ponownie. Aby temu zapobiec, trzeba zastosować specjalne przekształcenie, które powiększy powierzchnię obszarów zachowując pewien odstęp pomiędzy nimi. Przekształcenie to nazwane jest dylatacją bez stykania obszarów i może być realizowane jako pogrubienie z następującym elementem strukturalnym:

$$\begin{bmatrix} X & X & X \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Gdy opisaną wyżej metodę prowadzi się cyklicznie, aż do braku zmian w analizowanym obrazie uzyskuje się przekształcenie zwane SKIZ (ang. *skeleton by influence zone*). Skrót ten oznacza „szkielet stref wpływów”. Strefa wpływów danego punktu definiowana jest jako zbiór wszystkich punktów obrazu, dla których odległość do danego punktu jest mniejsza, niż do pozostałych. Operację tę można także wykonać stosując rotujący element strukturalny przedstawiony na rysunku 57.

X	1	X
0	0	X
0	0	0

0	X	X
0	0	1
0	X	X

0	0	0
X	0	X
X	1	X

X	X	0
1	0	0
X	X	0

Rys. 57. Rotujący element strukturalny.

Jeżeli punkt centralny i otoczenie jednego z elementów strukturalnych transformacja polegająca na uwidocznieniu na wynikowym obrazie wyłącznie niektórych poziomów szarości źródłowego obrazu – pominięciem wszystkich innych [19].

1.5.4.9 Wypukłe otoczenie

Wypukłe otoczenie figury można zdefiniować, jako najmniejsza figurę wypukłą zawierającą daną figurę. Dla figur wypukłych, ich wypukłe otoczenie jest zatem z nimi tożsame [19].

Wyznaczenie wypukłego otoczenia może być realizowane, jako pogrubianie kolejno dwoma następującymi elementami strukturalnymi:

$$\begin{bmatrix} 1 & X & X \\ 1 & X & 1 \\ 1 & 1 & X \end{bmatrix}, \text{ a następnie } \begin{bmatrix} X & 1 & X \\ 1 & X & 0 \\ X & 1 & X \end{bmatrix}$$

1.5.4.10 Przekształcenie trafi nie trafi

Przekształcenie to jest operacją wyboru, analizy, czy piksele wraz z otoczeniem mają bądź nie mają określonych właściwości geometrycznych [9].

Definicja

Do każdego punktu analizowanego obrazu przykładany jest punkt centralny danego elementu strukturalnego. Jeżeli lokalne otoczenie analizowanego punktu zgodne jest z elementem strukturalnym- odpowiedni punkt obrazu wynikowego uzyskuje wartość 1. W przeciwnym wypadku 0. Operacja ta jest przeprowadzana wielokrotni, aż do braku zmian wprowadzanych przez operację. Za pomocą tego przekształcenia można zdefiniować wszystkie inne przekształcenia morfologiczne. Przykładowo erozja za pomocą przekształcenia trafi ni trafi może być zdefiniowana następującym elementem strukturalnym:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

1.5.4.11 Rekonstrukcja

Rekonstrukcja polega na cyklicznym dokonywaniu dylatacji obrazu i wyznaczeniu części wspólnej z obrazu uzyskanego po dylatacji i obrazu wyjściowego całego przekształcenia (wykonuje się operację logiczną AND z obrazem wyjściowym a więc usuwa się te fragmenty, które po dylatacji „wyszły” poza odtwarzaną figurę). Cykl ten powtarza się, aż do uzyskania zbieżności, to znaczy braku zmian w obrazie [19].

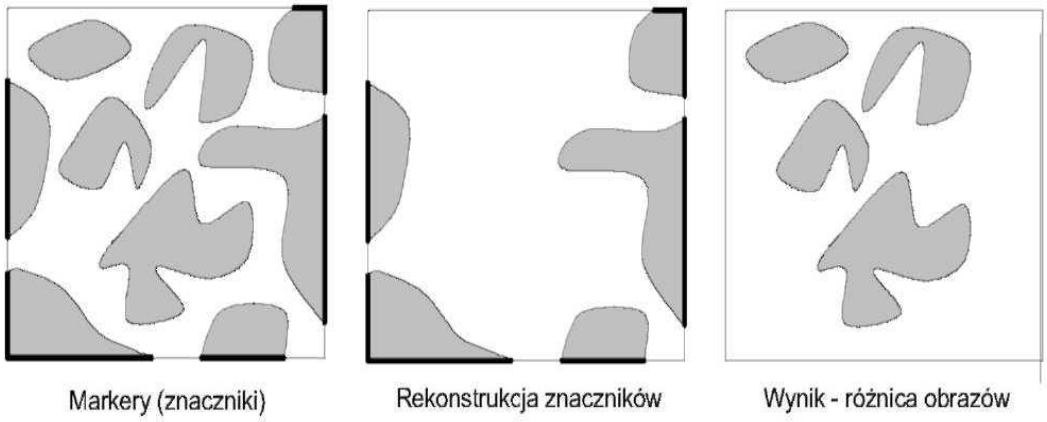
1.5.4.12 Czyszczenie brzegu

Czyszczenie brzegu ma na celu wyeliminowanie z obrazu wszystkich obszarów przecinających brzeg obrazu. Przekształcenie to często poprzedza dokonanie wnikliwszej analizy obrazu. Jest ono przydatne np. przed wykonaniem pomiarów w przypadku, gdy chcemy oceniać jedynie całkowicie widoczne obiekty. Jest to ważne i potrzebne właściwie we wszystkich dających się rozważać zastosowaniach techniki *computer vision.* Przykładowo, jeśli zadanie polega na ocenie parametrów (na przykład powierzchni i obwodu) widocznych na obrazie obiektów - to obiekty widoczne nie całkowicie nie mogą być brane pod uwagę, bo algorytm wyznaczający ich powierzchnię lub obwód z pewnością dostarczy wyników błędnych, co może znacząco wpłynąć na wyniki dalszych analiz i na wnioski z nich wypływające (na przykład, jeśli dalsze wnioskowanie uwzględnia wartość średniej powierzchni obiektu - to będzie ona zaniżona) [19].

Jak wspomniano wcześniej w przekształceniu tym wykorzystuje się rekonstrukcję. Obrazem markerów dla rekonstrukcji jest część wspólna obrazu wyjściowego i jego brzegu. Operacja przebiega w trzech etapach:

- tworzenie markerów - wspólnej części obrazu i jego brzegu,
- rekonstrukcja obiektów przeciętych przez brzeg obrazu,
- generacja różnicy obrazu wejściowego i obrazu z obiektami po rekonstrukcji [19].

Wynikiem przekształcenia jest różnica obrazu wyjściowego i wyniku rekonstrukcji [19].
Na rysunku 58 przedstawiono przykład działania algorytmu czyszczenia brzegu.



Rys. 58. Przykład działania algorytmu czyszczenia brzegu.

1.5.4.13 Zalewanie otworów

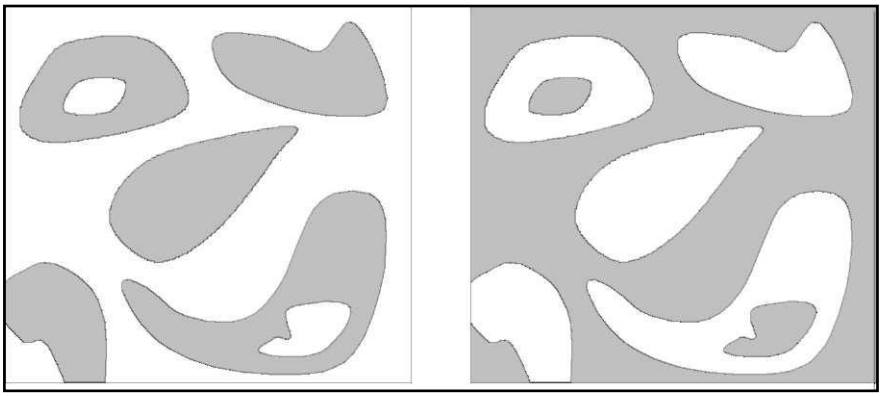
Przekształcenie polega na wypełnieniu zamkniętych obszarów w wyróżnionych obszarach w celu:

- ✓ likwidacji sztucznych otworów nieodpowiadającym rzeczywistym otworom powstałych np. na skutek odblasku światła
- ✓ wyznaczenia współczynników kształtu figury bez uwzględniania otworów
- ✓ wyznaczenia obrazu różnicowego – obrazu oryginalnego i obrazu z zalanymi otworami. Po takiej operacji otwory będą obiektami do dalszej analizy.

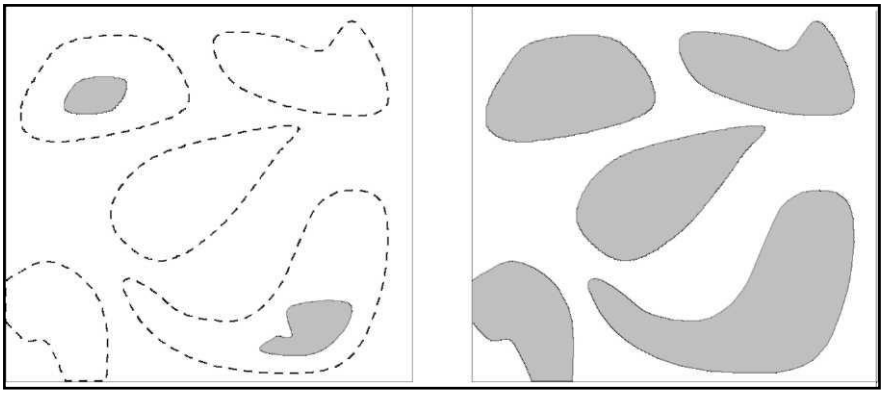
Algorytm realizujący funkcję zalewania otworów może sprowadzać się do trzech kroków:

- Wyznaczenie negatywu z obrazu wyjściowego.
- Wyczyszczenie brzegu uzyskanego negatywu (w wyniku tego pozostają na obrazie same otwory).
- Wyznaczenie sumy logicznej (operacji OR) obrazu wyjściowego i wyniku czyszczenia brzegu.

Na rysunku 59 i 60 przedstawiono przykład działania algorytmu zalewania otworów.



Rys. 59. Obraz wyjściowy oraz jego negatyw.



Rys. 60. Negatyw z wyczyszczonym brzegiem oraz obraz wynikowy.

Zagadnienia te opisane są w pozycjach [19] i [22].

1.5.4.14 Funkcja odległości

Funkcja odległości jest przekształceniem morfologicznym zdefiniowanym jedynie dla obrazów binarnych.

Definicja

Do każdego punktu analizowanego obrazu o wartości 1, odpowiadający punkt w obrazie wynikowym przyjmuje wartość równą minimalnej odległości tego punktu od brzegu figury. Pozostałe punkty uzyskują wartość 0. W praktyce funkcja odległości może być realizowana, jako suma kolejnych erozji obrazu wyjściowego [19].

1.5.4.15 Erozja warunkowa

Po wykonaniu odpowiednio dużej liczby kroków zwykła erozja doprowadza do usunięcia wszystkich cząstek. Erozja warunkowa natomiast eroduje obiekty w taki sposób, aby z każdego obiektu pozostał obraz znacznika (tj. punkt lub cząstka, która w następnym kroku erozji znika). Tak więc warunkiem w erozji warunkowej jest pozostawienie z każdego obiektu pewnego minimalnego zbioru punktów. Operacja ta jest bardzo ważnym krokiem przy rozdzielaniu sklejonych cząstek. Obraz powstały w wyniku erozji warunkowej stanowi zbiór znaczników, wykorzystywanych do rekonstrukcji granic między cząstkami. Tym samym erozja warunkowa pozwala na łatwe wyznaczenie liczby posklejanych cząstek.

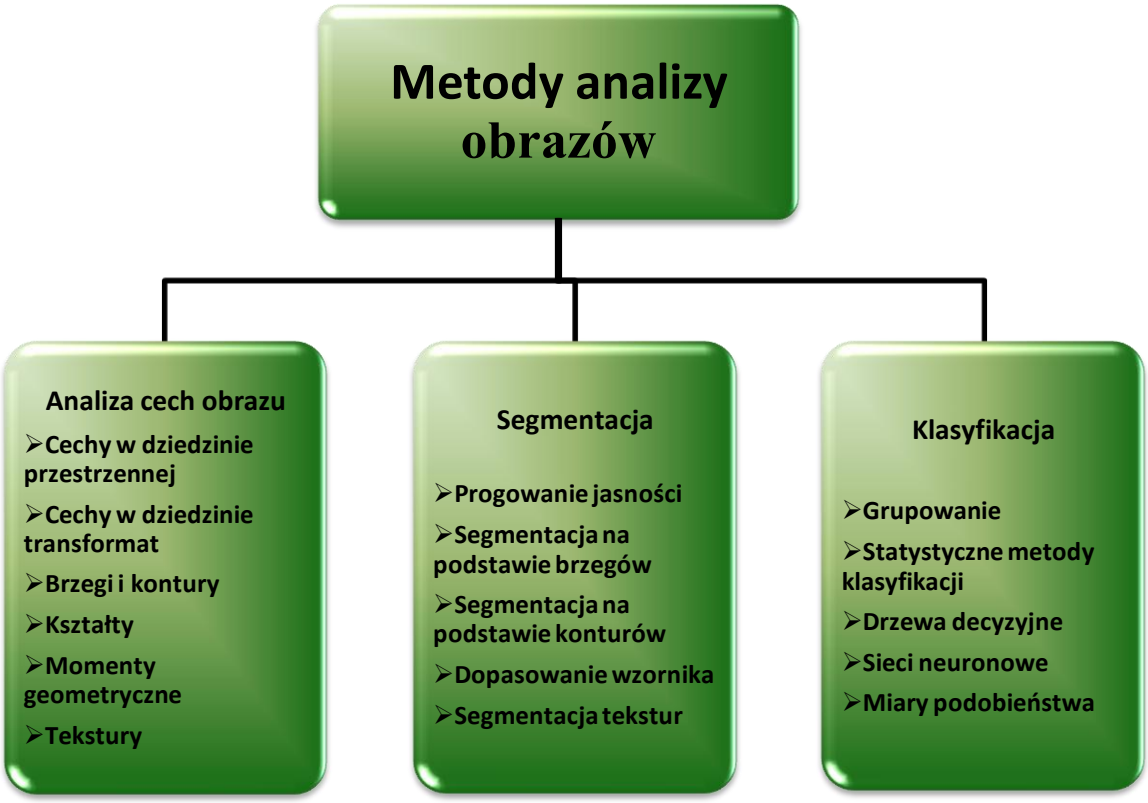
Algorytm erozji warunkowej można wykonać posługując się operacją erozji i rekonstrukcji. Etapy algorytmu erozji warunkowej są wykonywane aż do momentu, gdy kolejna operacja erozji doprowadzi do zniknięcia wszystkich obiektów. Różnice obrazów sumowane z poszczególnych etapów w efekcie dają obraz znaczników.
Erozja warunkowa jest więc pewną odmianą normalnej erozji. Przeprowadzana jest ona dla każdego obszaru aż do momentu, gdy kolejny krok usunąłby zupełnie pozostałość po obszarze. Pozostawia ona zatem pewien minimalny zbiór punktów z każdej figury. Dla przykładu erozja warunkowa pozostawia z prostokąta jedynie odcinek, a z koła – punkt [19].

1.6 Przetwarzanie i analiza obrazu cyfrowego

Po etapie doskonalenia obrazu i jego przetwarzania w większości systemów zautomatyzowanych trzeba przejść do etapu jego *analizy*. Wynikiem analizy obrazu mogą być dane jakościowe i ilościowe, opisujące określone cechy obrazu lub całej grupy obrazów [19].

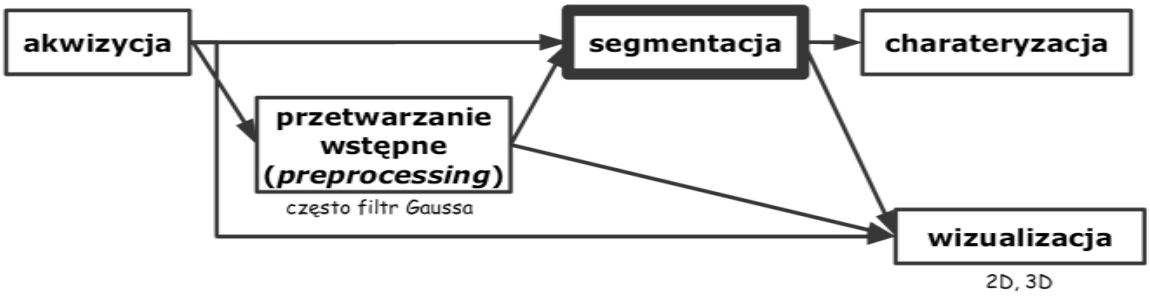
- Metody segmentacji obrazu (obraz binarny);
- Pomiar obiektów i ich kształtu (współczynniki kształtu, momenty geometryczne);
- Wymiar fraktalny;
- Szkieletyzacja;
- Operacje morfologiczne na obrazach binarnych oraz w skali szarości;

Analiza obrazu dotyczy zatem metod wyodrębniania danych (informacji) z obrazów w postaci numerycznej lub symbolicznej. Z reguły towarzyszy temu radykalna redukcja ilości informacji niezbędnych i istotnych z punktu widzenia użytkownika lub procesu. Metody analizy obrazów przedstawia poniższy schemat.



Rys. 61. Metody analizy obrazów cyfrowych.

W odróżnieniu od metod poprawy jakości obrazu, rejestracji obrazów, kodowania obrazów, wynikiem analizy obrazów jest nie obraz a dane w postaci numerycznej lub symbolicznej. Cykl przetwarzania obrazów cyfrowych przedstawia rysunek 62.



Rys. 62. Cykl przetwarzania obrazów cyfrowych.

1.6.1 Techniki segmentacji

Czynnością spinającą poziom wstępnego przetwarzania obrazu z programami analizy poszczególnych obiektów jest **proces segmentacji obrazu**. Polega on na podziale obrazu na fragmenty odpowiadające poszczególnym widocznym na obrazie obiektom [18].

Celem segmentacji jest zatem takie przetworzenie danych zawartych w obrazie, aby uzyskać taki jego podział, który pomoże w rozpoznaniu obiektów w nim zawartych i interpretacji. Ogólnie rzecz ujmując obraz jest pewnym rozkładem intensywności reprezentującym kilka obiektów bądź klas obiektów (obrazy wielo-odcieniowe), a w przypadku obrazów binarnych zestawem kilku rozdzielnych jednolitych elementów i tła [10].

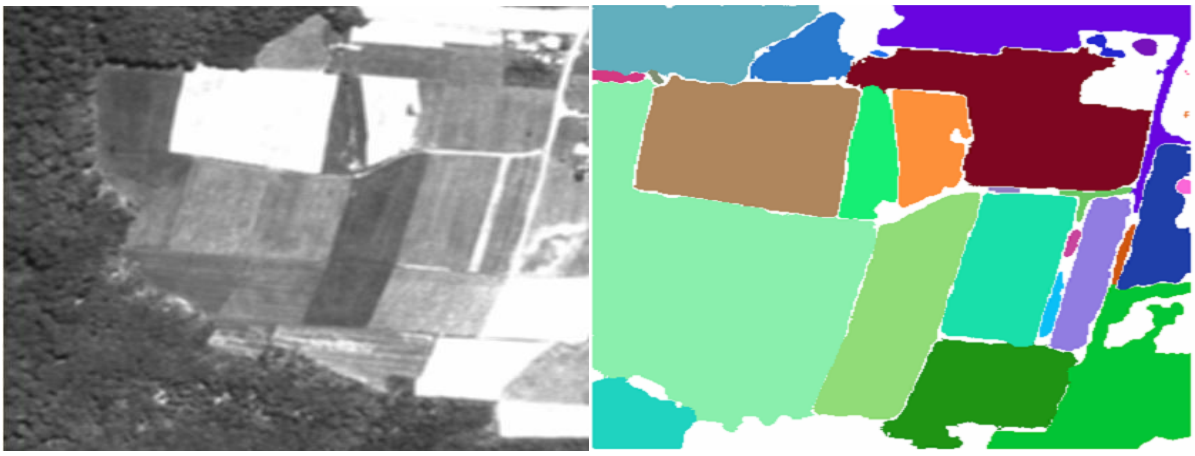
Segmentacja służy, do uproszczenia opisu obrazu poprzez zmniejszenie nadmiarowej informacji, grupowanie pikseli o zbliżonych intensywnościach lub łączenie pikseli opisujących poszczególne elementy. Formalnie w wyniku procesu segmentacji następuje podzielenie pikseli obrazu na kilka rozdzielnych klas.

Segmentacja jest więc techniką przetwarzania obrazu, umożliwiającą wydzielenie obszarów obrazu spełniających pewne kryteria jednorodności. Takimi kryteriami mogą być np. kształt obiektu, symetria obiektu, kolor itp. Najczęściej segmentacja polega bowiem na oddzieleniu poszczególnych obiektów wchodzących w skład obrazu i na wyodrębnieniu ich od tła na którym występują.

Ogólnie można wyróżnić **dwie techniki segmentacji**:

- Segmentacja przez podział obszaru (*region - splitting*)
- Segmentacja przez rozrost obszaru (*region – growing*)

Zadaniem procedur segmentacji obrazów jest jedynie podział obrazu na odpowiednie obszary a nie ich rozpoznawanie.



Rys. 63. Zdjęcie lotnicze oraz wykonanie na nim segmentacji.

1.6.1.1 Segmentacja przez podział obszaru

W metodzie tej wartość jasności każdego elementu obrazu jest porównywana z wartością progową, po czym element jest przydzielony do jednej z dwóch kategorii: o wartości progowej przekroczonej lub nieprzekroczonej. Wybory wartości progowej dokonuje się przeważnie na podstawie histogramu. Przy praktycznym stosowaniu tej metody wyłaniają się dwa problemy. Pierwszy polega na wyborze progu dyskryminacji, natomiast drugi polega na tym, że mechaniczne stosowanie metody progowania bardzo łatwo może doprowadzić do powstania całego szeregu „fałszywych obiektów" – izolowanych punktów mających wartość „1". Tworzenia izolowanych pseudo obiektów można uniknąć poprzez tzw. „określenie obszaru jednolitego". Odbywa się to poprzez kolejne dołączanie pikseli spełniających warunki progowania i będących sąsiadami jednego lub więcej pikseli należących już do obszaru. Bardziej uogólnioną metodą jest dołączanie pikseli, których atrybuty mieszczą się w określonym przedziale.

Rys. 64. Obraz przed obróbką.

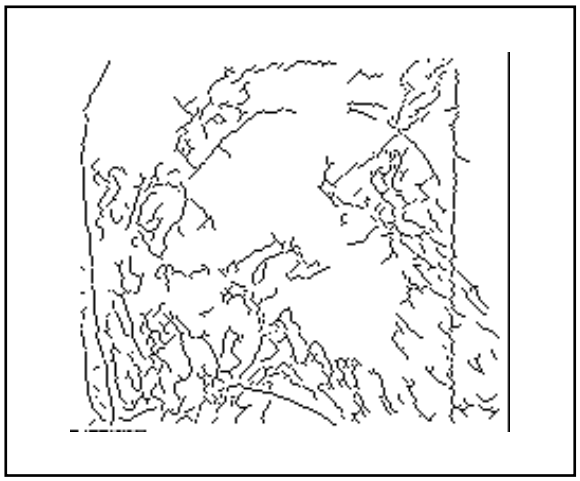
Rys. 65. Obraz po progowaniu jasności.

1.6.1.2 Segmentacja metodą wykrywania krawędzi

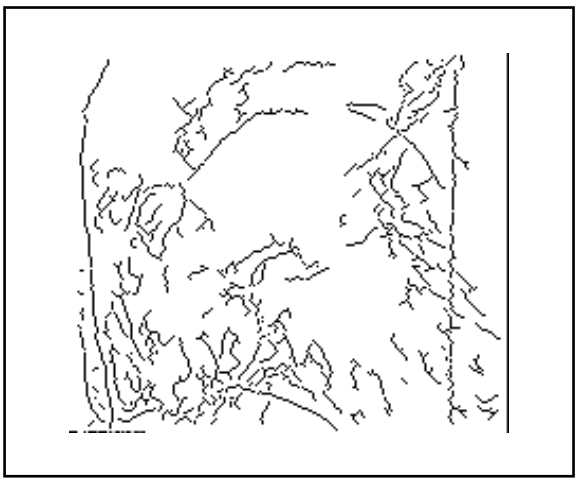
Metoda ta polega na wyszukiwaniu krawędzi pomiędzy obszarami. W tym celu jest stosowany operator gradientu, a następnie na gradiencie jest wykonywana operacja progowania. W kolejnym etapie elementy obrazu, które zostały zidentyfikowane, jako krawędzie, muszą być połączone dla uformowania krzywej zamkniętej otaczającej obszary. Techniką powszechnie stosowaną jest rozpatrywanie różnic między dwoma grupami elementów, podobnie jak w filtrach liniowych górnoprzepustowych. Aby można było uwzględnić różnice w ukierunkowaniu krawędzi trzeba zastosować więcej niż jeden filtr.

Proste detektory krawędzi są odpowiednie jedynie dla obrazów z dużym kontrastem i niskim poziomem zakłóceń o wysokich częstotliwościach.

Rys. 66. Obraz oryginalny.



Rys. 67. Krawędzie na oryginale.



Rys. 68. Krawędzie po rozciągnięciu histogramu.

Rys. 69. Krawędzie po wyrównaniu histogramu.

1.6.1.3 Segmentacja przez rozrost obszaru

Metoda rozrostu obszaru polega na poszukiwaniu grup elementów o zbliżonej jasności. Najprostsza postać tej metody to rozpoczęcie od jednego elementu i sprawdzanie czy elementy sąsiednie mają podobną jasność. Jeśli tak są one grupowane w obszar. W ten sposób powstają obszary, które rozrastają się z pojedynczych elementów obrazu. Następnie dla każdego obszaru stosuje się test jednolitości i w razie negatywnego wyniku obszar jest ponownie dzielony na mniejsze elementy. Proces ten jest powtarzany tak długo, aż wszystkie obszary są jednolite. Główną zaletą użycia małych obszarów zamiast elementów obrazu jest zmniejszenie wrażliwości na zakłócenia.

Jedno z kryteriów jednolitości jest oparte na porównaniu maksymalnej różnicy między wartością elementu obrazu $L(m,n)$ i wartością średnią dla obszaru. Dla obszaru R o wielkości $\#R$ ($\#R$ – oznacza np. liczba pikseli należących do danego obszaru) oblicza się średnią S:

$$S = \frac{1}{\#R} \sum_{m,n \in R} L(m,n)$$

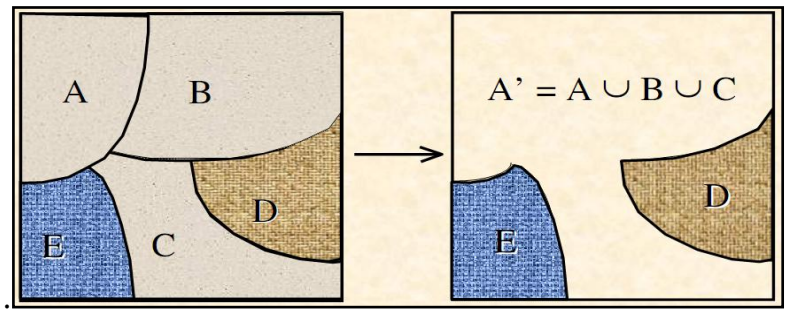
a następnie sprawdza się warunek:

$$\max_{m,n \in R} \left| L(m,n) - S \right| < T$$

Obszar jest jednolity, gdy powyższy warunek jest spełniony dla ustalonego progu T. Wybór tego progu oprzeć można na znanym fakcie: prawdopodobieństwo, że jasność elementu $L(m,n)$ różni się od jego średniej S o więcej niż pewną wielkość x jest dane następującą całką:

$$I_n = \frac{2}{\sqrt{2}\pi\sigma} \int_x^{\infty} e^{-\frac{z^2}{2\sigma^2 dz}}$$

gdzie σ jest odchyleniem standardowym zakłóceń. Ustalając próg T ustala się więc także prawdopodobieństwo tego, że do wydzielonego obszaru włączone będą obiekty nienależące do niego.



Rys. 70. Przykład segmentacji przez rozrost obszaru.

1.6.2 Segmentacja oparta na statystyce

Segmentacja tego typu określa pewne statystyczne własności obszarów, które są jednorodne lub wypełnione pewnym stałym wzorcem. W tych przypadkach nie można zastosować segmentacji opartych na uprzednio opisanych technikach, gdyż pojedyncze piksele znacznie różniąc się od siebie, tworzą pomimo to jednorodny obszar [19].

1.6.3 Segmentacja obiektów stykających się

Dla celów praktycznej segmentacji obrazów zawierających wiele stykających się obiektów używa się zwykle bardziej złożonego algorytmu, składającego się z następujących etapów:

1. Usunięcie elementów przeciętych przez brzeg obrazu.
2. Wykonanie operacji zamknięcia, a następnie operacji wypełniania otworów
3. Erozja warunkowa, w wyniku której powstaje obraz znaczników z każdego erodowanego obiektu.
4. Wykonanie jednej operacji dylatacji dla każdego zerodowanego obiektu.
5. Wykonanie operacji SKIZ aż do osiągnięcia stanu ustalonego, przy czym po każdej operacji dylatacji obraz jest mnożony logicznie z obrazem wejściowym.
6. Obcięcie gałęzi i przemnożenie logiczne obrazu po obcięciu z obrazem oryginalnym.

1.6.4 Obraz po segmentacji

Po segmentacji jest otrzymywany obraz wynikowy (np. binarny) na podstawie którego jest łatwiej wydzielać cechy/parametry obiektów wyodrębnionych w etapie segmentacji. Cechy obiektów obrazu można podzielić na kilka kategorii:

• cechy linii brzegowej obiektu (np. kształt tej linii),

• cechy pola obszaru obiektu (np. rodzaj tekstury obszaru, kolor).

• topologiczne cechy kształtu

Zależnie od celu analizy obrazu opis cech obiektu skupia się na jego obszarze (np. jego polu powierzchni) lub kształcie jego linii brzegowej (np. długości tej linii).

Obraz po segmentacji powinien mieć następujące cechy:

- Obraz powinien być jednorodny i jednolity (nie dotyczy to tekstur);
- Wnętrza obrazu powinny być proste bez wielu małych otworów;
- Obszary przylegające do siebie (graniczące ze sobą) powinny mieć inne wartości;
- Brzegi obszarów powinny być proste, nieposzarpane.

1.7 Rozpoznawanie i porównanie obrazów

W większości komputerowych systemów wizyjnych segmentacja jest ostatnim etapem przetwarzania obrazów (obiektów w obrazie). Kolejne operacje są związane procesem analizy obrazu lub/i z identyfikacją treści obserwowanych obrazów. W wyniku segmentacji otrzymuje się obszary lub kontury odpowiadające występującym w obrazie obiektom. Następnie zakłada się, że w analizowanych obrazach znajdują się jedynie obiekty określonych rodzajów i próbuje się klasyfikować (rozpoznawać) wydzielone segmenty.

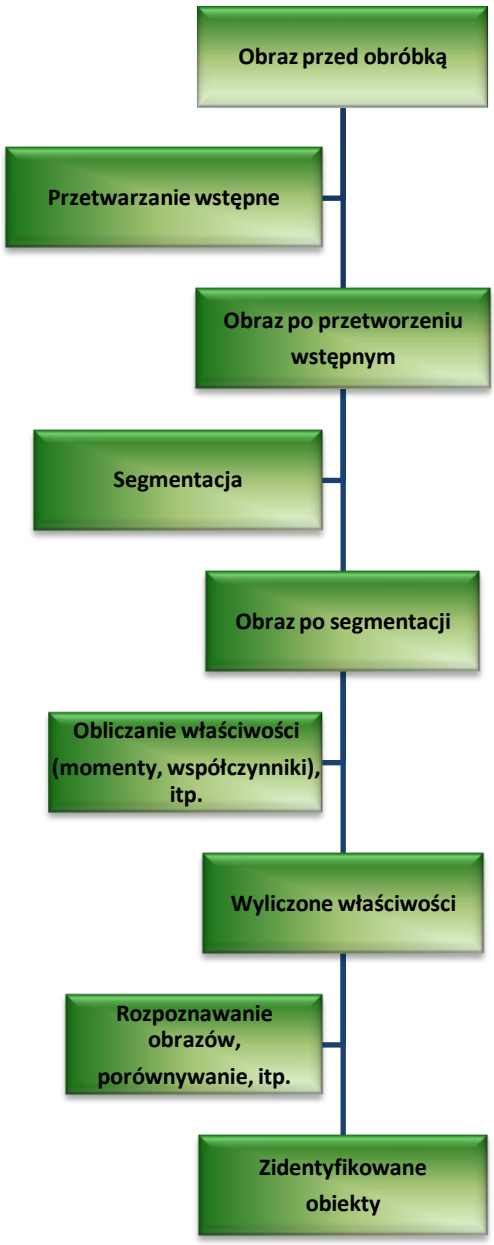
Rozpoznanie polega na przypisaniu każdego segmentu do określonej klasy, tzn. uznaje się, że dany segment wyobraża obiekt należący do tej klasy. W celu umożliwienia sklasyfikowania każdego elementu zwykle dopuszcza się też klasę "obiekty niezdefiniowane", gdyż pewne segmenty w wyniku np. nieprawidłowego wydzielenia lub należenia do tła mogą nie reprezentować żadnego obiektu. Zazwyczaj do identyfikacji obrazów stosuje się metody wykorzystujące pojęcie przestrzeni cech.

Przestrzenią cech nazywana jest pewna n-wymiarowa przestrzeń, w którą odwzorowane są rozważane obiekty, które opisane są przez pewien zbiór n-cech $X = \{x_1,...,x_n\}$.Cechami mogą być najróżniejsze własności, ale wartość ich dla każdego obiektu musi być pojedynczą liczbą (rzeczywistą lub całkowitą). Jeżeli 'obiektem' będzie np. próbka krwi to cechami mogą być: liczba leukocytów, erytrocytów, poziom hemoglobiny, itd. W przypadku dwuwymiarowych obiektów geometrycznych obserwowanych przez system wizyjny, cechami mogą być: powierzchnia, obwód, liczba otworów, lub stosunek tych wielkości itd. W teorii nie prowadzi się rozważań nad naturą obiektów i cech, ale zakłada się, że cechy są tak wybrane, by obiekt mógł zostać sklasyfikowany na podstawie jego wartości cech. Dla danego obiektu q można określić zbiór jego wartości cech zwany wektorem cech:

$$x(q) = [x_1(q), \dots, x_n(q)]$$

Wektor cech jest rezultatem odwzorowania obiektu w n-wymiarową przestrzeń cech (każdy wymiar tej przestrzeni jest, zatem związany z jedną cechą). Niech w rozważanym problemie występuje *m* klas obiektów $K_1,...,K_m$. Ponieważ identyfikacja obiektu jest realizowana na podstawie jego wektora cech, więc w przestrzeni cech powinno istnieć *m* tzw. obszarów decyzyjnych $O_1,...,O_m$ przypisanych poszczególnym klasom i tak dobranych, żeby na podstawie przynależności wektora *x(q)* do obszaru O_i można było wnioskować o przynależności obiektu *p* do klasy K_i.

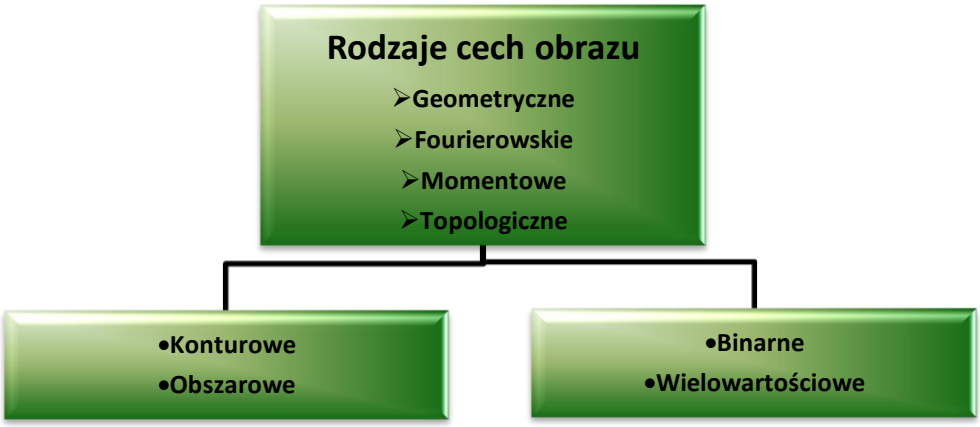
Często zdarza się, że ten sam wektor cech opisuje obiekty z różnych klas, wtedy przynależność do danej klasy ocenia się na podstawie prawdopodobieństwa przynależności obiektu do reprezentowanego przez dany wektor cech do klasy K_j i klasyfikuje się obiekt do tej klasy, dla której to prawdopodobieństwo jest największe. W efekcie oznacza to zdefiniowanie w przestrzeni cech obszarów decyzyjnych. W celu właściwej identyfikacji obiektów i odpowiedniego zakwalifikowania ich do określonych właściwych dla danego zadania technicznego klas, najważniejszym problemem staje się dobór odpowiedniej przestrzeni cech. Przede wszystkim zależy wybrać cechy najistotniejsze dla sklasyfikowania obiektów. Do tego jednak potrzebna jest znajomość różnych rodzajów cech i ich właściwości [10].



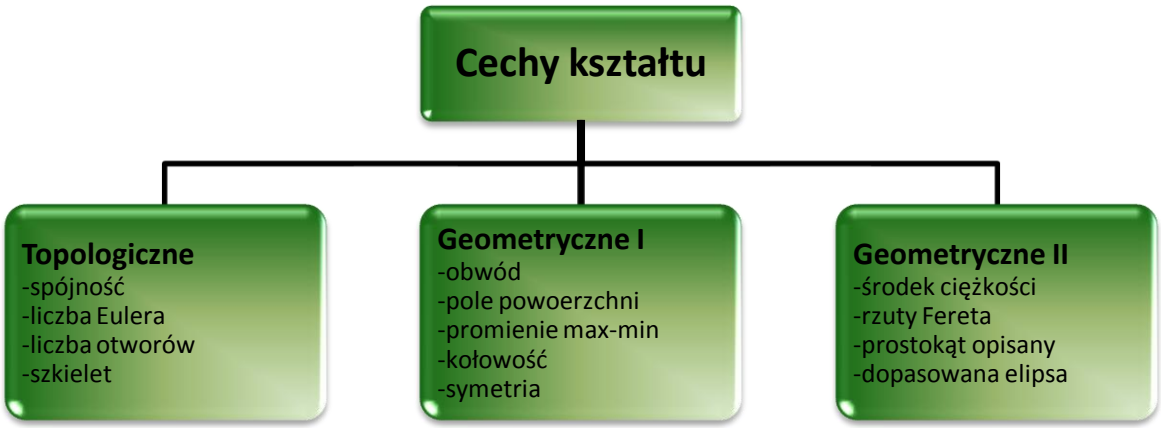
Rys. 71. Etapy przetwarzania i rozpoznawania obrazu cyfrowego.

1.8 Wyznaczanie podstawowych cech rozpoznanych obiektów

Często zdarza się, że ten sam wektor cech opisuje obiekty z różnych klas, wtedy przynależność do danej klasy ocenia się na podstawie prawdopodobieństwa przynależności obiektu do reprezentowanego przez dany wektor cech do klasy K j i klasyfikuje się obiekt do tej klasy, dla której to prawdopodobieństwo jest największe. W efekcie oznacza to zdefiniowanie w przestrzeni cech obszarów decyzyjnych. W celu właściwej identyfikacji obiektów i odpowiedniego zakwalifikowania ich do określonych właściwych dla danego zadania technicznego klas najważniejszym problemem staje się dobór odpowiedniej przestrzeni cech. Przede wszystkim zależy wybrać cechy najistotniejsze dla sklasyfikowania obiektów. Do tego jednak potrzebna jest znajomość różnych rodzajów cech i ich właściwości.



Rys. 72. Rodzaje cech obrazu.



Rys. 73. Rodzaje cech kształtu.

1.8.1 Cechy geometryczne

Cechy geometryczne dotyczą głównie opisu figur obrazów binarnych. Mogą być one wyrażone przez:

- ❖ - współczynniki kształtu,
- ❖ -współczynniki momentowe.

Cechy geometryczne mogą być wyznaczane zarówno w płaszczyźnie samego obrazu, jak i jego reprezentacji widmowej (w płaszczyźnie częstości przestrzennych). W tym drugim przypadku mówimy o cechach zwanych deskryptorami fourierowskimi.

- ➢ ***Współczynniki kształtu***

Współczynnik kształtu jest miarą podobieństwa kształtu figury do koła, gdyż dla kola współczynnik ten osiąga wartość minimalną i równą 1.

$$\gamma = \frac{L^2}{4\pi S}$$

Współczynniki kształtu powinny przede wszystkim dobrze różnicować figury o różnych kształtach. Współczynniki kształtu powinny być wrażliwe na zmienność kształtu figury, powinny jednak zachowywać niewrażliwość jeśli idzie o zmianę sposobu przedstawiania figury. Powinny też wykazywać dużą niezależność od obrotów. Ogólnie współczynniki kształtu dzielimy na dwie grupy:

- ❖ -szybkiego obliczania
- ❖ -dokładność opisu obiektów

Do pierwszej grupy należą współczynniki:

- Współczynniki cyrkularności

$$W1 = 2\sqrt{\frac{S}{\pi}} \qquad W2 = \frac{L}{\pi}$$

- Współczynnik Malinowskiej

$$W3 = \frac{L}{2\sqrt{\pi S}} - 1$$

- Współczynnik $Lp1$

$$W7 = \frac{r_{\min}}{R_{MAX}}$$

- Współczynnik $Lp2$

$$W8 = \frac{L_{MAX}}{L}$$

- Współczynnik Mz

$$W9 = \frac{2\sqrt{\pi S}}{L}$$

Do grupy drugiej zaliczamy współczynniki:

- Współczynnik Blaira-Blissa

$$W4 = \frac{S}{\sqrt{2\pi \sum_i r_i^2}}$$

- Współczynnik Danielssona

$$W5 = \frac{S^3}{\left(\sum_i l_i\right)^2}$$

- Współczynnik Harlicka

$$W6 = \sqrt{\frac{\left(\sum_i d_i\right)^2}{n\sum_i d_i^2 - 1}}$$

gdzie:

L - obwód rzutu obiektu

S - pole rzutu obiektu

d - odległość pikseli konturu obiektu od jego środka ciężkości

i - numer piksela obiektu

l_i - minimalna odległość piksela od konturu obiektu

r_i - odległość piksela obiektu od środka ciężkości obiektu

n - liczba punktów konturu

$r_{\min}$ - minimalna odległość konturu od środka ciężkości

R_{MAX} - maksymalna odległość konturu od środka ciężkości

L_{MAX} - maksymalny gabaryt obiektu

Dla każdej klasy obiektów oczekiwanych w obrazie należy drogą doświadczalną sprawdzić czy przyjęte współczynniki kształtu w wystarczający sposób różnicują analizowane obiekty.

Problemy ze współczynnikami kształtu:

- - czułe na duże zmiany skali,
- -czułe na dyskretyzację,
- - bardzo czułe na zniekształcenia związane z konfiguracją układu detekcji (perspektywa),
- zróżnicowana czułość na zmiany proporcji figur.

- ***<u>Pole obiektu</u>***

  Wyznaczenie pola powierzchni sprowadza się do zliczenia pikseli należących do interesującego nas obszaru. Dodatkowo po wycechowaniu obrazu, czyli po ustaleniu rzeczywistej odległości wyróżnionych punktów obrazu, można otrzymać wynik pomiaru w jednostkach, które nam są potrzebne.

  Cecha ta jest czuła na błędy wynikłe z niewłaściwej binaryzacji, jednak z drugiej strony jest nieczuła na przesunięcie i obrót obiektu w polu widzenia [19].

$$S = \sum_{i=1}^{n} \sum_{j=1}^{m} p(i, j)$$

$$p(i, j) = \begin{cases} 1 \ gdy \ obiekt \\ 0 \ w \ pozostalymprzypadku \end{cases}$$

- ***<u>Obwód obiektu</u>***

  Pomiar obwodu obiektu, czyli inaczej długości krawędzi obiektu jest dość trudny z uwagi na konieczność przybliżania ciągłej krzywej przez dyskretną kombinację punktów obrazu.

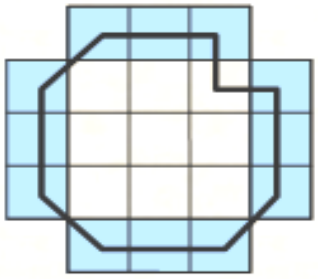
$$L = \sum_{i=1}^{n} \sum_{j=1}^{m} p(i, j)$$

$$p(i, j) = \begin{cases} 1 \ gdy \ kontur \\ 0 \ w \ pozostalymprzypadku \end{cases}$$

Obwód obiektu równy jest liczbie elementów jego konturu. Obwód figury wówczas wynosi:

$$T = \int \sqrt{x^2(t) + y^2(t)}$$

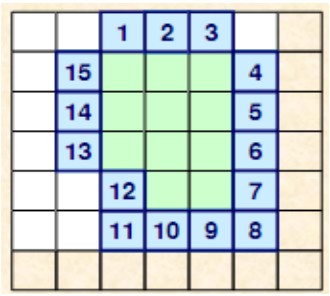
Dla siatki dyskretnej, długość brzegu figury nie jest liczbą punktów brzegowych. Przyjmuje się, że element konturu jest kwadratem o boku = 1.



Rys. 74. Reprezentacja elementów konturu.

W praktyce korzysta się z kilku sposobów pomiaru tej cechy:

- Zliczenie punktów brzegowych figury. Jest to najprostsza metoda, lecz dająca duże zafałszowania, w szczególności dla małych figur.



Rys. 75. Przykład zliczania punktów brzegowych figury.

- Zliczenie punktów brzegowych figury z uwzględnieniem ich położenia. Dla punktów sąsiadujących w pionie lub poziomie stosuje się wagę wielkości 1, a dla punktów sąsiadujących po przekątnej wagę $\sqrt{2}$

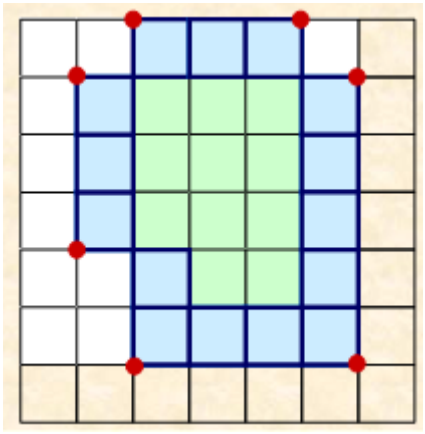
- Obwód obiektu można wyznaczyć na podstawie wzoru:

$$O = aN_B - bN_W$$

gdzie:

N_B- liczba zewnętrznych boków elementów konturu,

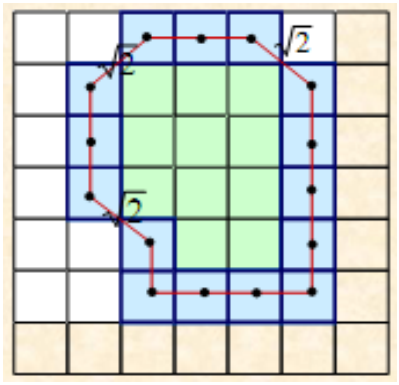
N_A- liczba wierzchołków konturu



Rys. 76. Przykład wyznaczania długości konturu.

$$a = \frac{\pi(1+\sqrt{2})}{8} \quad b = \frac{\pi}{8\sqrt{2}}$$

- Wyliczenie średniej z długości zliczonych po zewnętrznej i wewnętrznej stronie brzegu figury.Obwód obiektu równy jest sumie długości odcinków łączących środki elementów konturu [16].



Rys. 77. Przykład zliczania środków elementów konturu.

- Przybliżenie figury odpowiednim wielokątem. Teoretyczne jest to najlepsza metoda obliczania pola powierzchni, brakuje jednak jednoznacznie zdefiniowanego sposobu doboru wielokąta przybliżającego figurę.
- Przybliżanie długości brzegu liniami krzywymi. Metoda ta może wprowadzać znaczne błędy.
- Wykorzystanie formuły Croftona. Metoda ta opiera się na zasadzie Cauchy'ego

$$L = \int_0^{\pi} D(\alpha)d\alpha$$

gdzie:

L-obwód, *α*- kąt rzutu, *D(α)* –długość rzutu.

Po przekształceniach otrzymujemy formułę Croftona dla siatki kwadratowej uwzgledniając cztery podstawowe kierunki sumowania rzutów:

$$L = \frac{\pi}{4} \cdot \left[a \cdot \left(N_0 + N_{90} \right) + \frac{a}{\sqrt{2}} \cdot \left(N_{45} + N_{135} \right) \right]$$

gdzie:

N_0, N_{90}, N_{45}, N_{135} – rzuty figury dla wybranych kierunków rzutowania,

a – odległość punktów siatki.

- ***<u>Środek ciężkości</u>***

Środki ciężkości służą do określania położenia obiektu w obrazie. Ich wartości nie muszą być liczbami całkowitymi.

$$\tilde{x} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{m} k}{S} \qquad k = \begin{cases} i \;\; gdy\; obiekt \\ 0 \;\; w\; pozostalymprzypadku \end{cases}$$

$$\tilde{y} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{m} k}{S} \qquad k = \begin{cases} j \;\; gdy\; obiekt \\ 0 \;\; w\; pozostalymprzypadku \end{cases}$$

gdzie:

S - pole obiektu

L - obwód obiektu

$n \; x \; m$ - rozmiar obiektu

$\tilde{x}$ - współrzędna x środka ciężkości

$\tilde{y}$ - współrzędna y środka ciężkości

- ***<u>Zawartość</u>***- popularny współczynnik opisu kształtu niezależny od liniowych transformacji skali, rotacji) opisywana jest wzorem:

$$R_Z = \frac{L^2}{4\pi S}$$

- ***Centryczność-*** jest to stosunek długości maksymalnej cięciwy A obiektu do maksymalnej długości cięciwy B prostopadłej do A

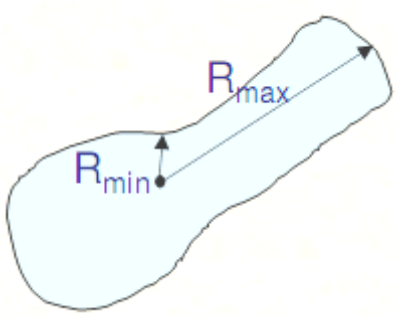
  Centryczność można także obliczyć za pomocą momentów w następujący sposób:

$$R_C = \frac{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2}{S}$$

  gdzie:

  S- pole powierzchni obiektu$\mu_{20}, \mu_{02}, \mu_{11}$- momenty geometryczne

- ***Promienie***



Rys. 78. Przykład zastosowania długości promieni figury.

  Promienie R_{min}, R_{max} są odpowiednio najdłuższymi i najkrótszymi promieniami wyprowadzonymi ze środka ciężkości figury. Stosunek $AR = R_{max}/R_{min}$ może służyć jako miara wydłużenia figury (ang. object aspect ratio).

- ***Smukłość***- jest stosunkiem długości boków prostokąta granicznego opisanego na obiekcie, tzn. takiego, którego pole jest minimalne. Zatem smukłość wykorzystując parametry a,b możemy wyrazić wzorem:

$$R_s = \frac{a}{b}$$

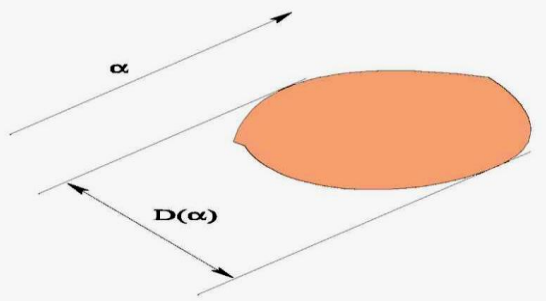
  Kryterium tego nie stosuje się do obiektów o kształtach zbliżonych do okręgu Wtedy smukłość liczymy, jako stosunek pola powierzchni obiektu do kwadratu jego szerokości *d*:

$$R_s = \frac{S}{(2d)^2}$$

- **Prostokątność** - jako stosunek pola powierzchni obiektu *S* do pola powierzchni prostokąta opisanego na tym obiekcie, którego pole jest minimalne.

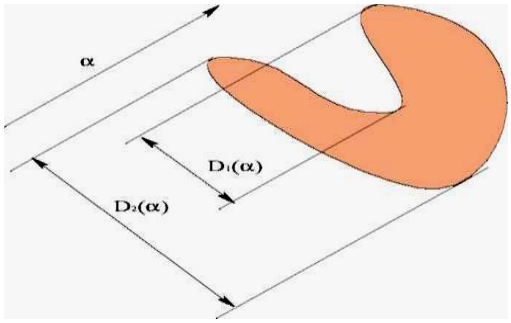
- ***Długość rzutów*** - gdy figura jest wypukła wystarczy jeden rzut, gdy jest wklęsła tworzymy kilka rzutów cząstkowych składających się na rzut rozwinięty.

  Rzutem figury $D(\alpha)$ w kierunku α wektora rzutowania nazywamy największą odległość pomiędzy wszystkimi prostymi, równoległymi do wektora $\vec{\alpha}$, mającymi część wspólną z analizowaną figurą.
  W przypadku figur wypukłych takich jak na rys. 79, mamy dwie linie styczne do figury, które mają dwa punkty wspólne z brzegiem figury [19].



Rys. 79. Rzut figury.

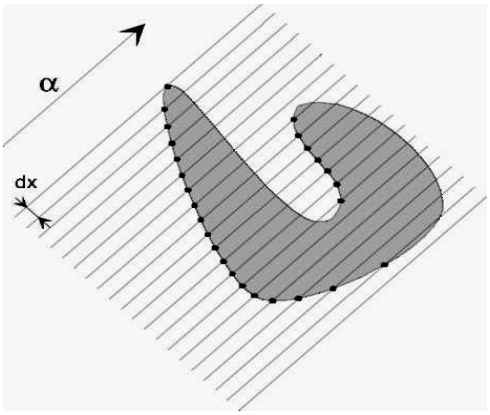
W przypadku figur wklęsłych w rejonie wklęsłości proste przecinające figurę maja więcej niż dwa punkty cząstkowe, zaznaczone odpowiednio jako *D1* i *D2,* które zsumowane tworzą rzut rozwinięty będący uogólnieniem normalnego rzutu prostopadłego [19].



Rys. 80. Rzut figury rozwinięty.

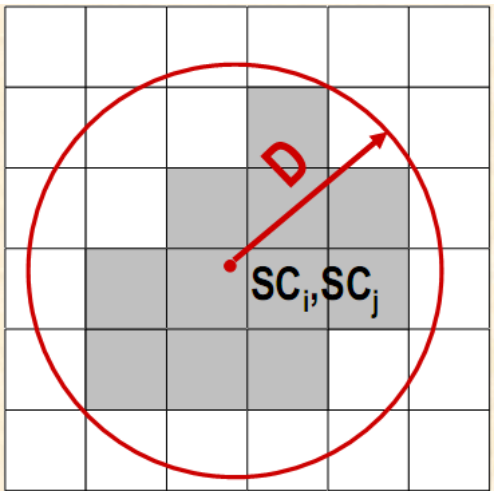
$$D(\alpha) = D_1 + D_2$$

Dla danej cząstki należy wyszukać wszystkie punkty, których lokalne otoczenie odpowiada „wchodzeniu" siecznej do cząstki, zliczyć te punkty i pomnożyć otrzymany wynik przez odległość między kolejnymi siecznymi *dx*. Gęstość linii rzutujących zależy od gęstości rastra tworzącego obraz [19].



Rys. 81. Wyznaczanie rzutu rozwiniętego figury.

- ***Średnika okręgu opisanego***



Rys. 82. Okrąg o średnicy D opisany na elemencie.

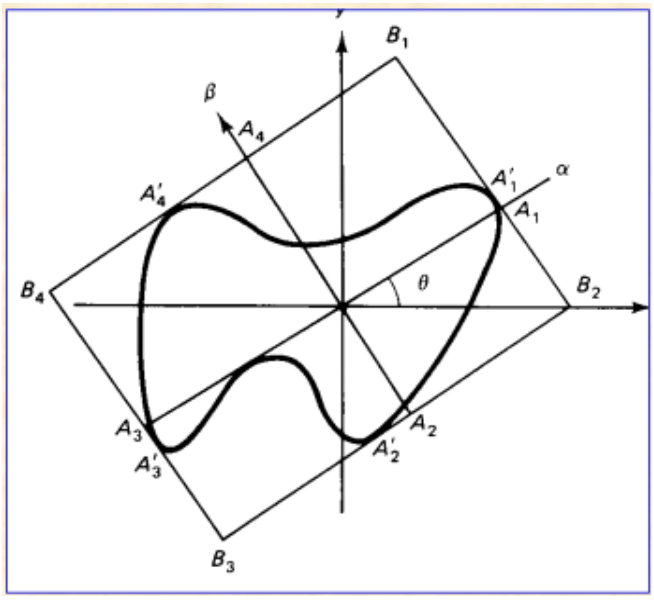
$$D = 2max\sqrt{(i_k - SC_i)^2 + \left(j_k - SC_j\right)^2}$$

$$k = 1,2,\dots,P$$

P- liczba elementów obiektu

- ***Prostokąt opisujący***

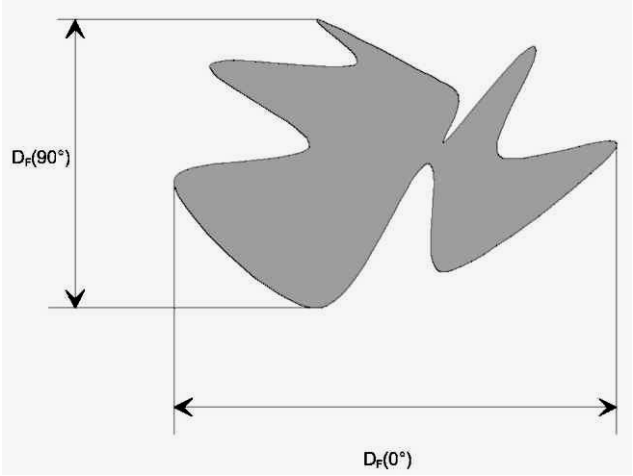
Prostokąt opisujący to prostokąt o najmniejszym polu powierzchni zawierający dany obiekt, przy czym kierunek dłuższego boku prostokąta jest równoległy do kierunku wyznaczającego oś najmniejszej bezwładności figury.



Rys. 83. Przykład prostokąta opisującego figurę.

- ***Średnice Fereta***

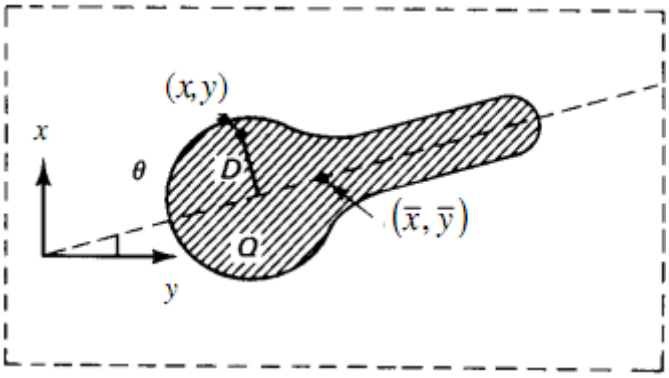
  Ułożenie przestrzenne obiektu można wyznaczyć przez rzutowanie obiektu na osie kartezjańskie. Najdłuższą cięciwą (Fereta) jest odcinek łączący najodleglejsze punkty brzegu figury. Wymiary obiektu wyrażające jego rozpiętość w poziomie i w pionie (współrzędne skrajnych punktów cząstki).



Rys. 84. Średnice Fereta.

- ***Kierunek***- jest to kierunek dłuższego z boków minimalnego prostokąta granicznego. Jeżeli momenty bezwładności pierwszego i drugiego rzędu są znane to kierunek można wyrazić wzorem:

$$\theta = \frac{1}{2}\tan^{-1}\left(\frac{2\mu_{11}}{\mu_{20} - \mu_{02}}\right)$$



Rys. 85. Przykład figury i osi o najmniejszym momencie bezwładności.

Wyliczanie kierunku ma sens jedynie dla obiektów smukłych. Smukłość i prostokątność są niezależne od transformacji liniowych takich jak przesunięcia, obrotu i skalowania. Kierunek zaś jest zależny jedynie od obrotu [10].

1.8.2 Cechy momentowe

Dwuwymiarowy moment rzędu *(p+q)* dla funkcji *f(x,y)*:

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q f(x,y)dxdy \qquad m_{pq} = \sum_{i=1}^{n}\sum_{j=1}^{n} i^p j^q x_{ij}$$

gdzie:

$f(x,y)$-intensywność w pikselu (x,y)
$p,q=0,..,n$

W procesie charakteryzowania figur przydatne są zwłaszcza momenty pierwszego i drugiego rzędu. Momenty pierwszego rzędu określają położenie środka ciężkości a momenty drugiego rzędu ($p=0$, $q=2$ lub $p=2$, $q=0$) są miarą bezwładności danego obiektu [10].

W przypadku rozważań geometrycznych można jeszcze wprowadzić pojęcie momentu konturowego wyznaczanego wzdłuż konturu L:

$$C_{pq} = \int_{A} x^p y^q \, dL$$

Momenty centralne: m_{pq}, C_{pq} tworzą cechy niezmiennicze względem przesunięcia. Za pomocą: m_{pq}, C_{pq} oraz m_{00}, C_{00} można skonstruować cechy niezależne względem przesunięcia, obrotu i zmiany skali:

Moment centralny f(x,y):

$$M_{pq} = \sum_{i=1}^{n}\sum_{i=1}^{n}\left(i-\tilde{i}\right)^p\left(j-\tilde{j}\right)^q x_{ij} \qquad M_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(x-\tilde{x})^p(y-\tilde{y})^q f(x,y)dxdy$$

gdzie:

$$\tilde{j} = \frac{m_{01}}{m_{00}} \qquad \tilde{i} = \frac{m_{10}}{m_{00}} \qquad \tilde{y} = \frac{m_{01}}{m_{00}} \qquad \tilde{x} = \frac{m_{10}}{m_{00}}$$

Momenty centralne można przedstawić za pomocą momentów zwykłych:

$$M_{00} = m_{00} \qquad M_{11} = m_{11} - \frac{m_{10} m_{01}}{m_{00}}$$

$$M_{01} = m_{01} - \left(\frac{m_{01}}{m_{00}}\right) m_{00} \qquad M_{02} = m_{02} - \frac{m_{01}^2}{m_{00}}$$

$$M_{10} = m_{10} - \left(\frac{m_{10}}{m_{00}}\right) m_{00} \qquad M_{20} = m_{20} - \frac{m_{10}^2}{m_{00}}$$

$$M_{21} = m_{21} - 2m_{11}\tilde{i} - m_{20}\tilde{j} + 2m_{01}\tilde{i}^2 \qquad M_{30} = m_{30} - 3m_{20}\tilde{i} + 2m_{10}\tilde{i}^2$$

$$M_{12} = m_{12} - 2m_{11}\tilde{j} - m_{02}\tilde{i} + 2m_{10}\tilde{j}^2 \qquad M_{03} = m_{03} - 3m_{02}\tilde{j} + 2m_{01}\tilde{j}^2$$

Z powyższych zależności możemy wyznaczyć niezmienniki momentowe:

$$\mathbf{M1} = \frac{\mathbf{M}_{20} + \mathbf{M}_{02}}{\mathbf{m}_{oo}^2}$$

$$\mathbf{M2} = \frac{(\mathbf{M}_{20} + \mathbf{M}_{02})^2 + 4\mathbf{M}_{11}^2}{\mathbf{m}_{oo}^4}$$

$$\mathbf{M3} = \frac{(\mathbf{M}_{30} + 3\mathbf{M}_{12})^2 + (3\mathbf{M}_{21} - \mathbf{M}_{03})^2}{\mathbf{m}_{oo}^5}$$

$$\mathbf{M4} = \frac{(\mathbf{M}_{30} + \mathbf{M}_{12})^2 + (\mathbf{M}_{21} - \mathbf{M}_{03})^2}{\mathbf{m}_{oo}^5}$$

$$\mathbf{M5} = \frac{(\mathbf{M}_{30} - 3\mathbf{M}_{12})(\mathbf{M}_{30} + \mathbf{M}_{12})\left[(\mathbf{M}_{30} + \mathbf{M}_{12})^2 - 3(\mathbf{M}_{21} + \mathbf{M}_{03})^2\right] +}{+ (3\mathbf{M}_{21} - \mathbf{M}_{03})(\mathbf{M}_{21} + \mathbf{M}_{03})\left[3(\mathbf{M}_{30} + \mathbf{M}_{12})^2 - (\mathbf{M}_{21} + \mathbf{M}_{03})^2\right]}{\mathbf{m}_{00}^{10}}$$

$$M6 = \frac{(M_{20} - M_{02})\left[(M_{30} + M_{12})^2 - (M_{21} + M_{03})^2\right] +}{} + \frac{4M_{11}(M_{30} + M_{12})(M_{21} + M_{03})}{m_{oo}^7}$$

$$M7 = \frac{M_{20}M_{02} - M_{11}^2}{m_{oo}^4}$$

$$M8 = \frac{M_{30}M_{12} + M_{21}M_{03} - M_{12}^2 - M_{21}^2}{m_{oo}^5}$$

$$M9 = \frac{M_{20}\left(M_{21}M_{03} - M_{12}^2\right) + M_{02}\left(M_{03}M_{12} - M_{21}^2\right) - M_{11}(M_{30}M_{03} - M_{21}M_{12})}{m_{oo}^7}$$

$$M10 = \frac{(M_{30}M_{03} - M_{12}M_{21})^2 - 4\left(M_{30}M_{12} - M_{21}^2\right)\left(M_{03}M_{21} - M_{12}\right)}{m_{oo}^{10}}$$

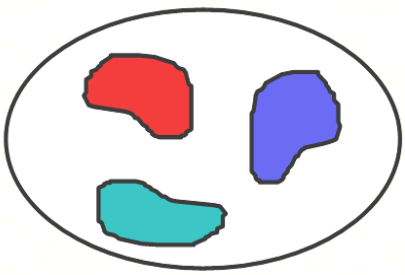
Powyższe zależności wyprowadził i opisał w swojej książce [18] R. Tadeusiewicz.

Niezmienniki momentowe są cechami o fundamentalnym znaczeniu przy rozpoznawaniu obiektów płaskich, bo umożliwiają rozpoznanie obiektów przy ich różnej lokalizacji w polu widzenia (w tym ruchome obiekty) oraz przy różnych powiększeniach układu (także ruchy kamer w górę i w dół). Istnieją także wyrażenia momentowe używające momentów obszarowych, które są niezmiennicze względem dowolnego nieosobliwego przekształcenia liniowego, dzięki czemu kamera może być dowolnie umieszczona względem obserwowanych obiektów płaskich. Metody momentowe (momenty konturowe) mogą też znaleźć zastosowanie do identyfikacji obiektów nie w pełni widocznych. Istota takiej identyfikacji polega na badaniu zmian, jakim ulegają momenty konturowe w miarę stopniowego przesłaniania obiektu. Przy wyborze cech uwzględnić należy zarówno rozdzielające własności poszczególnych cech, jak też ich obliczeniową złożoność (ilość obliczeń niezbędnych do wyznaczenia ich wartości) oraz inne praktyczne uwarunkowania [10].

1.8.3 Cechy topologiczne

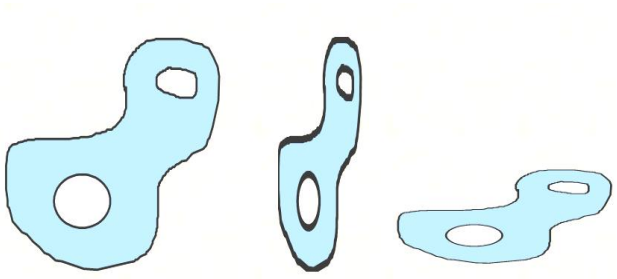
Topologiczne cechy kształtów to cechy, które są niezmienne względem tzw. transformacji „rubber-sheet", tj. przekształcenia, w których nie dopuszcza się cięcia płaszczyzny oraz tworzenia połączeń pomiędzy jej brzegami. Cechy topologiczne, oprócz cech geometrycznych, są dodatkowym sposobem opisu kształtu obiektów.

- ✓ ***Spójność obszaru***



Rys. 86. Obszar zawierający trzy spójne obiekty C=3.

- ✓ ***Liczba otworów***

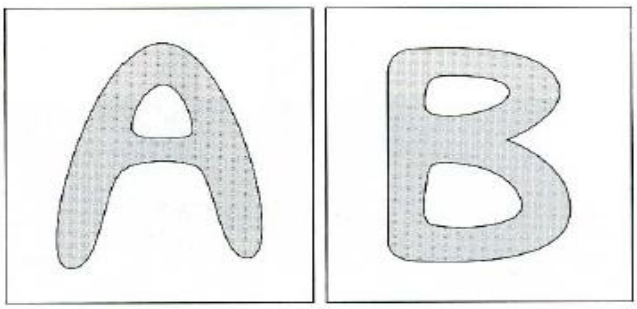


Rys. 87. Obszary z dwoma otworami H=2.

- ✓ ***Liczba Eulera*** jest zależnością pomiędzy spójnością obiektu i jego liczbą otworów:

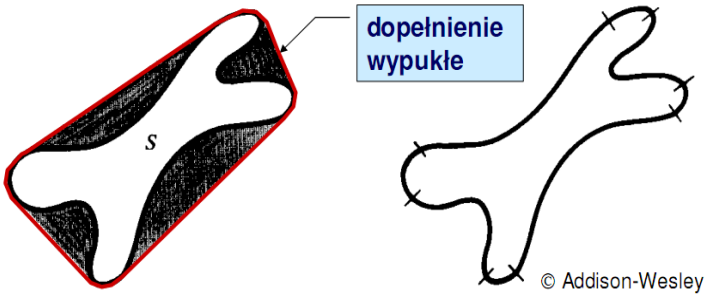
$$E = C - H$$

Jest to liczba obiektów C występująca na obrazie binarnym pomniejszona o liczbę „dziur" H w tych obiektach (może być ujemna). Liczba Eulera jest niezmienną cechą topologiczną.



Rys. 88. Przykład liczby Eulera dla litery A=0 i B=-1.

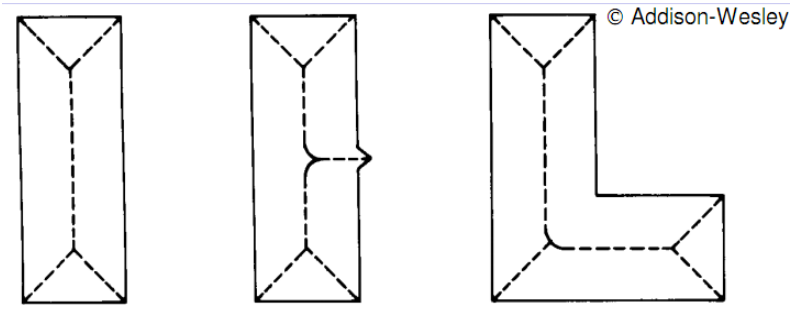
✓ ***Dopełnienie wypukłe*** C_H obszaru S jest obszarem o najmniejszym polu powierzchni, dla którego suma $C_H \cup S$ jest figurą wypukłą [16].



Rys. 89. Przykład dopełnienia wypukłego.

✓ ***Szkielet*** obiektu wyznacza się za pomocą algorytmów ścieniania (zwanych szkieletyzacją). Linia szkieletowa obiektu jest podstawową cechą kształtu wykorzystywaną w rozpoznawaniu znaków alfanumerycznych i innych zastosowaniach np. testowaniu jakości płytek drukowanych, granulometrii itd.

Szkielet obiektu można wyznaczyć za pomocą transformacji osi środkowej (ang. *medial axis transformation*). Oś środkowa to zbiór punktów obiektu, które mają więcej niż jeden punkt brzegu położony najbliżej punktu osi [16].



Rys. 90. Linie szkieletowe trzech przykładowych obiektów.

Cechy topologiczne (spójność, wklęsłość, wypukłość) szczególnie są użyteczne w przypadku analizy obrazów biomedycznych, gdyż występujące tam obiekty wykazują znaczne zróżnicowanie kształtów i spójności, a właśnie za pomocą pojęć topologicznych można opisać właściwości wspólne dla całej klasy obiektów. Na przykład liczba jąder komórkowych to typowa cecha topologiczna (liczba Eulera). Może ona być wykryta poprzez określenie liczby otworów w obszarach komórki. Analizę przeprowadza się po binaryzacji obrazu z odpowiednim progiem odcięcia. Cechy topologiczne są niezależne od położenia obiektu względem układu współrzędnych, co może być ich zaletą, ale dotyczą zbyt ogólnych własności obiektów i dlatego są mało praktyczne do celów przemysłowych.

1.8.4 Metody identyfikacji

Rys. 91. Metody identyfikacji obrazu.

System rozpoznawania obrazu można zaprojektować w formie ustalonego szablonu decyzyjnego lub jako system uczący się na podstawie uaktualnianych opisów cech.
Faza uczenia polega na optymalizowaniu zestawu sklasyfikowanych wektorów cech i na ich podstawie modyfikacji postaci funkcji rozdzielającej. Poza etapem uczenia działanie algorytmów identyfikacji obrazów przebiega analogicznie. Składają się na niego trzy podstawowe operacje:

1. Przypisanie obiektom q wektorów cech $\bar{x}(q)$ w przestrzeni n-wymiarowej

2. Obliczenie wartości funkcji przynależności. Jest ona miarą stopnia dopasowania obiektu opisanego wektorem do klasy i. W przypadku niejednoznacznej klasyfikacji obiektu, analizuje się prawdopodobieństwo przynależności do danej klasy:

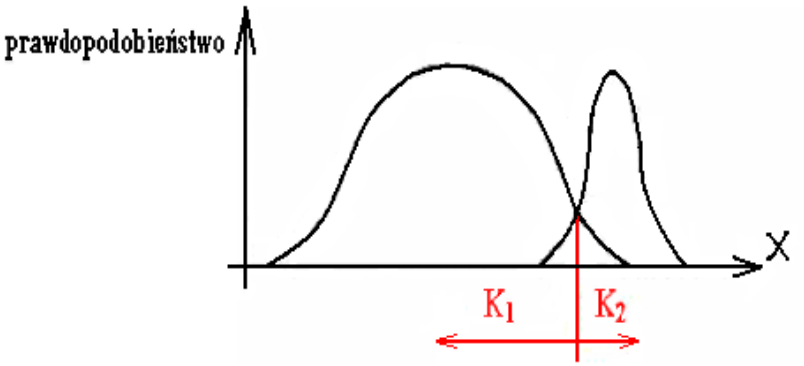
$$q_i(x) = P(q \epsilon K_i) = \frac{P(\bar{x}|K_i)P(K_i)}{\sum P(\bar{x}|K_i)P(K_i)}$$

gdzie:

$P(\bar{x}|K_i)$- rozkład gęstości prawdopodobieństwa dla wektora cech w obrębie klasy K_i

$P(K_i)$- prawdopodobieństwo pojawienia się obiektów z klasy K_i

3. Proces podejmowania decyzji, czyli przyporządkowywanie obiektu q opisanego wektorem cech do klasy i, dla której wartość funkcji przynależności jest maksymalna. W przypadku przestrzeni charakteryzowanej przez dwie klasy obiektów, podjęcie decyzji o zakwalifikowaniu do jednej z nich odbywa się na zasadzie określenia stopnia przynależności, czyli prawdopodobieństwa, z jakim dany obiekt należy do jednej z klas K_1 lub K_2 [10].



Rys. 92. Zasada określania stopnia przynależności obiektu do danej klasy.

2 Analiza i tworzenie modeli układów logicznych

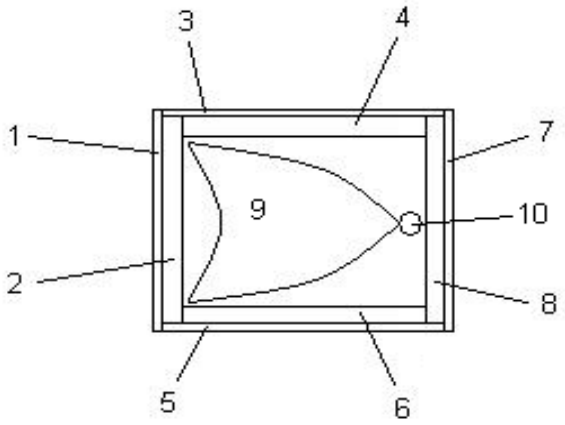
2.1 Technika rozpoznawania obrazu w programie *bramki*

2.1.1 Model bramki jako obiektu złożonego

W poniższym rozdziale przedstawiono kolejne kroki realizacji oprogramowania do rozpoznawania bramek oraz połączeń między nimi.

Podstawą rozpoznawania obiektów w stworzonym w ramach niniejszej pracy programie *bramki* jest ich kilkustopniowa analiza zgodnie z przyjętym modelem bramki. Model ten zakłada, że bramka logiczna jest obiektem złożonym, tzn. takim, którego poszczególne elementy w procesie rozpoznawania można traktować osobno, a całościowo dające informację o rozpoznawanej bramce.

Model bramki przyjęty w pracy składa się z następujących elementów: część główna – jest to bramka logiczna NOT, AND lub OR, negacja – kółko, symbol oznaczający negację, otoczenie lewe, otoczenie górne, otoczenie dolne, otoczenie prawe, otoczenie jedno-pikselowe lewe, otoczenie jedno-pikselowe górne, otoczenie jedno-pikselowe dolne, otoczenie jedno-pikselowe prawe.

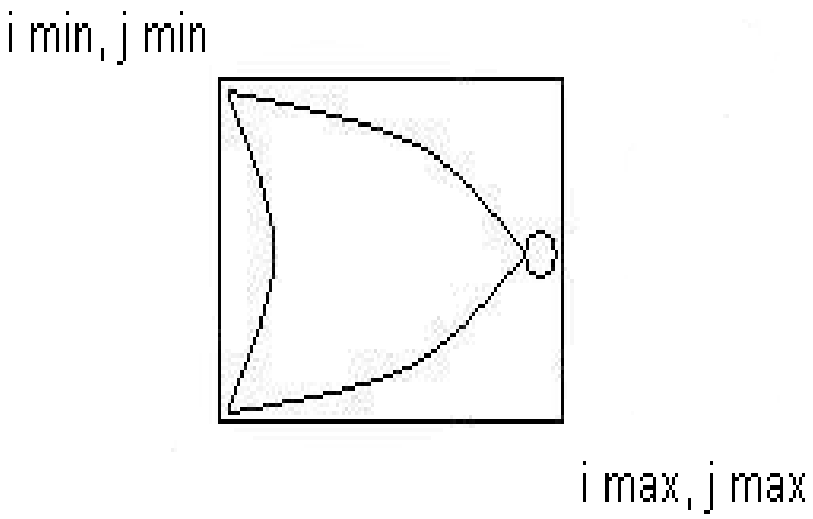


Rys. 93. Model bramki jako obiektu złożonego.

Na model bramki w programie *bramki* składają się następujących elementy:

1. Otoczenie jedno-pikselowe lewe
2. Otoczenie lewe
3. Otoczenie jedno-pikselowe górne
4. Otoczenie górne
5. Otoczenie jedno-pikselowe dolne
6. Otoczenie dolne
7. Otoczenie jedno-pikselowe prawe
8. Otoczenie prawe
9. Część główna
10. Negacja

Identyfikacja części głównej odbywa się po wycięciu z całego obrazu części zawierającej prostokąt o współrzędnych $i_{min}, j_{min}, i_{max}, j_{max}$ w którym mieści się cała bramka która ma zostać rozpoznana.



Rys. 94. Model zawierający współrzędne prostokąta w którym znajduje się bramka.

Jeśli obiekt zawarty w prostokącie zawiera dwie części tzn. bramkę i symbol negacji to zostaje on rozdzielony, po czym osobno jest rozpoznawana bramka, a osobno kółko oznaczające negację. Jeżeli w prostokącie znajduje się tylko jeden obiekt, przyjmuje się, że jest to bramka i zostaje rozpoznawany jako bramka. Jako kryterium służące do odróżniania negacji od bramki przyjęto stosunek pół obiektów. Jeżeli stosunek jednego obiektu do drugiego jest mniejszy niż przyjęty próg to zakładamy, że obiekt o mniejszym polu jest rozpoznany jako negacja, a drugi jako bramka. Część rozpoznana jako bramka zastaje poddana dalszej analizie, kolejno są obliczane następujące parametry: pole powierzchni, obwód, współczynnik Malinowskiej, współczynnik Blaira-Blissa, współczynnik Lp1, a także dodatkowo niezmienniki momentowe M1, M2 oraz M7. Sposób obliczania poszczególnych parametrów został opisany już wcześniej w części teoretycznej. Wspomnieć należy jedynie, że w pracy do pełnej identyfikacji bramek zostały użyte współczynniki kształtu Malinowskiej, Blaira-Blissa oraz Lp1, gdyż dają one wystarczającą dokładność oraz duże prawdopodobieństwo poprawności rozpoznania bramek, czego dowodzą opisane w kolejnym rozdziale liczne przykłady i analizy.

Analizując środek prostokąta zawierającego część główną można rozpoznać następujące rodzaje bramek: NOT, AND, OR, NAND oraz NOR. Pozostałe bramki XOR oraz XNOR mogą zostać rozpoznane przez sprawdzenie poszczególnych otoczeń bramki: lewego,

górnego, dolnego oraz prawego. W celu rozpoznania „linii” oznaczającej XOR zostaje obliczana suma „zaświeconych” pikseli w danym otoczeniu bramki.

Końcowym etapem rozpoznawania bramek jest analiza połączeń pomiędzy nimi, czyli sprawdzenie, która bramka z którą jest połączona. W tym celu wprowadzono w modelu dodatkowe otoczenia: otoczenie jedno-pikselowe lewe, otoczenia jedno-pikselowe górne, otoczenia jedno-pikselowe dolne, otoczenie jedno-pikselowe prawe. Po operacji indeksacji połączeń pomiędzy bramkami, zostają przeszukiwane otoczenia bramek: otoczenie jedno-pikselowe lewe, otoczenie jedno-pikselowe górne, otoczenie jedno-pikselowe dolne oraz otoczenie jedno-pikselowe prawe. Jeśli po tej samej stronie bramki pojawią się dwa obiekty, którym w procesie indeksacji zostały przypisane dwa różne indeksy to przyjmujemy, że są to wejścia bramek, jeśli jest jeden obiekt to przyjmujemy, że jest to wyjście bramki.

Jeśli chodzi o rozpoznawanie samych bramek to jest ono realizowane w następujący sposób: po obliczeniu współczynników Malinowskiej, Blaira-Blissa oraz współczynnika Lp1 jest podejmowana decyzja o rodzaju bramki. W pierwszej kolejności rozpoznawana jest bramka AND od bramek NOT oraz OR za pomocą dwóch pierwszych współczynników, a następnie za pomocą współczynnika Lp1 jest odróżniana bramka NOT od bramki OR. Wartości progowe tych współczynników zostały dobrane eksperymentalnie na podstawie przeprowadzonych doświadczeń. Jeśli wartości współczynników dla danej bramki mieszczą się w danym zakresie to program wyświetli na końcowym schemacie nazwę tej bramki.

2.1.2 Struktura macierzy opisującej rozpoznawany układ

W celu ułatwienia pisania programu i możliwości szybkiego odnajdywania pożądanych cech wprowadzono w programie *bramki* macierz G, która zawiera większość potrzebnych parametrów, takich jak np. rodzaj bramki, wartość współczynników kształtu czy informację o połączeniach pomiędzy bramkami.

Ogólna budowa macierzy zastała podana poniżej.

$$\begin{bmatrix} nr\ kolumny = nr\ rozpoznawanej\ bramki \\ współrzędna\ i_{min}\ \ prostokąta\ w\ którym\ znajduje\ się\ bramka \\ współrzędna\ j_{max}\ \ prostokąta\ w\ którym\ znajduje\ się\ bramka \\ współrzędna\ i_{min}\ \ prostokąta\ w\ którym\ znajduje\ się\ bramka \\ współrzędna\ j_{max}\ \ prostokąta\ w\ którym\ znajduje\ się\ bramka \\ 0 - br.\ bez\ negacji, 1 - br.\ z\ negacją \\ 1 - br.\ NOT, 2 - br.\ OR, 3 - br.\ AND \\ 0 - br.\ zwykła, 1 - XOR \\ pusta \\ indeks\ pierwszego\ połączenia\ w\ otoczeniu\ 1 \\ indeks\ drugiego\ połączenia\ w\ otoczeniu\ 1 \\ indeks\ pierwszego\ połączenia\ w\ otoczeniu\ 7 \\ indeks\ drugiego\ połączenia\ w\ otoczeniu\ 7 \\ indeks\ pierwszego\ połączenia\ w\ otoczeniu\ 3 \\ indeks\ drugiego\ połączenia\ w\ otoczeniu\ 3 \\ indeks\ pierwszego\ połączenia\ w\ otoczeniu\ 5 \\ indeks\ drugiego\ połączenia\ w\ otoczeniu\ 5 \\ pusta \\ pusta \\ obwód\ figury\ L \\ pole\ figury\ S \\ pole\ częściowe\ figury\ S1 \\ współczynnik\ Blaira - Blissa \\ współczynnik\ Malinowskiej \\ niezmiennik\ momentowy\ M1 \\ niezmiennik\ momentowy\ M2 \\ niezmiennik\ momentowy\ M7 \end{bmatrix}$$

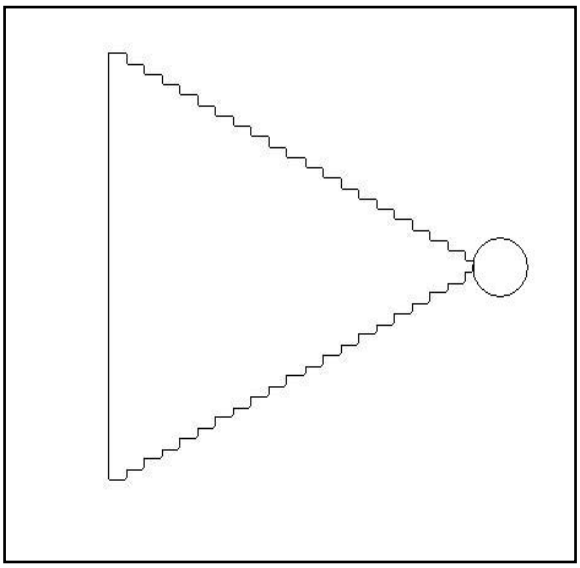
2.2 Wyniki przeprowadzonych analiz w programie *bramki*

2.2.1 Analiza wyników rozpoznawania pojedynczych bramek

W niniejszym rozdziale omówione zostały wyniki analizy obrazów zawierających pojedyncze bramki, a także porównane wartości współczynników *Malinowskiej* i *Blaira-Blissa*, niezmienników momentowych *M1*, *M2*, *M7* oraz współczynnika *Lp1* na podstawie których podejmowana jest decyzja o rodzaju rozpoznanej bramki.

Pierwszym krokiem było obliczenie współczynników dla bramek wzorcowych, których wartości były punktem odniesienia w procesie rozpoznawania obiektów testowych.

Na rysunku 95 przedstawiono obraz bramki wzorcowej NOT oraz wartości obliczonych dla niej współczynników zestawionych w tabeli 2. Wartości współczynników tej bramki jak i wartości współczynników bramek testowych (znajdujące się w tabeli 12 w dodatku C) decydują o doborze progu na podstawie którego dalej jest podejmowana decyzja o rodzaju bramki. Widać, że dla bramki wzorcowej (idealnej) wartość współczynnika *Malinowskiej* wynosi *0.3235* natomiast średnia wartość wynikająca z obliczenia tego współczynnika dla serii testowej bramek NOT wynosi ok. *0.5* co wynika z odręcznego narysowania bramki. Różnią się one więc znacznie wartościami. Bardziej zbliżone do siebie są natomiast wartości współczynników *Blaira-Blissa*, którego wartości zarówno dla bramki wzorcowej jak i dla średniej wartości obliczonej dla serii bramek testowych jest niemal identyczna i wynosi *0.6602*. Również wartości współczynnika *Lp1* zarówno dla bramki wzorcowej jak i dla bramek testowych mają podobne wartości i są równe *0.1725*.

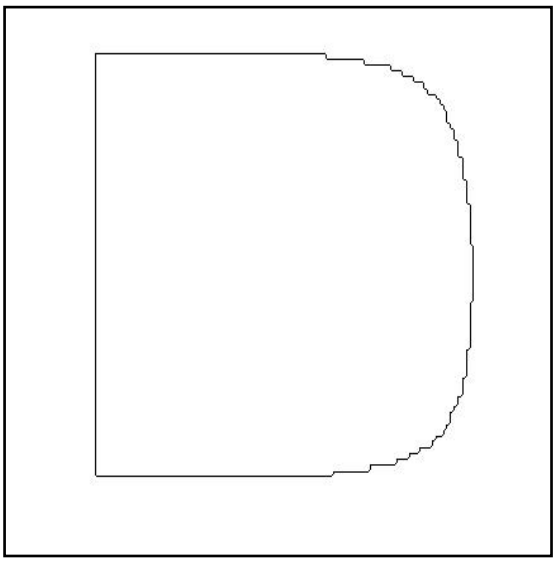


Rys. 95. Obraz bramki wzorcowej NOT.

Tabela 2. Wartości niezmienników momentowych M1, M2, M7 oraz współczynników Malinowskiej, Blaira-Blissa oraz Lp1 dla bramki wzorcowej NOT.

Bramka NOT	
Wielkość	Wartość
M1	0.3652
M2	0.1334
M7	0.0289
W. Malinowskiej	0.3235
W. Blaira-Blissa	0.6602
Lp1	0.1725

Rysunek 96 to obraz bramki wzorcowej AND a poniżej wartości opisujących ją współczynników przedstawione w tabeli 3. Wartość współczynnika Malinowskiej wynosi 2.9523, co potwierdza dodatkowo podobna wartość tego współczynnika otrzymana z obliczenia go dla serii bramek testowych AND. Współczynnik Blaira-Blissa dla bramki wzorcowej ma wartość 0.2210, natomiast dla bramek wzorcowych mieści się w zakresie od 0.2113 do 0.2838.

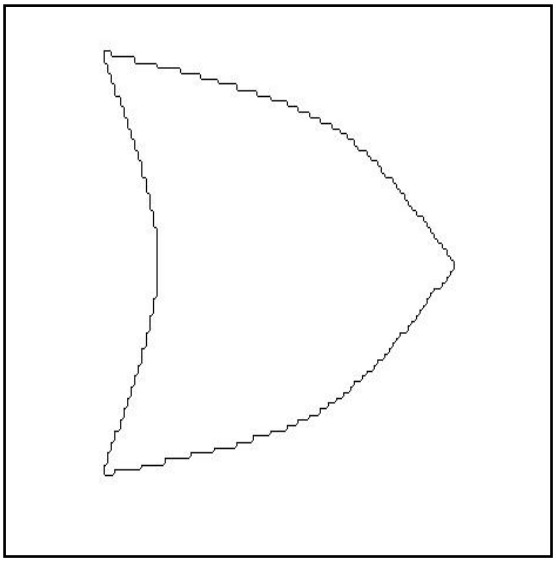


Rys. 96. Obraz bramki wzorcowej AND.

Tabela 3. Wartości niezmienników momentowych M1, M2, M7 oraz współczynników Malinowskiej, Blaira-Blissa oraz Lp1 dla bramki wzorcowej AND.

Bramka AND	
Wielkość	Wartość
M1	3.2585
M2	10.6197
M7	2.0751
W. Malinowskiej	2.9523
W. Blaira-Blissa	0.2210
Lp1	0.1616

Dla bramki wzorcowej OR, pokazanej na rysunku 97 wartości wszystkich współczynników dla bramki wzorcowej jak i dla serii bramek testowych są zbliżone i wynoszą kolejno: dla współczynnika Malinowskiej 0.5682, dla współczynnika Blaira-Blissa 0.5243 oraz dla współczynnika Lp1 0.3861.



Rys. 97. Obraz bramki wzorcowej OR.

Tabela 4. Wartości niezmienników momentowych M1, M2, M7 oraz współczynników Malinowskiej, Blaira-Blissa oraz Lp1 dla bramki wzorcowej OR.

Bramka OR	
Wielkość	**Wartość**
M1	0.5789
M2	0.3351
M7	0.0837
W. Malinowskiej	0.5682
W. Blaira-Blissa	0.5243
Lp1	0.3861

W tabeli 5 znajduje się zestawienie obliczonych wartości współczynników dla bramek wzorcowych. Wyraźnie widać, że wartość współczynnika Malinowskiej dla bramki AND wynosi 2.9523 i znacząco różni się od wartości dla bramki NOT (0.3235) oraz OR (0.5682), co stanowi znakomitą i pewną podstawę do rozróżniania tej pierwszej od dwóch pozostałych. Podobnie jest ze współczynnikiem Blaira-Blissa. Dla bramki AND wynosi 0.221 i jest dużo niższy niż dla bramki NOT dla której wynosi 0.6602 oraz dla bramki OR dla której wynosi 0.5243. Różnice te zostaną potwierdzone przy analizie serii odręcznie narysowanych bramek, gdyż są zbliżone do bramek wzorcowych.

Inaczej jest natomiast ze współczynnikiem Lp1, będącym jak już było wcześniej wspomniane stosunkiem wartości promienia okręgu wpisanego do wartości promienia okręgu opisanego. Tu dla odmiany niemal identyczne są wartości tego współczynnika dla bramek NOT i AND i wynoszą odpowiednio 0.1725 dla bramki NOT oraz 0.1616 dla AND-a. Dla bramki OR wartość jest większa i wynosi 0.3861.

Tabela 5. Porównanie wartości niezmienników momentowych M1, M2, M7 oraz współczynników Malinowskiej, Blaira-Blissa oraz Lp1 dla bramek wzorcowych NOT, OR oraz AND.

Współczynnik	Bramka NOT	Bramka OR	Bramka AND
M1	0.3652	0.5789	3.2585
M2	0.1334	0.3351	10.6197
M7	0.0289	0.0837	2.0751
W. Malinowskiej	0.3235	0.5682	2.9523
W. Blaira-Blissa	0.6602	0.5243	0.2210
Lp1	0.1725	0.3861	0.1616

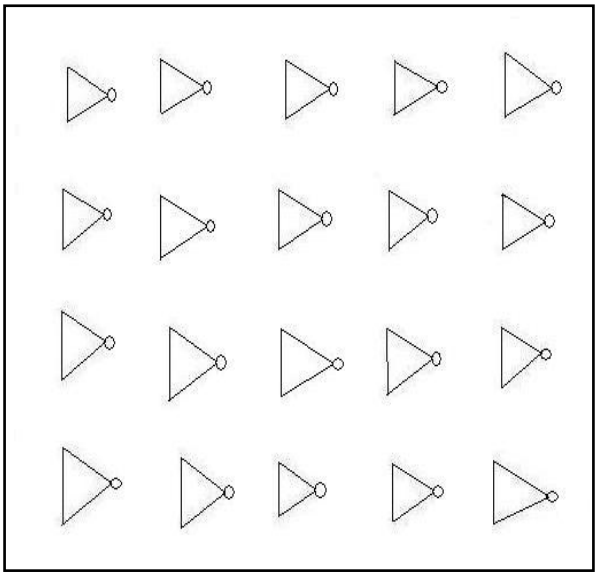
2.2.2 Analiza wyników rozpoznawania serii pojedynczych bramek

Kolejnym ważnym etapem testowania programu po obliczeniu wartości momentów oraz współczynników dla bramek wzorcowych, była symulacja przeprowadzona dla serii dwudziestu odręcznie narysowanych bramek NOT, AND oraz OR. Celem tej symulacji było sprawdzenie skuteczności oraz zachowania się programu w zależności od różnic w narysowanych bramkach oraz jego radzenia sobie z rozpoznawaniem bramek narysowanych w różnej skali oraz proporcjach. Inną zaletą takiego testu programu jest również możliwość obserwacji zakresu zmian poszczególnych niezmienników momentowych oraz współczynników kształtu dla poszczególnej bramki.

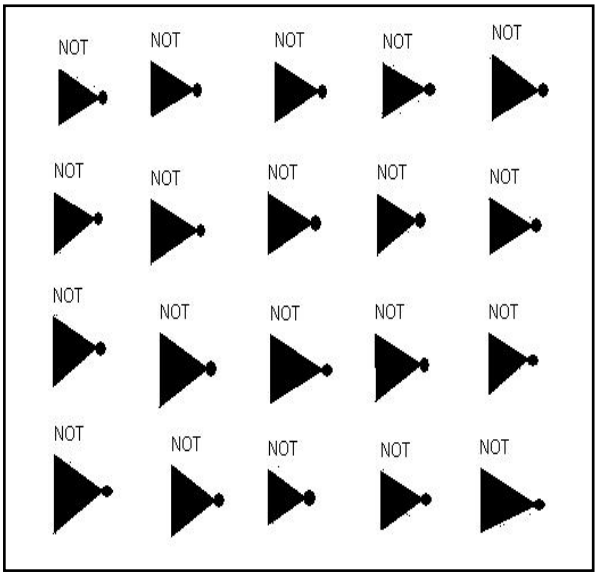
Dane takie pozwalają na weryfikację poprawnego doboru progów wartości parametrów na podstawie których podejmowana jest decyzja o rodzaju bramki, w tym przypadku jest to współczynnik Malinowskiej, współczynnik Blaira-Blissa oraz współczynnik Lp1. Zostały również obliczone niezmienniki momentowe M1, M2 oraz M7, w celu określenia ich przydatności w procesie rozpoznawania bramek oraz pokazania ich jako alternatywy dla rozpoznawania bazującego na wymienionych wcześniej trzech współczynnikach kształtu.

W pierwszej kolejności testowi zostało poddanych dwadzieścia odręcznie (czyli mające przypadkowo dobrane proporcje i skalę) narysowanych bramek NOT dla których zostały obliczone wartości współczynników kształtu oraz niezmienników momentowych. Zbiór bramek, na których wykonano test pokazuje rysunek 98.

Już na pierwszy rzut oka można zauważyć jak znaczące są różnice pomiędzy poszczególnymi bramkami. Na rysunku 99 mamy wynik symulacji programu. Jak widać wszystkie bramki zostały rozpoznane prawidłowo bez większych problemów. Z tabeli 6 widać, że współczynnik Malinowskiej dla bramki NOT zmienia się w zakresie 0.4522 do 0.5450, natomiast współczynnik Blaira-Blissa od 0.6515 do 0.6671. Współczynnik Lp1 przyjmuje wartości z zakresu od 0.0944 do 0.1744. w tabeli 12 w **Dodatku C** zostały zestawione ze sobą wartości niezmienników momentowych oraz współczynników kształtu dla bramek NOT, AND oraz OR co znacznie ułatwia ich interpretację i porównanie.



Rys. 98. Zbiór bramek testujących NOT

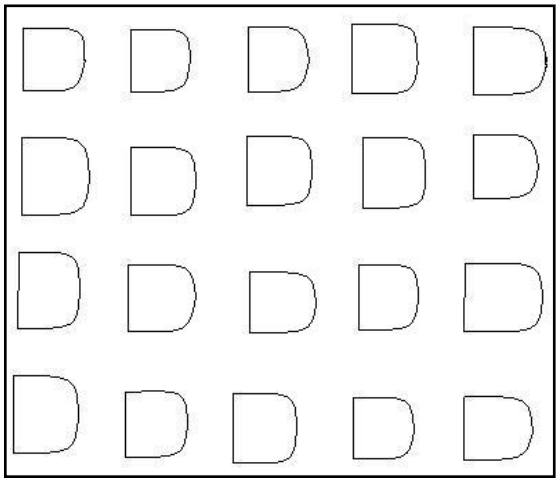


Rys. 99. Efekt symulacji programu.

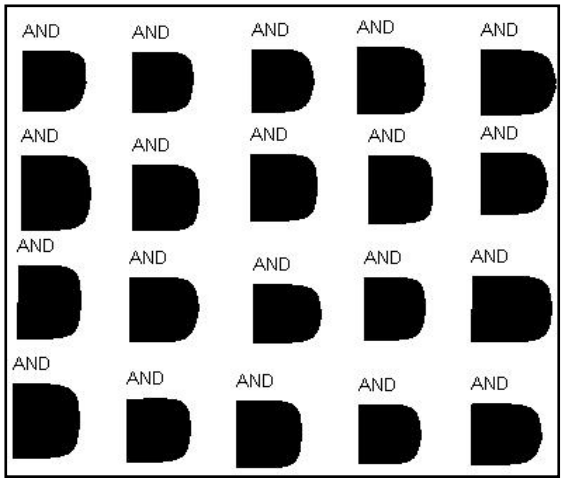
Tabela 6. Wartości współczynników kształtu oraz niezmienników momentowych dla serii bramek NOT.

Nr bramki	M1	M2	M7	W. Malinowskiej	W. Blaira-Blissa	Lp1
1.	0.3637	0.1325	0.0282	0.4787	0.6615	0.1045
2.	0.3692	0.1363	0.0293	0.4912	0.6566	0.0944
3.	0.3640	0.1328	0.0285	0.4522	0.6612	0.1201
4.	0.3727	0.1389	0.0294	0.5163	0.6535	0.1516
5.	0.3749	0.1406	0.0299	0.5126	0.6515	0.1668
6.	0.3634	0.1321	0.0283	0.4830	0.6618	0.1520
7.	0.3576	0.1280	0.0275	0.4540	0.6671	0.1366
8.	0.3666	0.1344	0.0286	0.5415	0.6589	0.1588
9.	0.3594	0.1291	0.0275	0.4864	0.6655	0.1295
10.	0.3601	0.1297	0.0275	0.5450	0.6648	0.1039
11.	0.3673	0.1349	0.0291	0.4698	0.6583	0.1744
12.	0.3656	0.1337	0.0287	0.4875	0.6598	0.1678
13.	0.3717	0.1384	0.0312	0.4546	0.6543	0.1414
14.	0.3688	0.1360	0.0293	0.4912	0.6569	0.1136
15.	0.3625	0.1314	0.0281	0.4912	0.6626	0.1399
16.	0.3630	0.1318	0.0284	0.5301	0.6622	0.1613
17.	0.3737	0.1396	0.0302	0.5493	0.6526	0.1190
18.	0.3618	0.1316	0.0282	0.4843	0.6633	0.0731
19.	0.3679	0.1353	0.0289	0.5026	0.6578	0.1676
20.	0.3648	0.1331	0.0289	0.5032	0.6605	0.1205

Rysunek 100 to zbiór dwudziestu bramek do użytych do testu programu do poprawności rozpoznawania bramki AND, natomiast na następnym rysunku 101 wynik symulacji w programie *bramki*. Widać, że i tym razem program doskonale sobie poradził z rozpoznaniem wszystkich bramek. Wartości współczynników uzyskane w wyniku symulacji zostały umieszczone w tabeli 7. Wartość współczynnika Malinowskiej zmienia się w zakresie od 2.3141 do 3.3478. Współczynnik Blaira-Blissa przyjmuje wartości od 0.2004 do 0.2853, natomiast współczynnik Lp1 od 0.1080 do 0.3171.



Rys. 100. Zbiór bramek testujących AND.

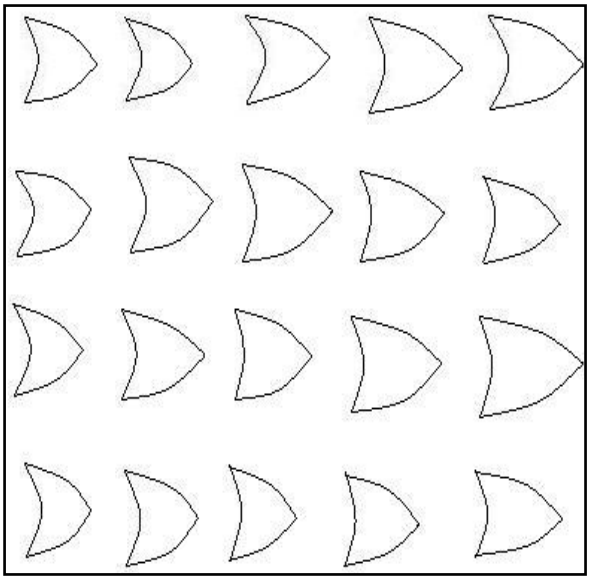


Rys. 101. Efekt symulacji programu.

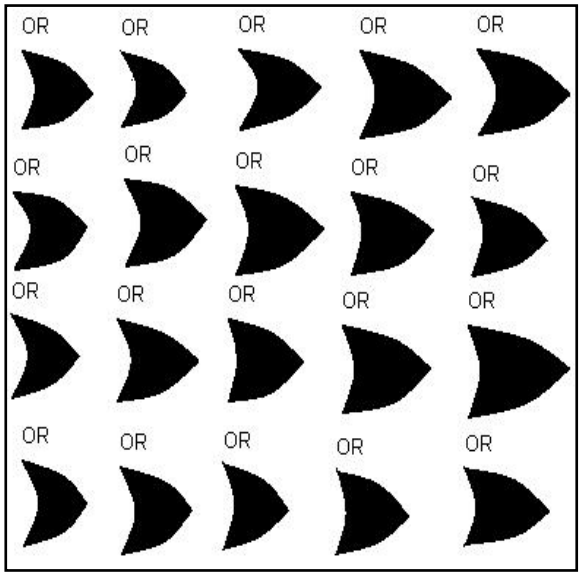
Tabela 7. Wartości współczynników kształtu oraz niezmienników momentowych dla serii bramek AND.

Nr bramki	M1	M2	M7	W. Malinowskiej	W. Blaira-Blissa	Lp1
1.	2.8262	7.9879	1.4959	2.7932	0.2373	0.1740
2.	3.5016	12.7096	2.9312	2.7525	0.2132	0.3171
3.	2.7155	7.3774	1.4091	2.7908	0.2421	0.1538
4.	2.3171	5.3697	1.0686	2.7283	0.2621	0.1080
5.	3.0156	9.3261	1.9276	2.7226	0.2297	0.2119
6.	2.3250	5.4056	1.0303	2.6472	0.2616	0.1175
7.	2.6241	6.8888	1.3641	2.8383	0.2463	0.1274
8.	3.3704	11.3612	2.2698	3.1468	0.2173	0.1625
9.	3.5657	12.7158	2.5747	3.2362	0.2113	0.1635
10.	3.3756	11.3964	2.2500	3.1082	0.2171	0.1722
11.	1.9765	3.9130	0.6995	2.3693	0.2838	0.1318
12.	2.4119	5.8175	1.0611	2.5900	0.2569	0.1697
13.	2.7879	7.7729	1.5546	2.9216	0.2389	0.1230
14.	2.7793	7.7243	1.5016	2.8829	0.2393	0.1487
15.	2.9186	8.5203	1.6683	2.9321	0.2335	0.1420
16.	3.9636	15.7249	3.1453	3.3478	0.2004	0.1865
17.	3.8191	14.9102	3.5216	2.9636	0.2041	0.2762
18.	2.1715	4.7152	0.8485	2.4479	0.2707	0.1526
19.	1.9548	3.8214	0.6834	2.3141	0.2853	0.1411
20.	2.2184	4.9281	0.9047	2.5358	0.2679	0.1296

Test dla bramek OR został przeprowadzony dla serii bramek na rysunku 102. Efekt symulacji pokazuje rysunek 103 i jak widać również w tym przypadku wszystkie bramki zostały rozpoznane poprawnie. Wartości współczynników kształtu Malinowskiej, Blaira-Blissa oraz Lp1 zmieniają się w zakresach odpowiednio: od 0.5803 do 0.7260, od 0.5117 do 0.5901 oraz od 0.2947 do 0.4278.



Rys. 102. Zbiór bramek testujących OR



Rys. 103. Efekt symulacji programu.

Tabela 8. Wartości współczynników kształtu oraz niezmienników momentowych dla serii bramek OR.

Nr bramki	M1	M2	M7	W. Malinowskiej	W. Blaira-Blissa	Lp1
1.	0.4571	0.2089	0.0522	0.5803	0.5901	0.3957
2.	0.5734	0.3287	0.0817	0.7260	0.5269	0.3818
3.	0.5404	0.2921	0.0730	0.6731	0.5427	0.3266
4.	0.4755	0.2261	0.0564	0.5916	0.5785	0.3644
5.	0.5158	0.2662	0.0665	0.6231	0.5555	0.3303
6.	0.5214	0.2719	0.0679	0.6190	0.5525	0.4278
7.	0.5043	0.2543	0.0635	0.6579	0.5618	0.3946
8.	0.5522	0.3049	0.0761	0.6520	0.5369	0.4193
9.	0.5227	0.2733	0.0683	0.6404	0.5518	0.3462
10.	0.5766	0.3328	0.0828	0.6904	0.5254	0.2804
11.	0.5504	0.3029	0.0757	0.6452	0.5378	0.3535
12.	0.4814	0.2318	0.0579	0.5848	0.5750	0.4006
13.	0.5533	0.3061	0.0764	0.6433	0.5363	0.3363

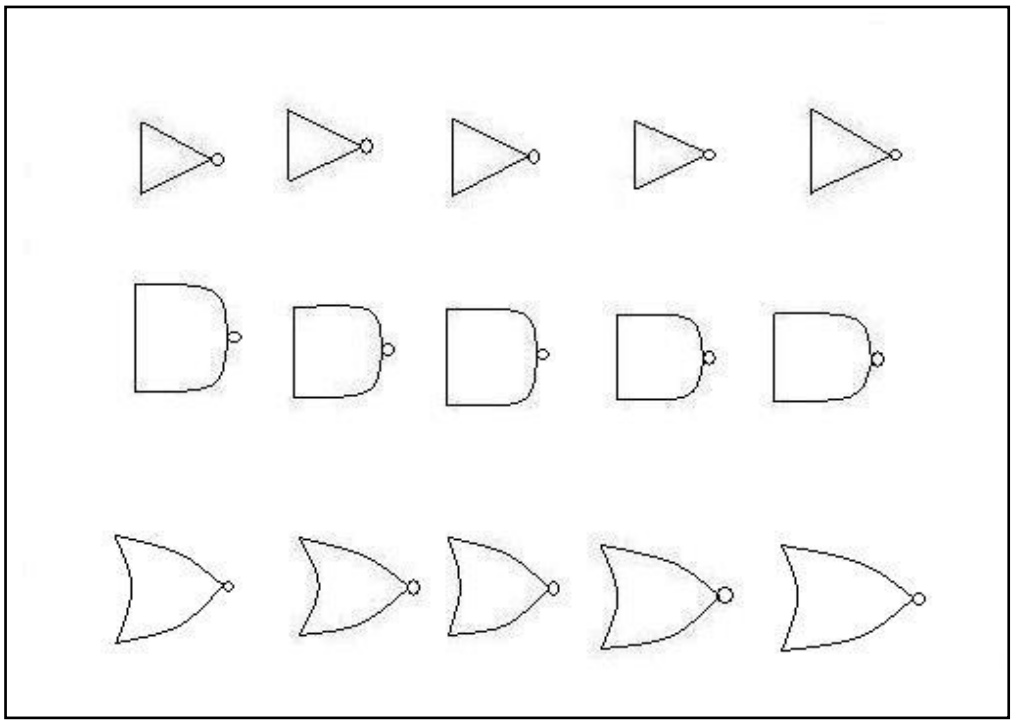
14.	0.5797	0.3363	0.0839	0.6705	0.5240	0.3300
15.	0.5579	0.3115	0.0776	0.6547	0.5341	0.3166
16.	0.5352	0.2864	0.0716	0.6253	0.5453	0.3155
17.	0.6078	0.3695	0.0923	0.7239	0.5117	0.2947
18.	0.5479	0.3002	0.0747	0.6134	0.5390	0.3185
19.	0.5562	0.3095	0.0773	0.6348	0.5349	0.3826
20.	0.5223	0.2730	0.0681	0.6101	0.5520	0.3915

Porównując współczynniki kształtu otrzymane dla każdej z serii odręcznie narysowanych bramek NOT, AND oraz OR można zweryfikować poprawność doboru progów, według których możliwe jest poprawne przeprowadzenie identyfikacji poszczególnych bramek.

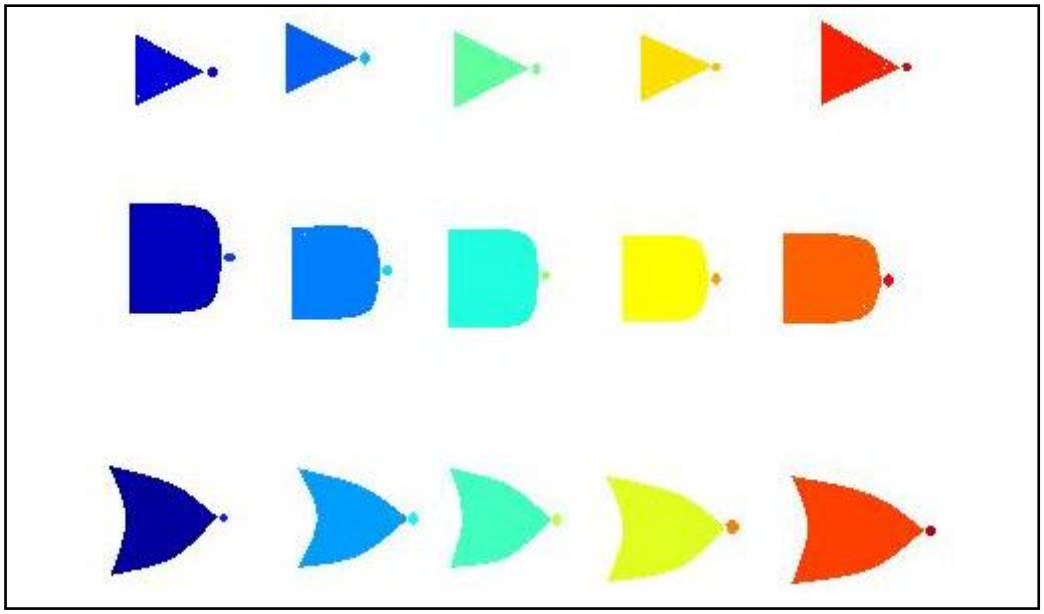
Dla bramki NOT współczynnik Malinowskiej zmienia się w zakresie 0.4522 do 0.5450, dla bramki OR od 0.5803 do 0.7260, a dla bramki AND od 2.3141 do 3.3478. Można stąd łatwo zauważyć, że wartości tego współczynnika dla bramki NOT i OR są bardzo zbliżone do siebie, dlatego nie umożliwia on rozróżnienia tych dwóch rodzajów bramek. Współczynnik ten jest natomiast bardzo przydatny przy identyfikacji bramki AND. Jego wartość jest dla AND-a o ok. 2 większa niż dla dwóch pozostałych. W programie przyjęto więc dwa progi identyfikacji: wartości mniejsze od 1 to bramka NOT lub OR, wartości większe od 2 to bramka AND. Bardzo podobne właściwości wynikają dla współczynnika Blaira-Blissa. Dla bramki AND zmienia się on od 0.2004 do 0.2853, natomiast dla bramek NOT i OR odpowiednio od 0.6515 do 0.6671 oraz od 0.5117 do 0.5901, czyli wynika stąd że wartości współczynników dla bramek NOT i OR są zbliżone do siebie, a jednocześnie różne od bramki AND o ok. 0.3. Ten współczynnik również posłużył do odróżniania bramki AND od bramek NOT i OR. Do identyfikacji w programie przyjęto progi: wartości mniejsze od 0.3 to bramka AND, wartości większe od 0.4 to bramka NOT lub OR. Współczynnik Lp1 umożliwia odróżnienie bramki NOT i OR od siebie. Dla NOT-a jego wartość zmienia się od 0.0944 do 0.1744, a dla OR-a od 0.2947 do 0.4278. Zostały więc przyjęte następujące wartości progowe: wartości mniejsze od 0.2 to NOT, natomiast wartości większe od 0.25 to OR.

2.2.3 Analiza wyników rozpoznawania złożonych bramek

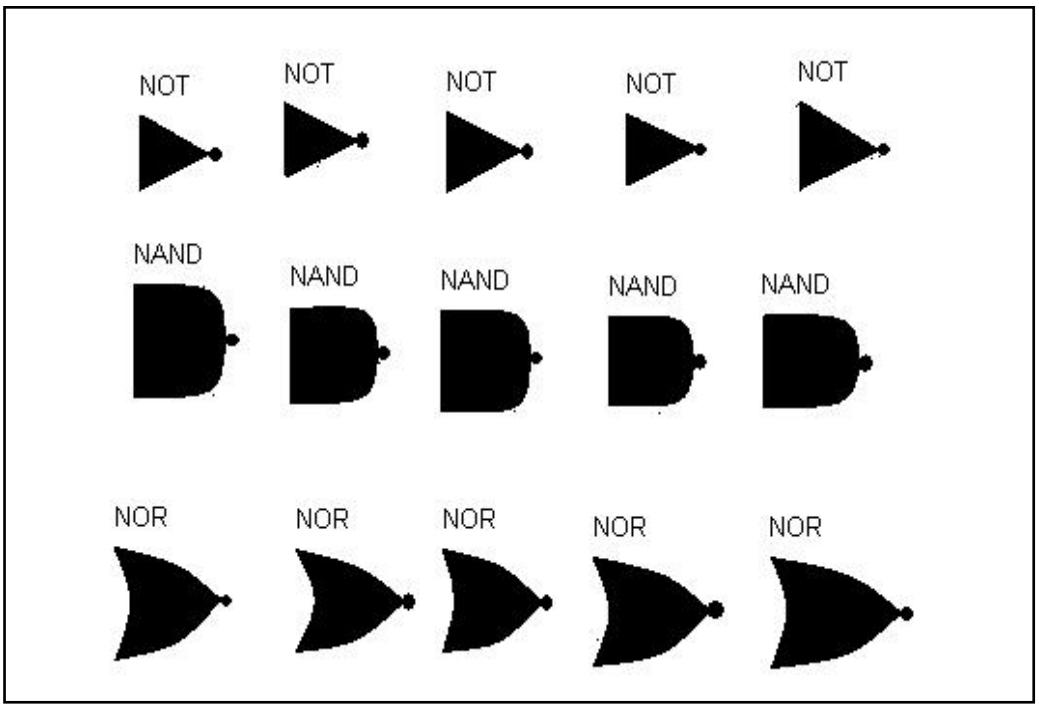
W poprzednim rozdziale dokonano analizy programu pod kątem rozpoznawania pojedynczych bramek. Teraz zastaną opisane przykłady rozpoznawania bramek złożonych, czyli składających się z więcej niż jednej „części". Pod pojęciem pojedynczej „części" rozumiemy elementy, których już nie dzielimy na mniejsze i na których bezpośrednio wykonujemy rozpoznawanie w programie. Do bramek złożonych zaliczamy w pracy bramki NOT (rysowana z kółkiem), NAND, NOR, XOR oraz XNOR. Cztery pierwsze składają się z dwóch „części", natomiast piąta z trzech „części". Rysunek 104 przestawia zbiór bramek NOT, NAND oraz NOR poddanych testowi na rozpoznawanie negacji. Jak wspomniano wcześniej kółko oznaczające negację już na etapie indeksacji zostaje rozpoznane jako osobny obiekt, dodatkowa „część" do pojedynczej bramki, co na rysunku 105 widać w postaci odmiennego koloru kółka w porównaniu do bramki. Rysunek 106 to efekt rozpoznawania bramek. Widać, że program prawidłowo rozpoznał wszystkie bramki.



Rys. 104. Obraz wyjściowy przy teście rozpoznawania negacji.



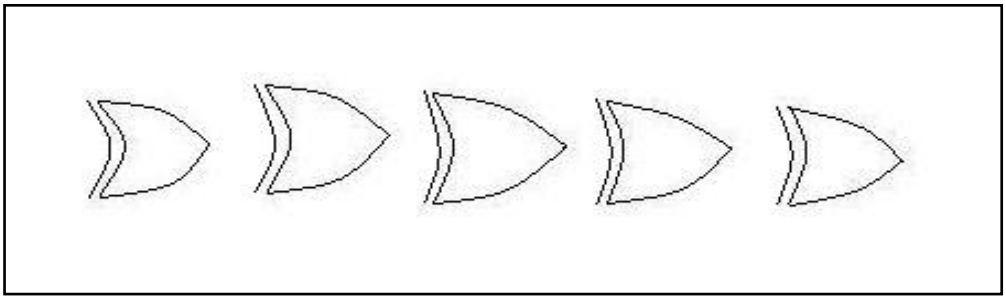
Rys. 105. Efekt indeksacji bramek NOT, NAND oraz OR.



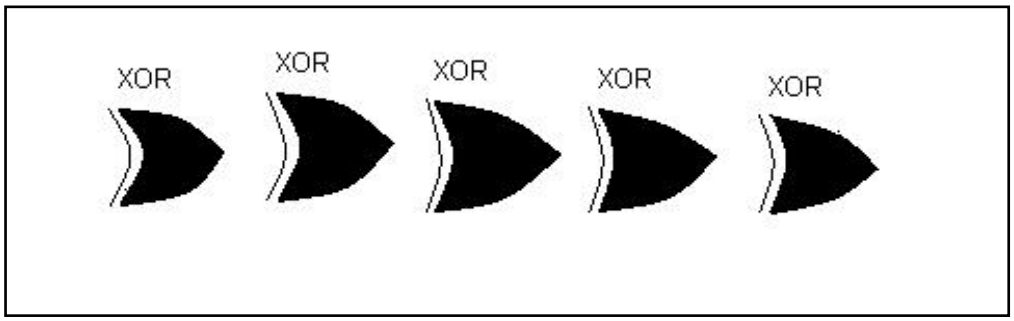
Rys. 106. Obraz końcowy demonstrujący wyniki rozpoznawania.

Oprócz testów dla bramek NOT, NAND oraz NOR przeprowadzono je również dla bramek XOR oraz XNOR, czyli z punktu widzenia rozpoznawania obrazu bramka plus dodatkowo kreska niestykająca się z bramką. Od strony technicznej sposób rozpoznawania tego rodzaju bramki omówiono już wcześniej w rozdziale 2.2.1.

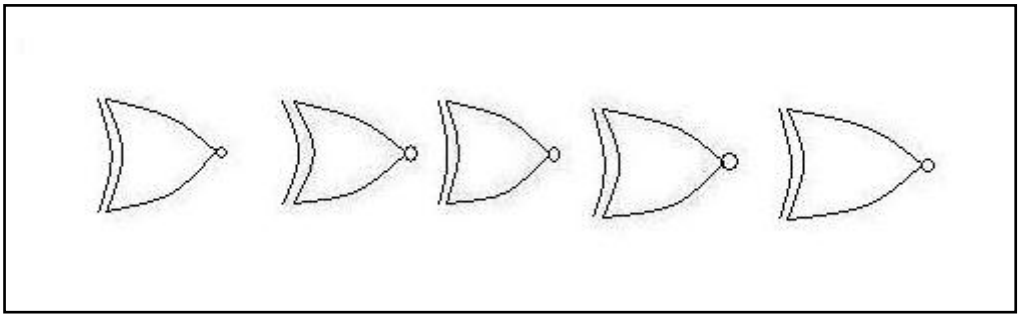
Rysunki 107 oraz 109 to obrazy bramek złożonych XOR oraz XNOR, natomiast rysunki 108 i 110 efekt rozpoznawania w programie *bramki*. Jak widać program prawidłowo rozpoznał wszystkie bramki.



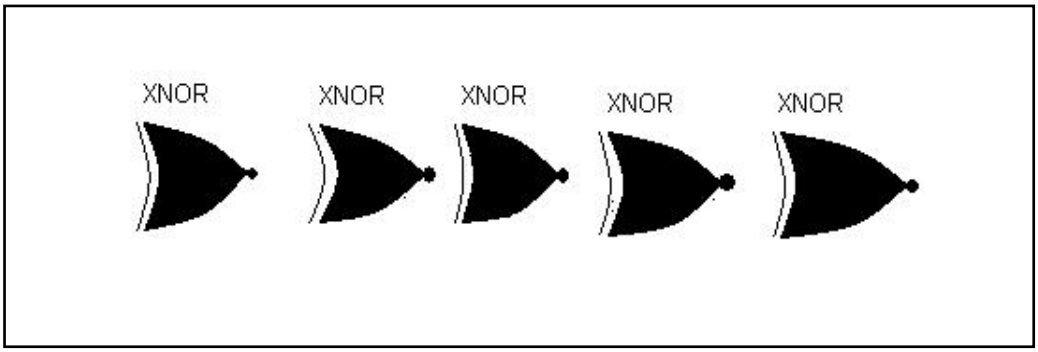
Rys. 107. Obraz wyjściowy przy teście rozpoznawania bramki XOR.



Rys. 108. Obraz końcowy demonstrujący wyniki rozpoznawania bramek XOR.



Rys. 109. Obraz wyjściowy przy teście rozpoznawania bramki XNOR.

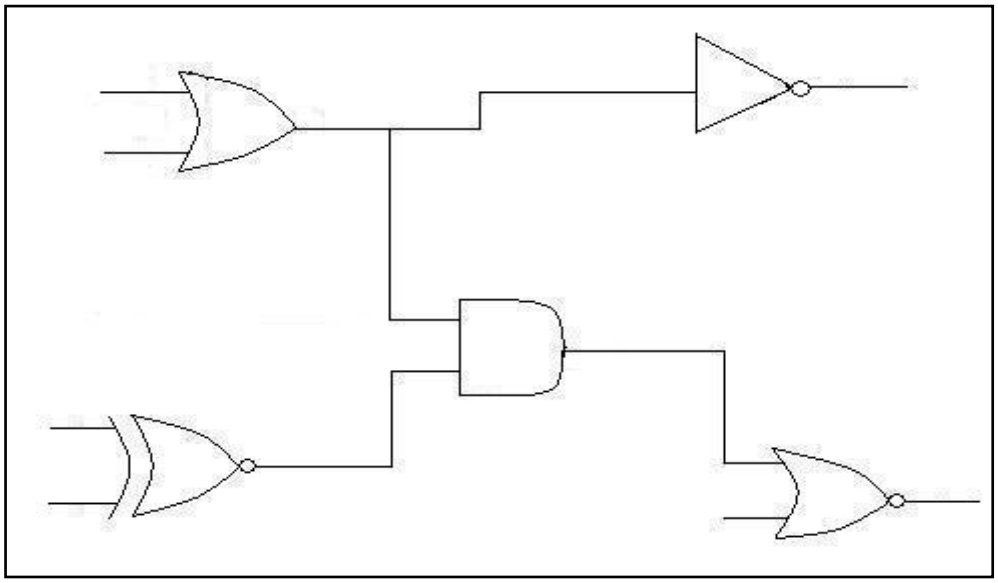


Rys. 110. Obraz końcowy demonstrujący wyniki rozpoznawania bramek XNOR.

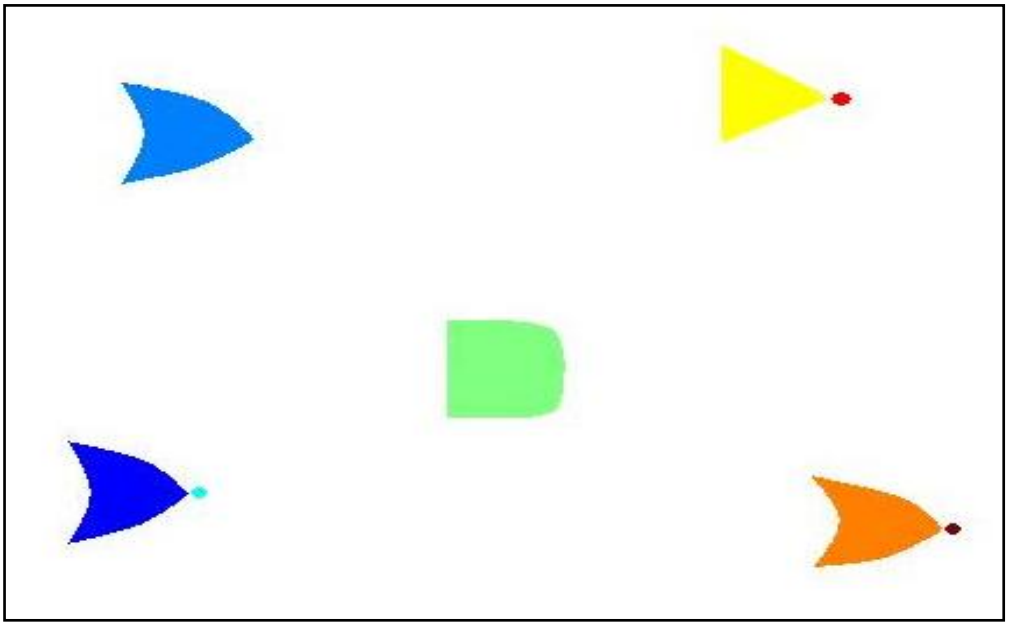
2.2.4 Analiza wyników rozpoznawania złożonych układów

Analizie poddano trzy odmienne schematy. Rysunki 111, 114, 117 pokazują schematy, na których przeprowadzono test, rysunki 112, 116 i 118 to obrazy schematów po operacji indeksacji, natomiast rysunki 113, 115 oraz 119 to obraz wynikowy z etykietami opisującymi rodzaj rozpoznanej bramki. Poniżej każdego obrazu wynikowego znajduje się funkcja opisująca rozpoznany układ, w taki sposób, że jest on bardzo łatwy do zaadoptowania do innych programów takich jak np. Prolog. Program rozpoznał prawidłowo wszystkie bramki, w tym schemat z jedną bramką narysowaną „od góry do dołu".

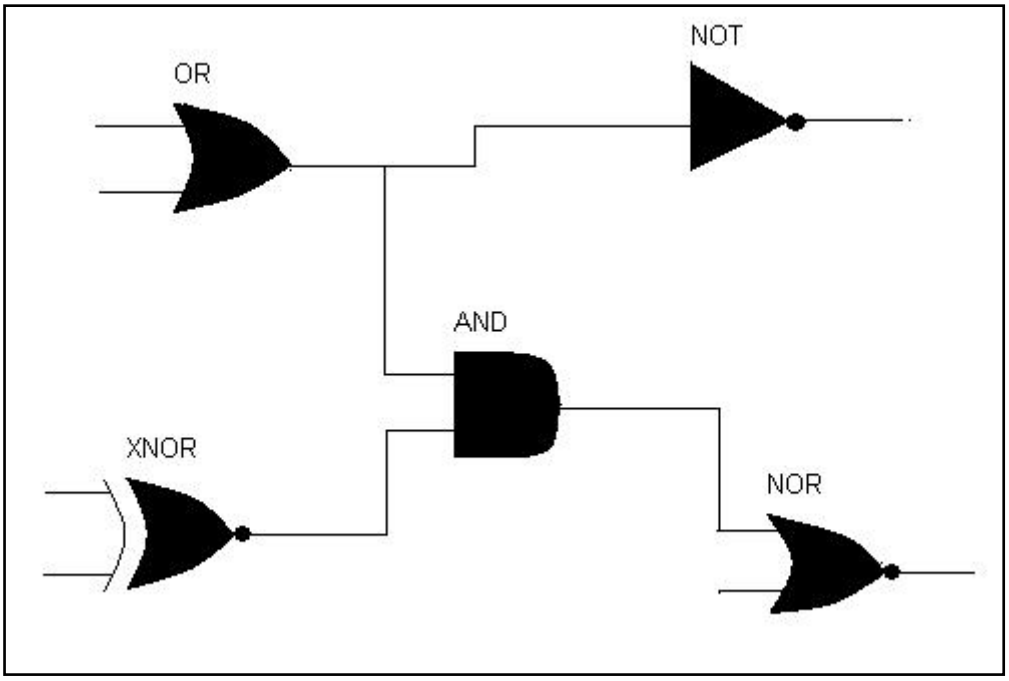
Przykład 1



Rys. 111. Układ wejściowy, który zastał poddany procesowi rozpoznawania.



Rys. 112. Zbiór bramek po operacji indeksacji.

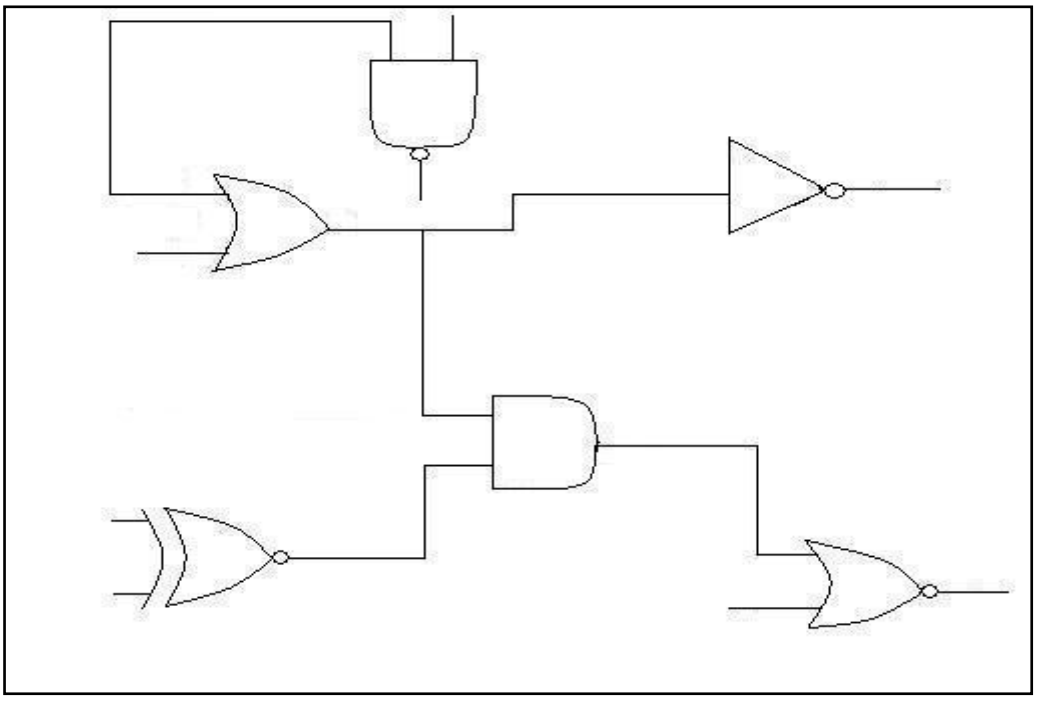


Rys. 113. Obraz wynikowy z etykietami opisującymi rodzaj rozpoznanej bramki.

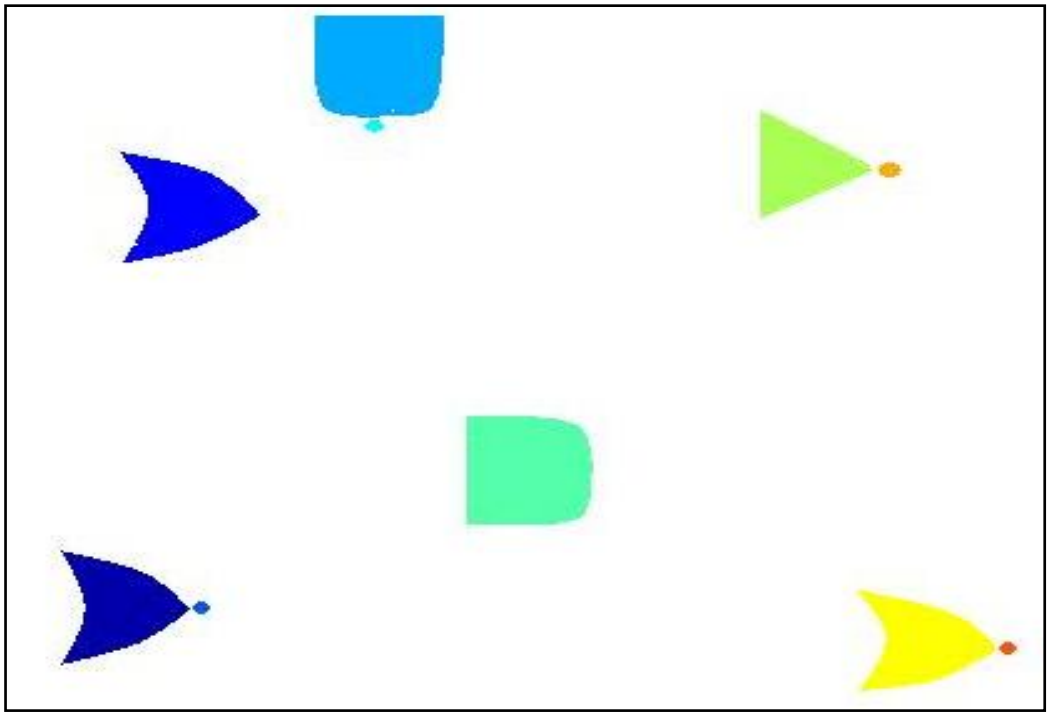
Kod w Prologu wygenerowany za pomocą programu *bramki*:

Układ_*przykład1*(A, B, C, D, E, F, G, I, K):-br XNOR(B, A, E), br OR(C, D, F), br AND(F, E, G), br NOT(F, I), br NOR(G, H, K)

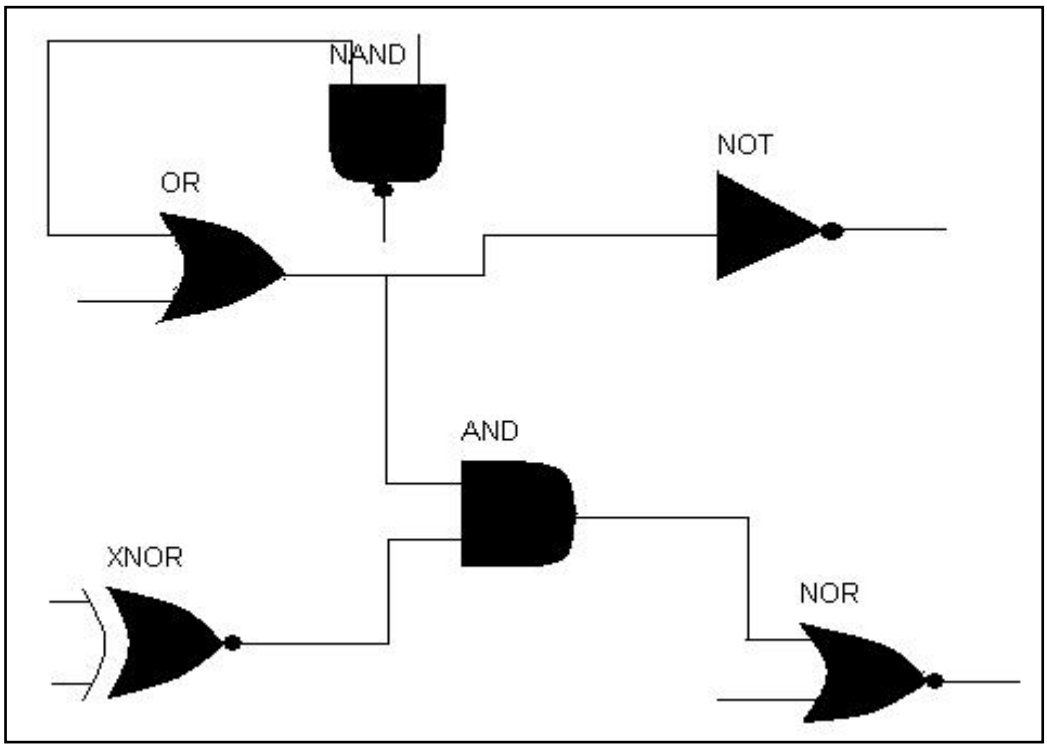
Przykład 2



Rys. 114. Układ wejściowy, który zastał poddany procesowi rozpoznawania.



Rys. 115. Zbiór bramek po operacji indeksacji.

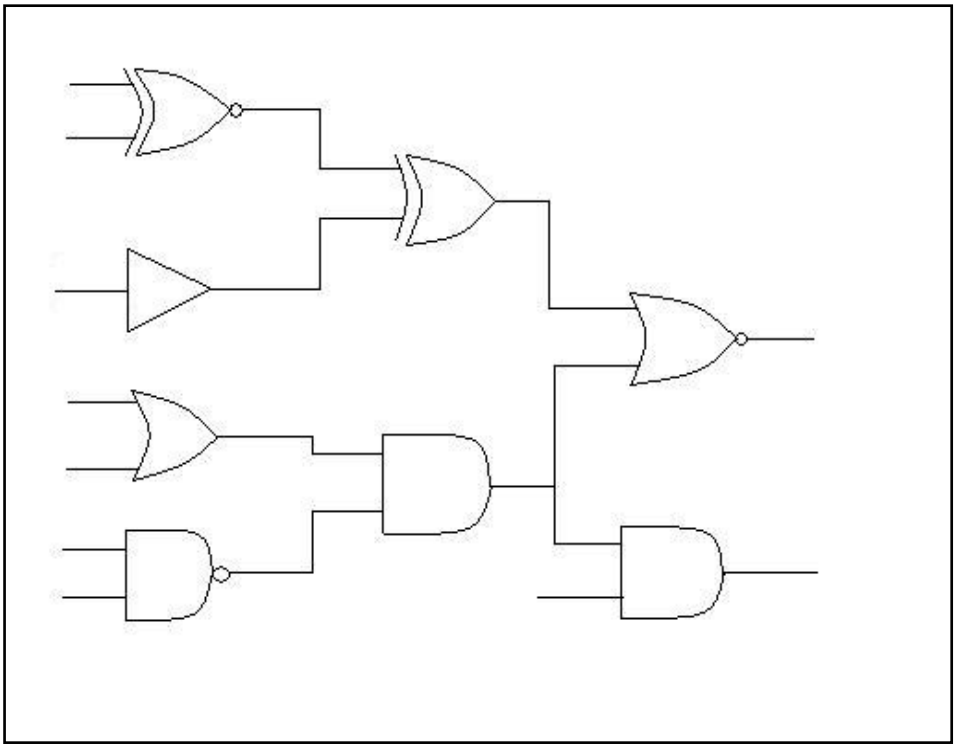


Rys. 116. Obraz wynikowy z etykietami opisującymi rodzaj rozpoznanej bramki.

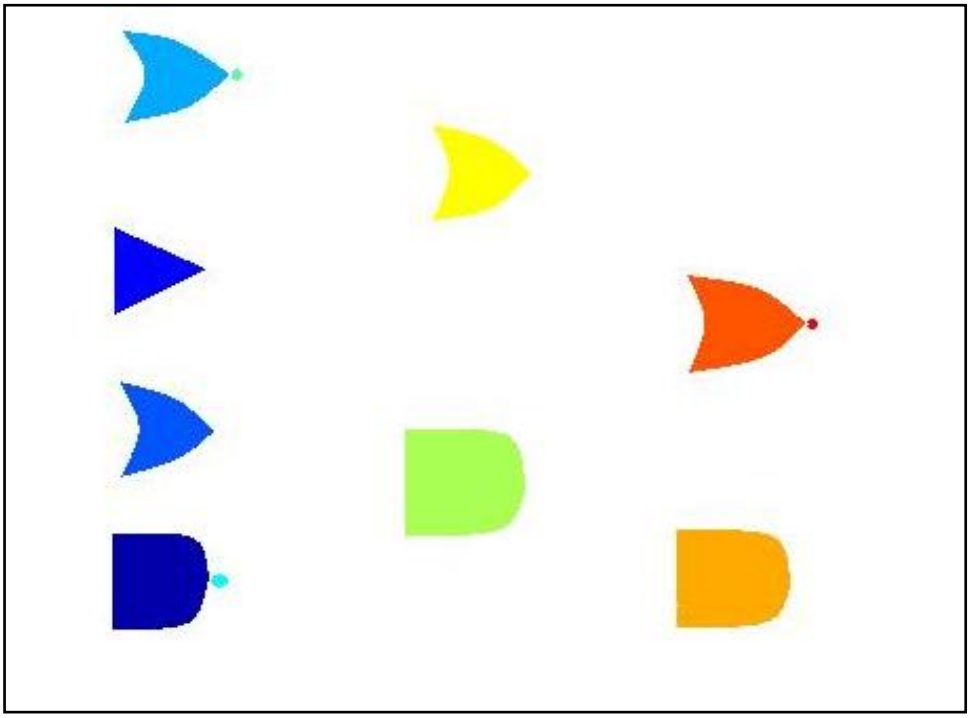
Kod w Prologu wygenerowany za pomocą programu *bramki*:

Układ*_przykład2*(A, B, C, D, E, F, G, I, J, K, L, M):-br XNOR(B, C, E), br OR(A, D, F), br NAND(G, F, I), br AND(F, E, J), br NOT(I, L), br NOR(J, K, M)

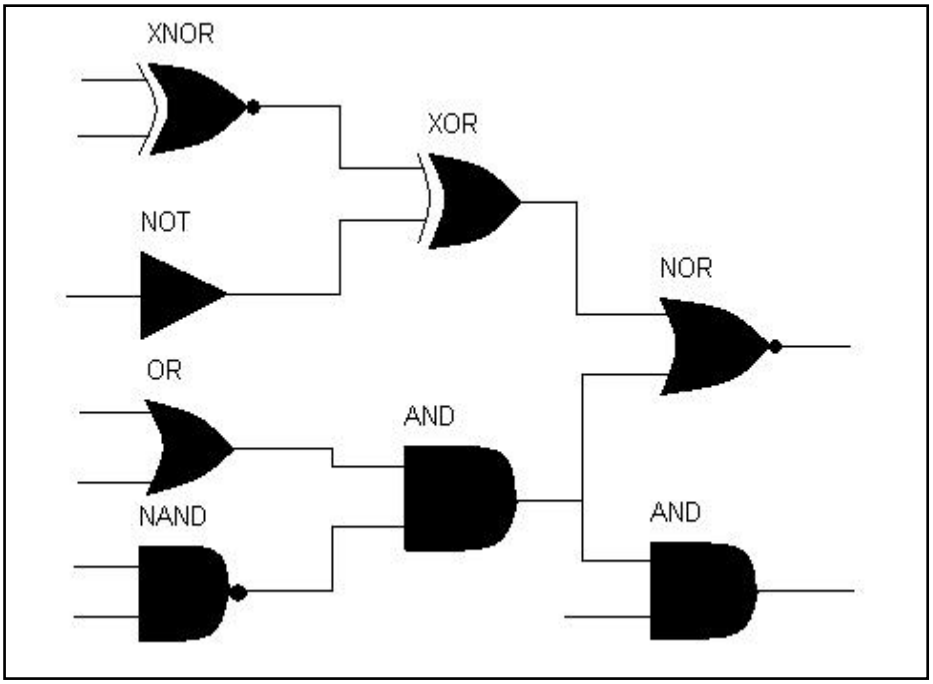
Przykład 3



Rys. 117. Układ wejściowy, który zastał poddany procesowi rozpoznawania.



Rys. 118. Zbiór bramek po operacji indeksacji.



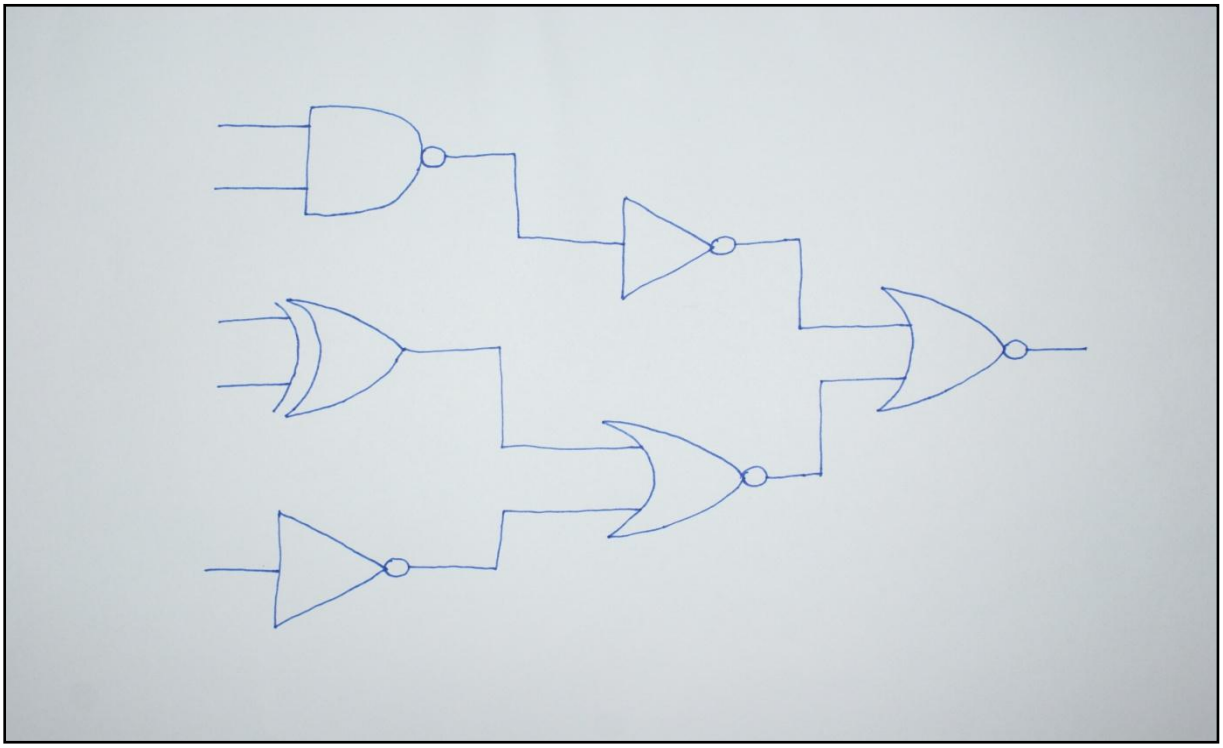
Rys. 119. Obraz wynikowy z etykietami opisującymi rodzaj rozpoznanej bramki.

Kod w Prologu wygenerowany za pomocą programu *bramki*:

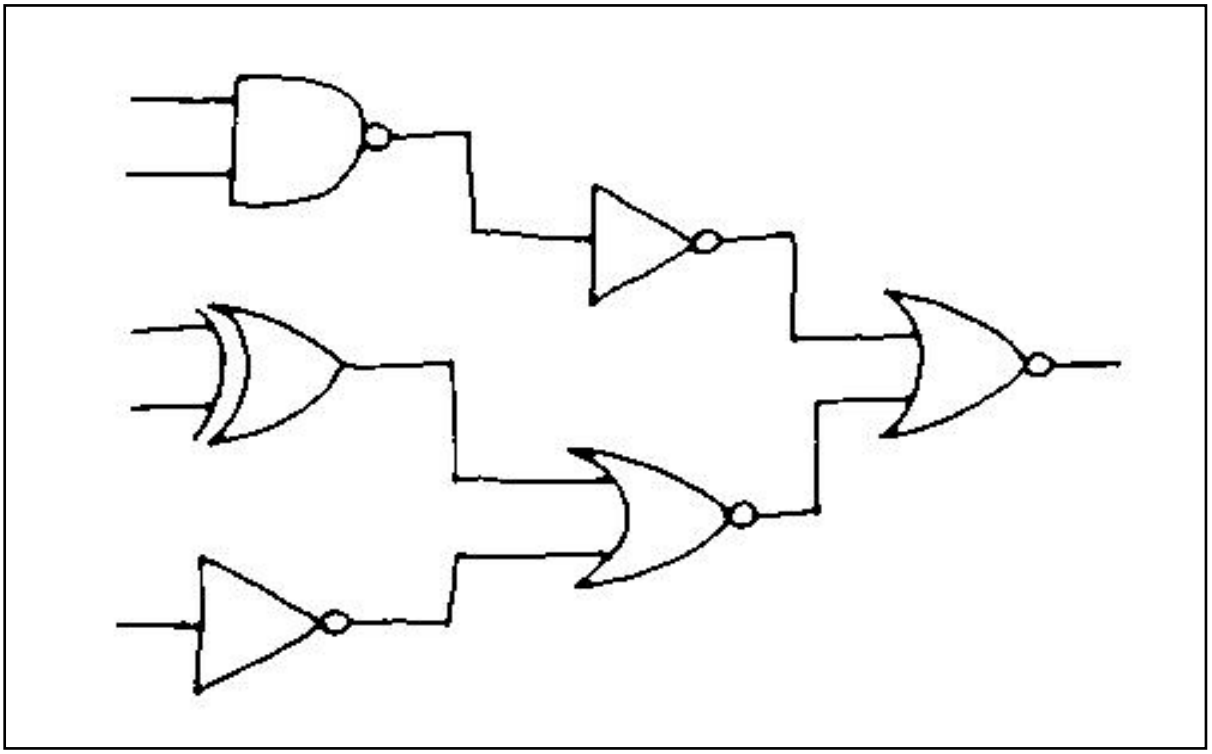
Układ_*przykład3*(A, B, C, D, E, F, G, H, I, J, K, L, M, N, P):- brNAND(B, C, J), brNOT(A, H), brOR(F, E, I), brXNOR(G, D, K), brAND(I, J, L), brXOR(K, H, M), brAND(L, N, O), br NOR(M, L, P)

2.3 Przykłady rozpoznawania odręcznych schematów układów logicznych

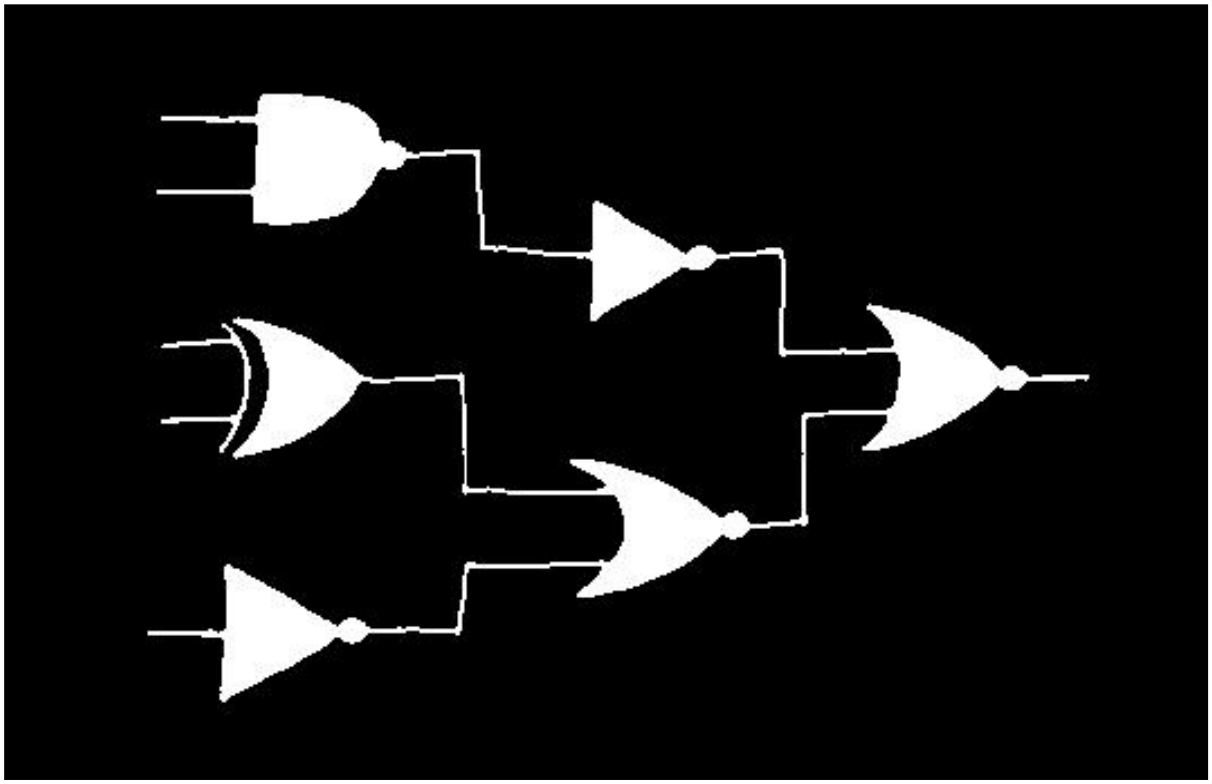
Przykład 4



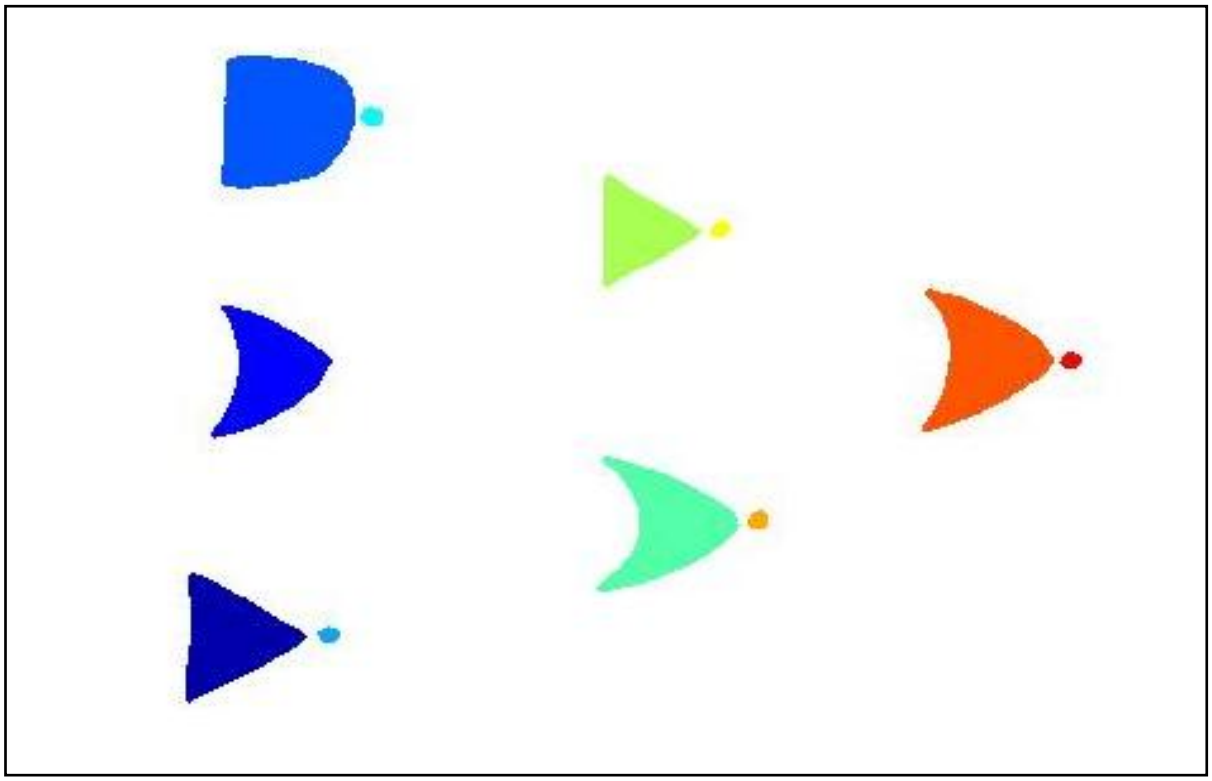
Rys. 120. Oryginalny obraz *przykład 4*, który został poddany procesowi rozpoznawania, zawierający narysowany odręcznie schemat.



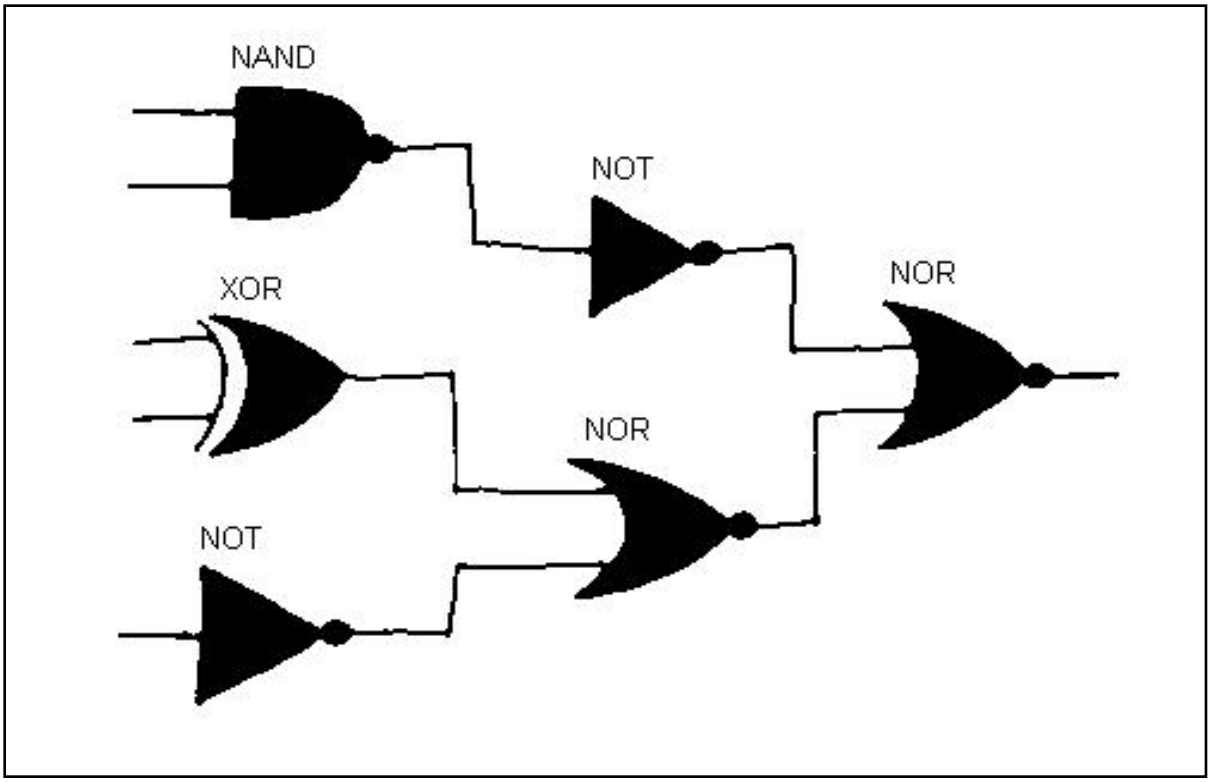
Rys. 121. Obraz *przykład 4* po procesie segmentacji.



Rys. 122. Obraz *przykład 4* po operacji zalewania otworów.



Rys. 123. Obraz *przykład 4* po operacji indeksacji.



Rys. 124. Obraz prezentujący wynik rozpoznawania obrazu *przykład 4*.

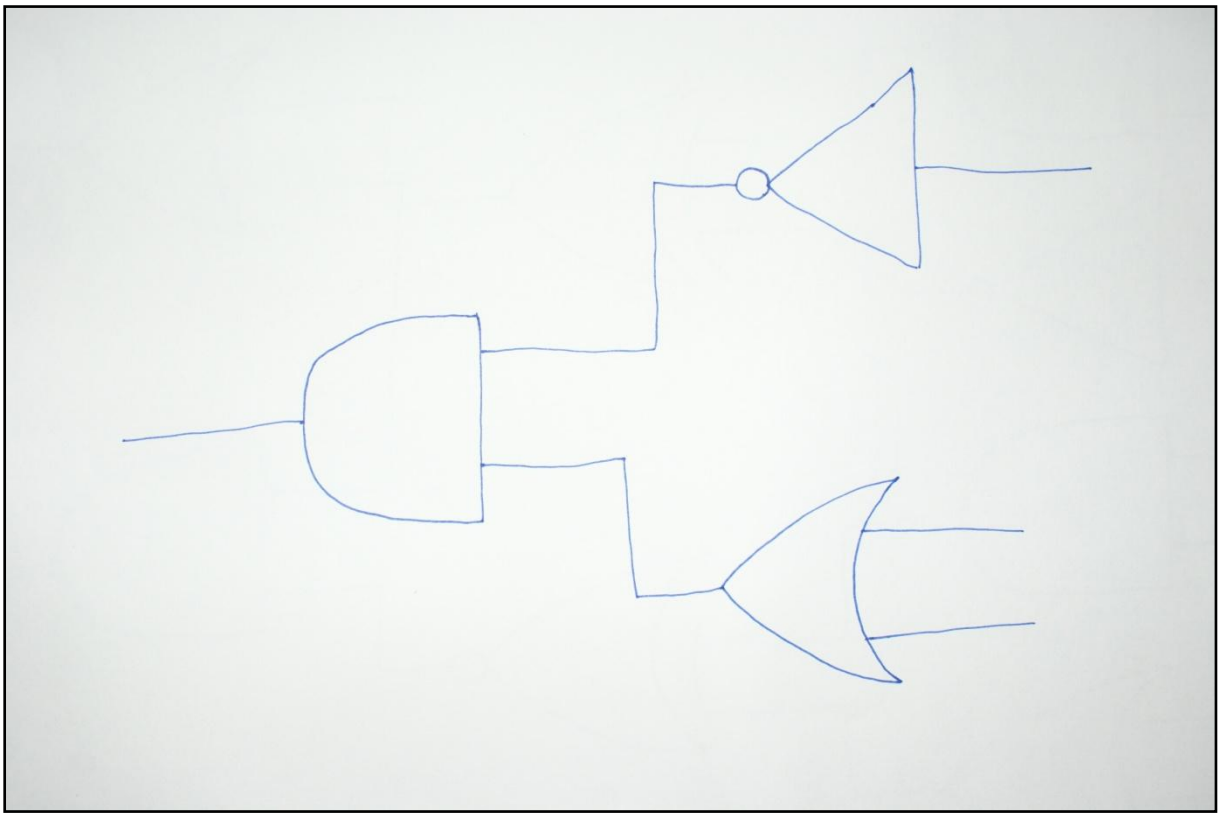
Tabela 9. Zestawienie wyników rozpoznawania odręcznego schematu *przykład 4* w programie bramki.

WBB=0.6172	WBB=0.5806	WBB=0.3325	WBB=0.6172	WBB=0.6245	WBB=0.5831
WM=0.4800	WM=0.5251	WM=1.4570	WM=0.5159	WM=0.4696	WM=0.5448
M1=0.4178	M1=0.4721	M1=1.4397	M1=0.4178	M1=0.4081	M1=0.468
M2=0.1746	M2=0.2236	M2=2.0918	M2=0.1746	M2=0.1666	M2=0.2193
M7=0.0405	M7=0.0555	M7=0.5010	M7=0.0435	M7=0.0385	M7=0.0547
Lp1=0.2055	Lp1=0.4024	Lp1=0.3210	Lp1=0.2861	Lp1=0.2118	Lp1=0.3765

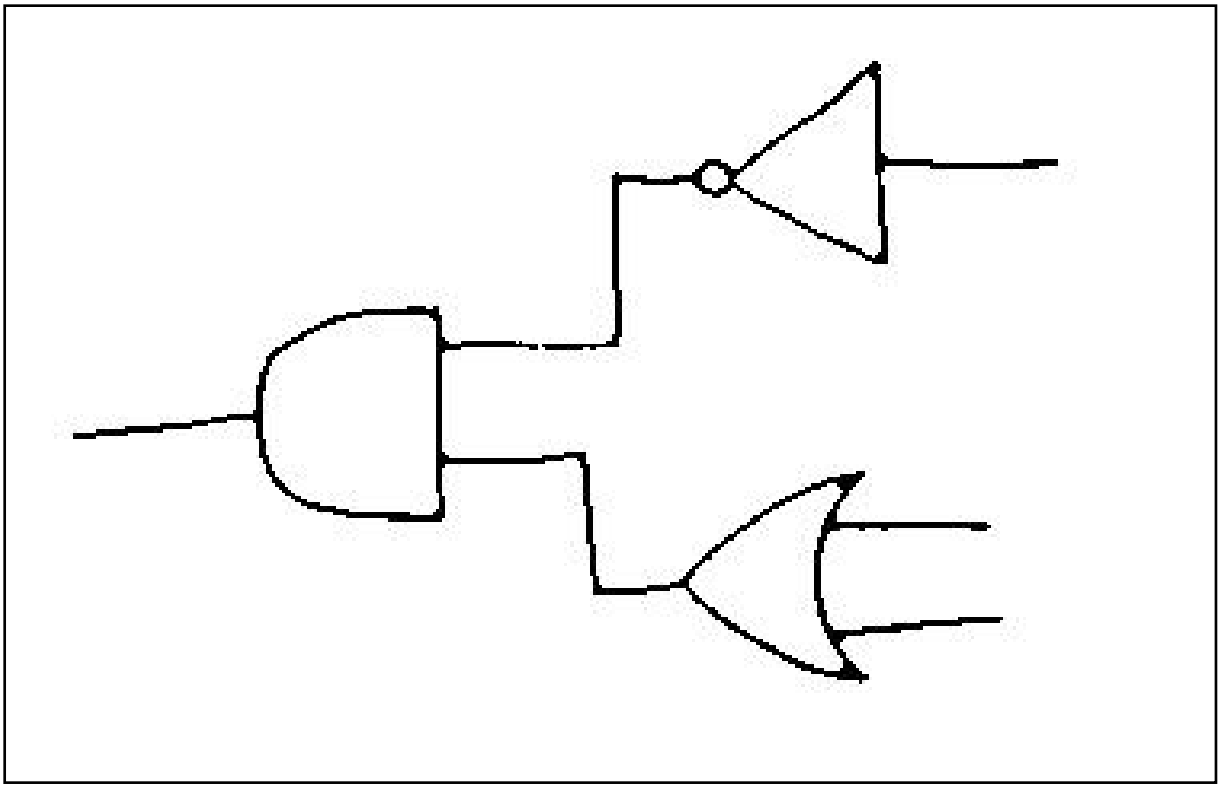
Kod w Prologu wygenerowany za pomocą programu *bramki*:

Układ_*przykład 4*(A,B,C,D,E,F,G,H,I,J,K):- brNOT(A,G), brXOR (D,E,F), brNAND (C,B, H), brNOR (F,G,J), brNOT(H,I), brNOR(I,J,K)

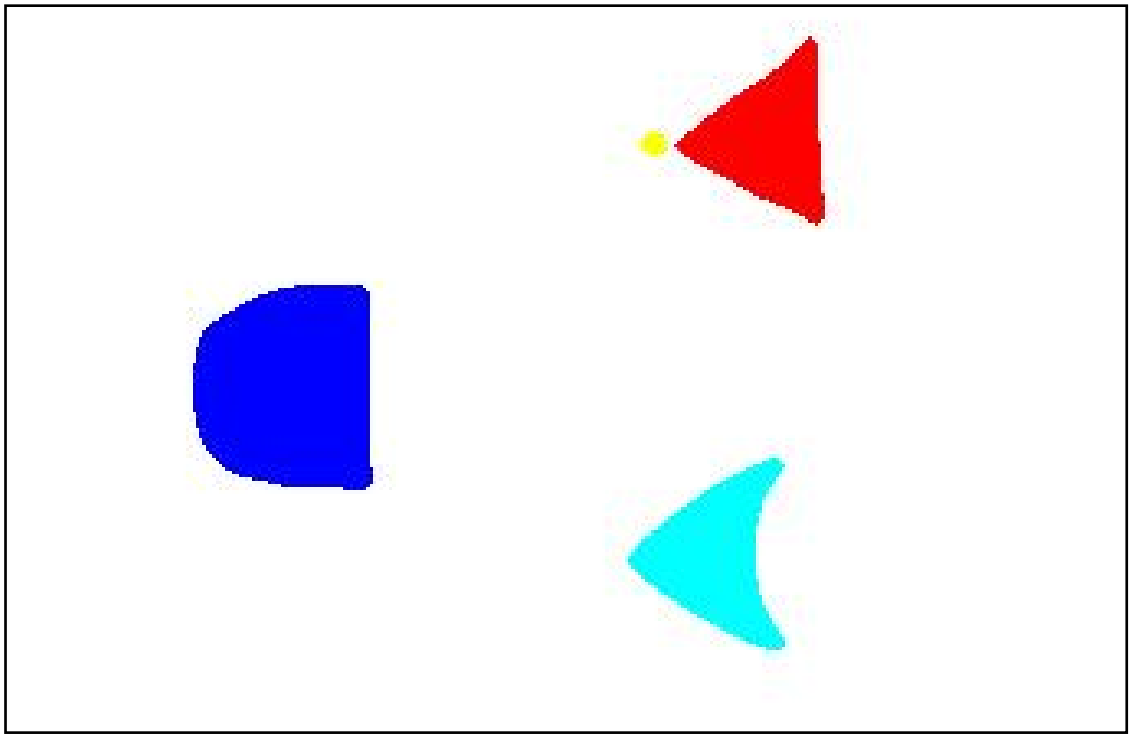
Przykład 5



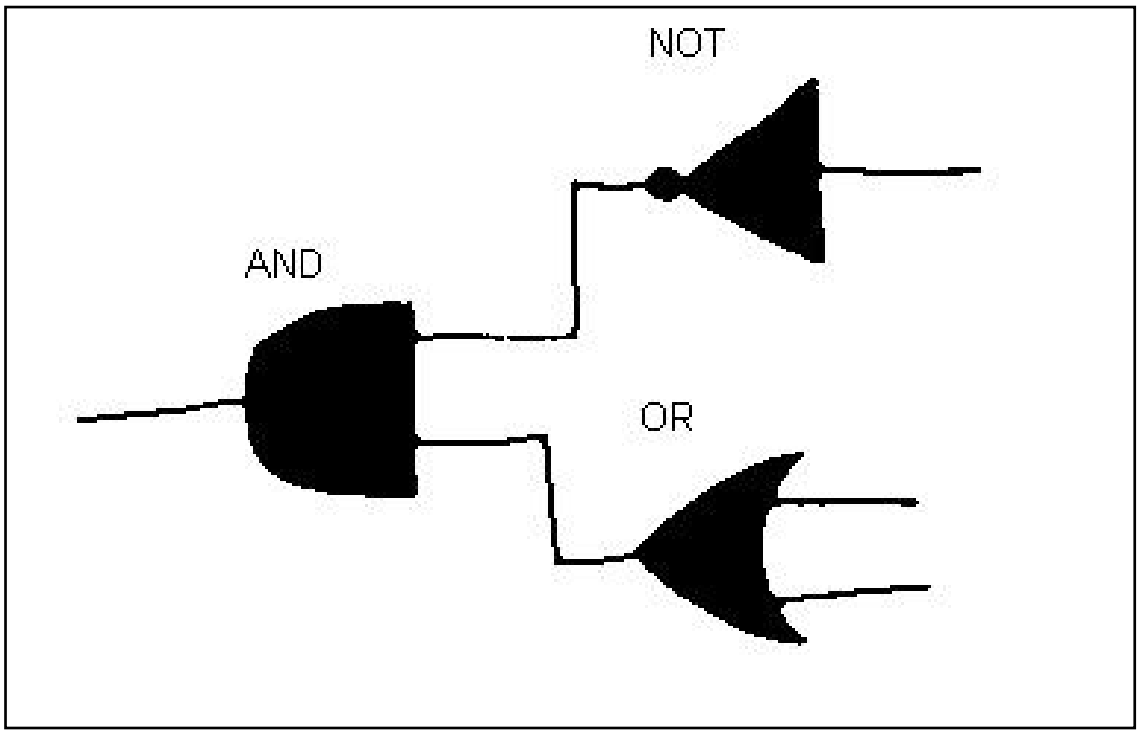
Rys. 125. Oryginalny obraz *przykład 5*, który został poddany procesowi rozpoznawania, zawierający narysowany odręcznie schemat.



Rys. 126. Obraz *przykład 5* po procesie segmentacji.



Rys. 127. Obraz *przykład 5* po operacji indeksacji.



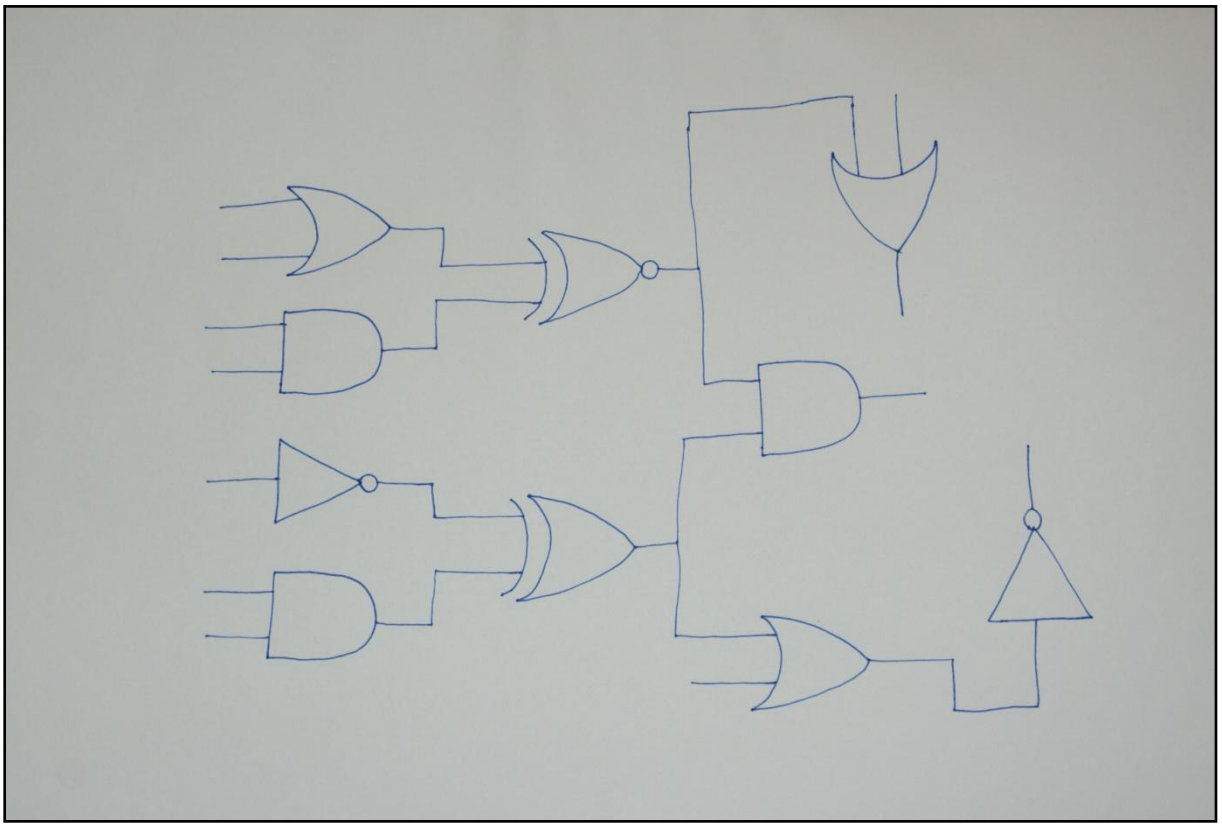
Rys. 128. Obraz prezentujący wynik rozpoznawania obrazu *przykład 5*.

Układ_*przykład5*(A,B,C,D,E,F):- brAND(B,C,A), brOR(E,F,C),
brNOT(G,B).

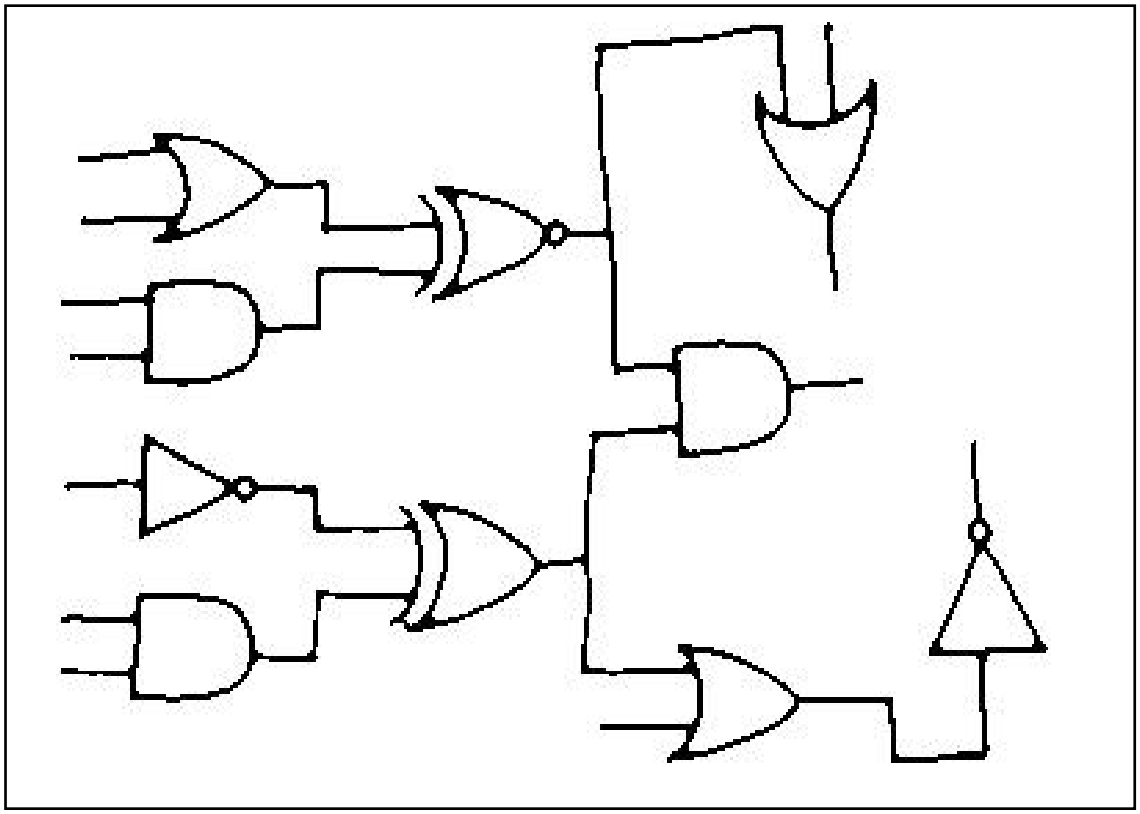
Tabela 10. Zestawienie wyników rozpoznawania odręcznego schematu *przykład 5* w programie bramki.

WBB = 0.3024	WBB = 0.5921	WBB = 0.6279
WM = 1.6567	WM = 0.5397	WM = 0.4243
M1 = 1.7400	M1 = 0.4539	M1 = 0.4037
M2 = 3.0283	M2 = 0.2061	M2 = 0.1637
M7 = 0.7287	M7 = 0.0515	M7 = 0.0384
Lp1= 0.3558	Lp1 = 0.3163	Lp1 = 0.1259

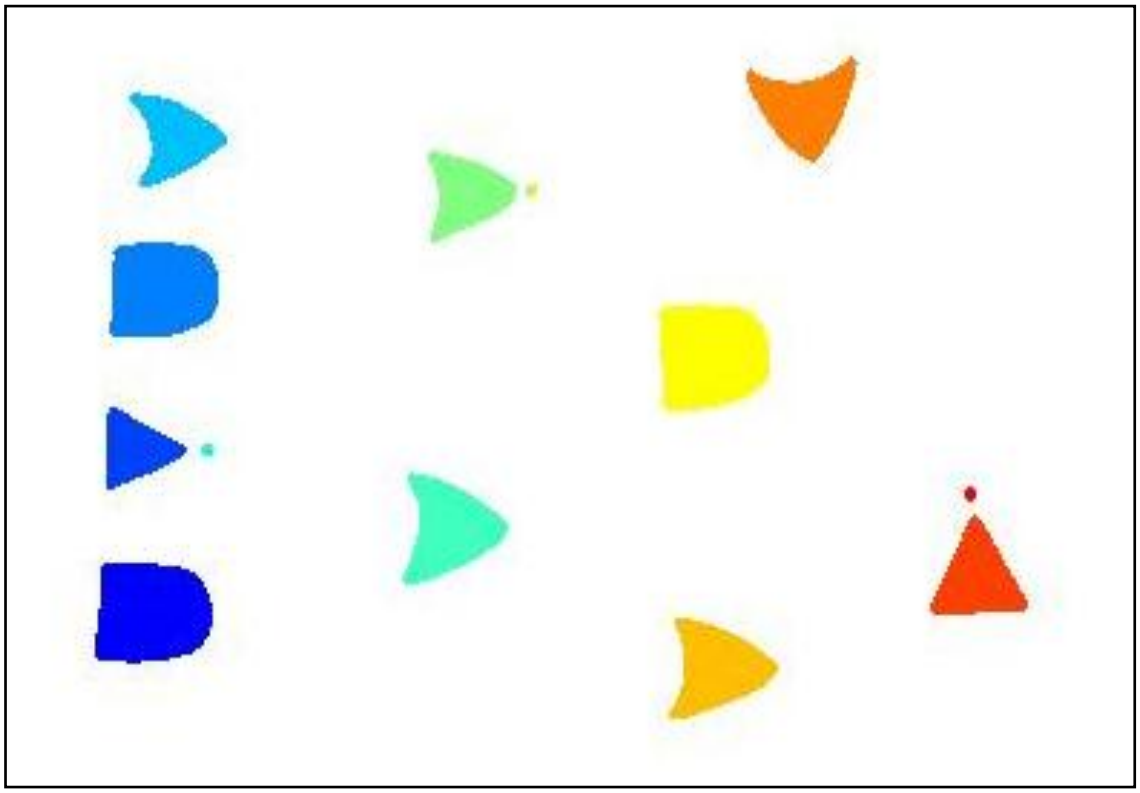
Przykład 6



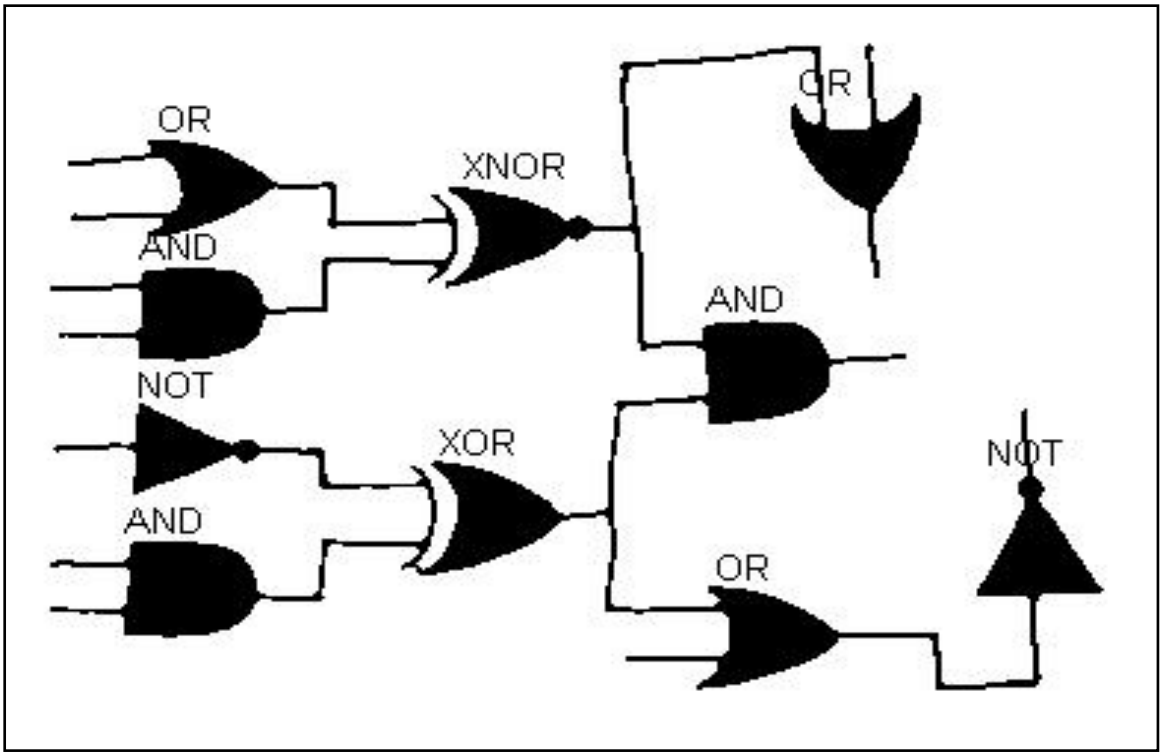
Rys. 129. Oryginalny obraz *przykład 6*, który został poddany procesowi rozpoznawania, zawierający narysowany odręcznie schemat.



Rys. 130. Obraz *przykład 6* po procesie segmentacji.



Rys. 131. Obraz *przykład 6* po operacji indeksacji.



Rys. 132. Obraz prezentujący wynik rozpoznawania obrazu *przykład 6*.

Kod w Prologu wygenerowany za pomocą programu *bramki*:

Układ_*przykład6*(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S):- brAND(A,B,I), brNOT(C,H), brAND(D,E,J), brOR(F,G,K), brXOR(H,I,L), brXNOR(K,J,M), brAND(L,M,P), brOR(L,N,O), brOR(M,Q,R), brNOT(O,S).

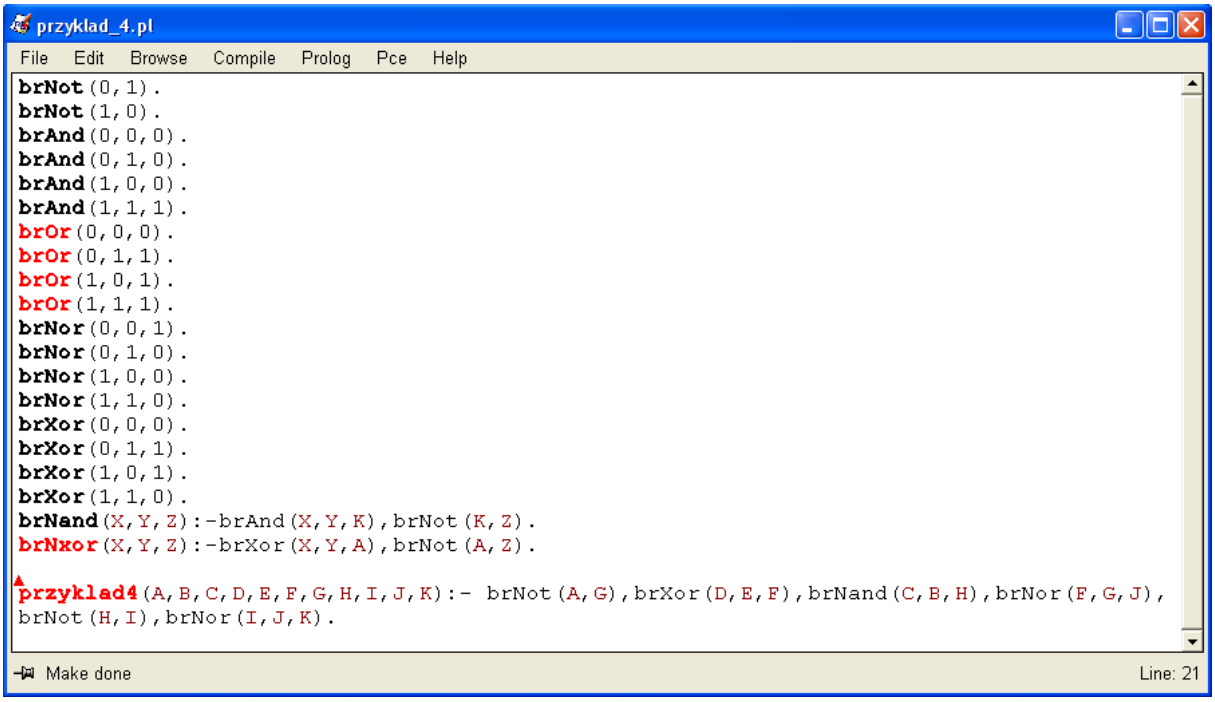
Tabela 11. Zestawienie wyników rozpoznawania odręcznego schematu przykład 6 w programie bramki.

WBB= 0.2884	WBB= 0.5966	WBB= 0.2566	WBB= 0.5678	WBB= 0.5429	WBB= 0.5330	WBB= 0.3257	WBB= 0.5265	WBB= 0.5803	WBB= 0.6116
WM = 1.7237	WM = 0.6387	WM = 2.0296	WM = 0.6336	WM = 0.6610	WM = 0.7267	WM = 1.4631	WM = 0.6995	WM = 0.5696	WM = 0.5680
M1 = 1.9137	M1 = 0.4471	M1 = 2.4168	M1 = 0.4936	M1 = 0.5400	M1 = 0.5603	M1 = 1.5005	M1 = 0.5741	M1 = 0.4727	M1 = 0.4255
M2 = 3.6713	M2 = 0.1999	M2 = 6.0970	M2 = 0.2445	M2 = 0.2926	M2 = 0.3141	M2 = 2.2516	M2 = 0.3299	M2 = 0.2236	M2 = 0.1814
M7 = 0.9106	M7 = 0.0418	M7 = 1.3959	M7 = 0.0606	M7 = 0.0726	M7 = 0.0778	M7 = 0.5603	M7 = 0.0822	M7 = 0.0557	M7 = 0.0402
Lp1 = 0.4766	Lp1 = 0.1614	Lp1 = 0.4856	Lp1 = 0.3778	Lp1 = 0.3473	Lp1 = 0.2857	Lp1 = 0.3986	Lp1 = 0.3723	Lp1 = 0.3653	Lp1 = 0.1637

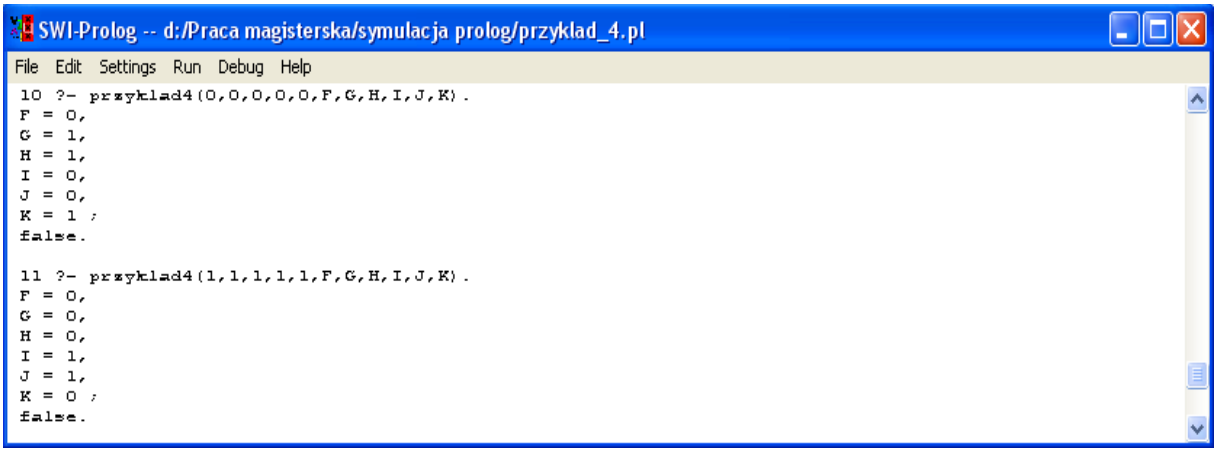
2.4 Wykorzystanie programu bramki do analizy działania układów logicznych

Stworzony w ramach niniejszej pracy program pozwala na analizę działania układów logicznych dzięki kompatybilności z innym programem. Automatycznie wygenerowany kod w programie *bramki* można wprowadzić do programu Prolog w celu uzyskania np. wszystkich kombinacji wyjść w zależności od wejść. Poniżej pokazano realizację kilku przykładów rozpoznanych w programie *bramki*, a następnie przetestowanych w *Prologu*.

- Kod z obrazu *Przykład 4* w Prologu

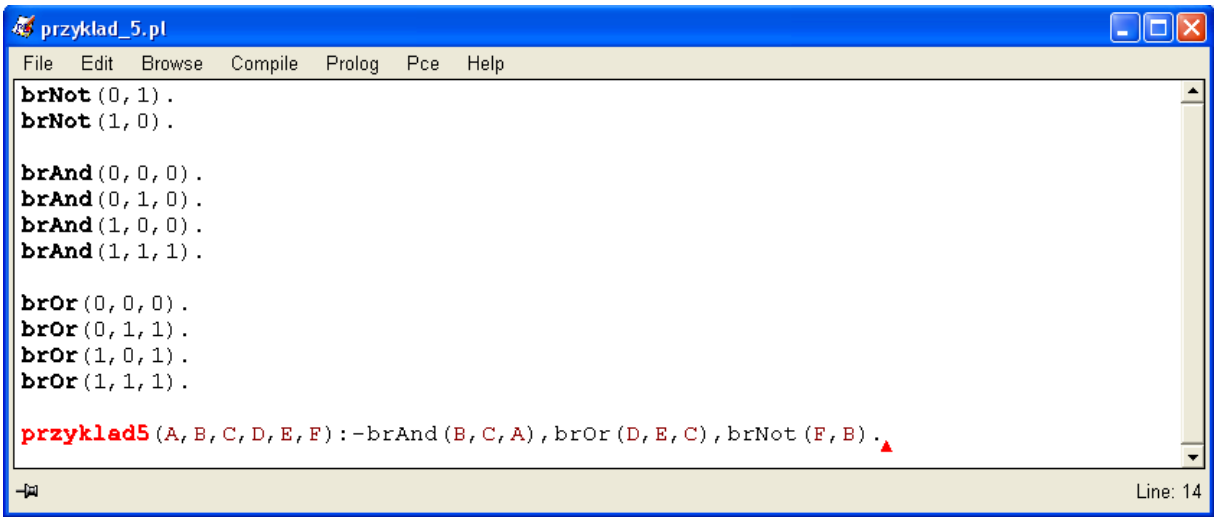


Rys. 133. Kod źródłowy w Prologu do obrazu *Przykład 4*.

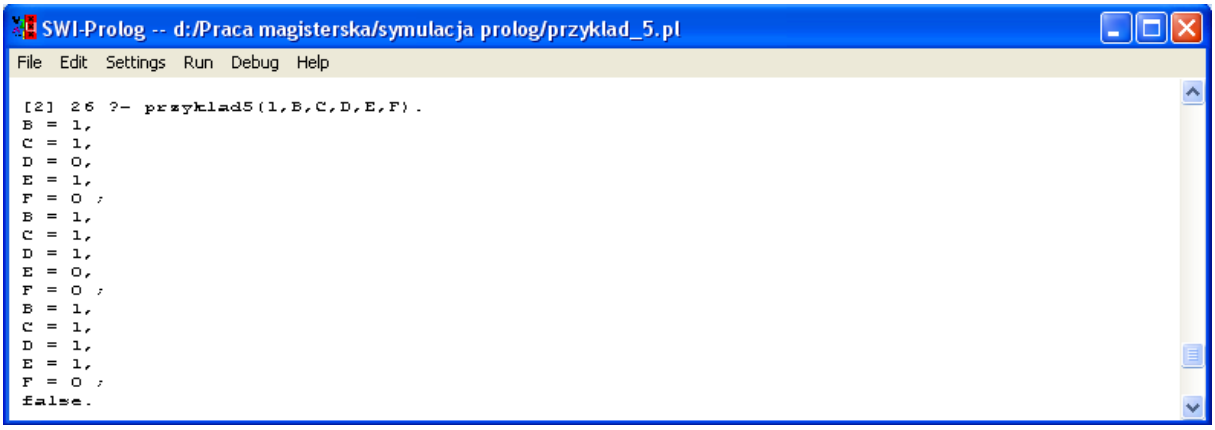


Rys. 134. Wynik symulacji dla układu z obrazu *Przykład 4*.

- Kod z obrazu *Przykład 5* w Prologu

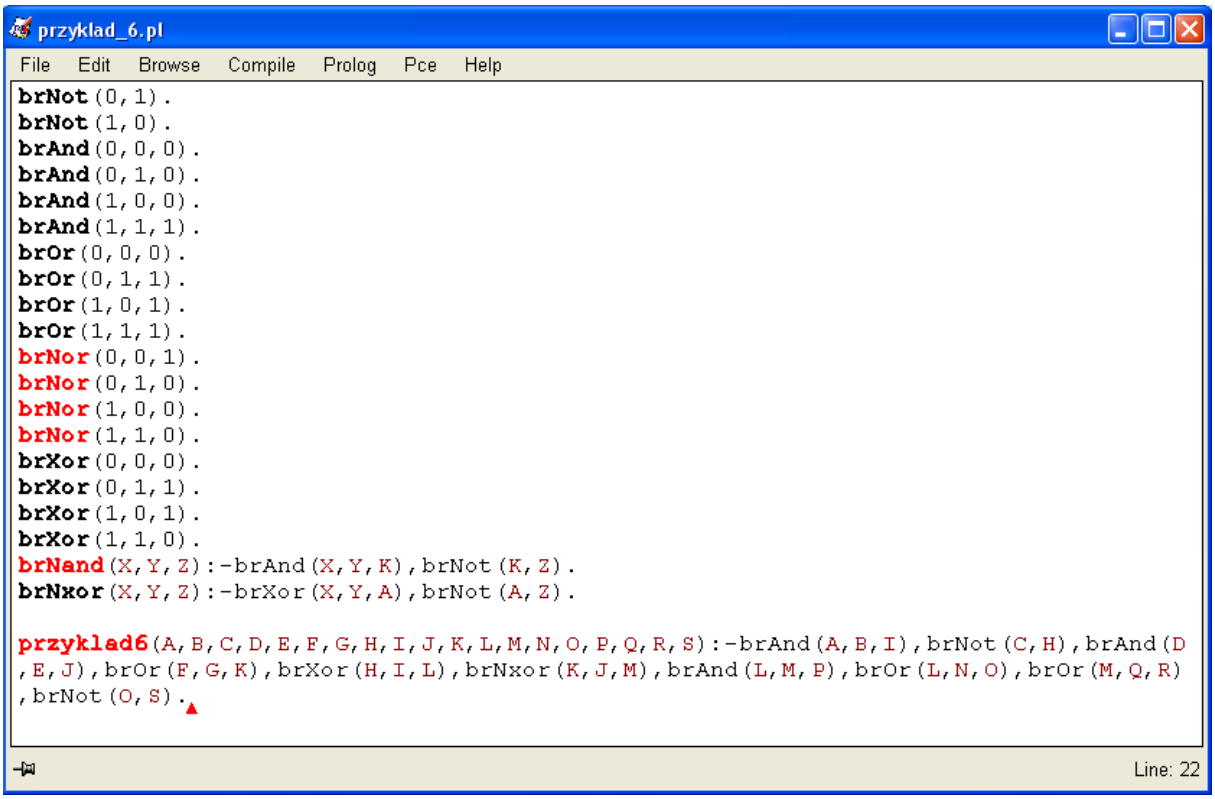


Rys. 135. Kod źródłowy w Prologu do obrazu *Przykład 5*.

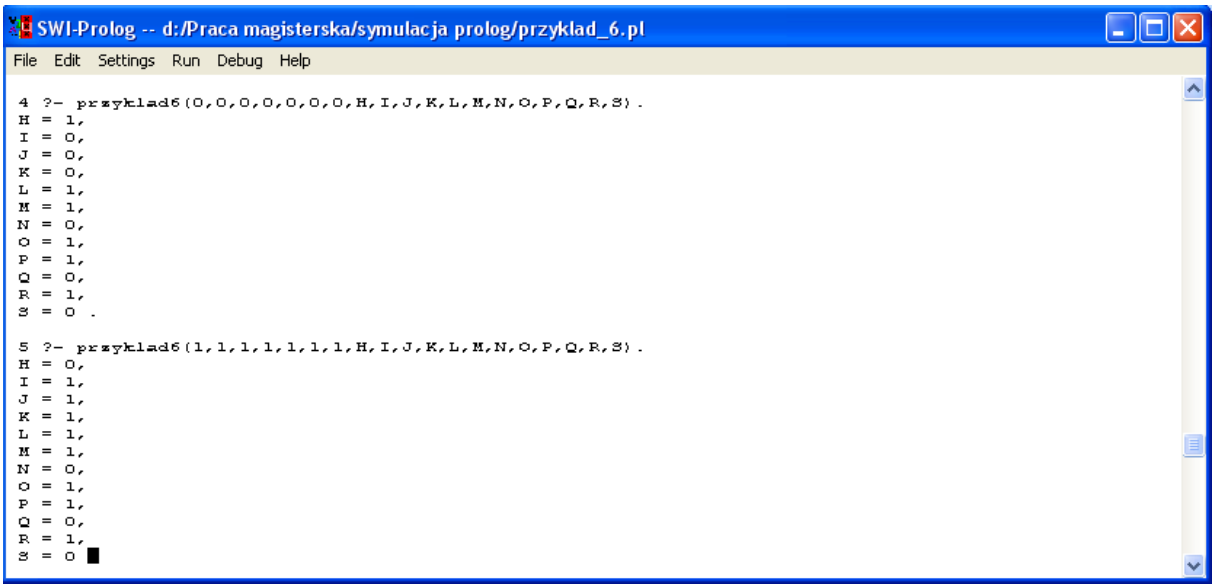


Rys. 136. Wynik symulacji dla układu z obrazu *Przykład 5*.

- Kod z obrazu *Przykład 6* w Prologu



Rys. 137. Kod źródłowy w Prologu do obrazu *Przykład 6*.



Rys. 138. Wynik symulacji dla układu z obrazu *Przykład 6*.

2.5 Wnioski

Na podstawie pierwszej części testów, opisanej w rozdziale 2.2.1 można ocenić przydatność zastosowanych współczynników kształtu oraz metod identyfikacji w procesie rozpoznawania obrazów.

Obliczenia i symulacje wykonane dla bramek wzorcowych pozwoliły na wstępne dobranie progów, dla jakich można było wykonywać identyfikację, a testy przeprowadzone na seriach podobnych bramek pozwoliły na ocenę zachowania się współczynników kształtu przy zmianie skali oraz proporcji w rysowaniu bramek.

Wartości współczynników zarówno dla bramek wzorcowych jak i dla serii bramek testowych są bardzo zbliżone. Te niewielkie różnice dowodzą, że zastosowane metody rozpoznawania bramek logicznych są odpowiednie i dają dużą pewność wyniku. Potwierdzają również trafność doboru współczynników kształtu, które zostały zastosowane. Wynika stąd, że program może zostać wykorzystywany do generowania modeli symulacyjnych układów logicznych na podstawie schematów drukowanych i odręcznych.

Testy pokazują, że najmniej problemu sprawia rozpoznawanie bramki AND, gdyż wartości współczynników kształtu Malinowskiej oraz Blaira-Blissa dla tej bramki różnią się znacznie od wartości dla bramek NOT czy OR. Współczynnik Malinowskiej dla bramek AND wynosi średnio 2.9523 czyli różni się o ok. 2.5 od współczynników dla bramek NOT (0.3235) oraz bramki OR (0.5682). Podobne właściwości wykazuje współczynnik Blaira-Blissa, gdyż dla bramki AND wynosi średnio 0.2210 natomiast dla bramek NOT oraz OR odpowiednio 0.6602 i 0.5243. Różnią się one więc jedynie ok. 0.4 od AND-a.

Do rozróżniania bramki OR od NOT zaistniała konieczność wprowadzenia dodatkowego współczynnika Lp1, który daje dostatecznie pewną informację do tego, aby stwierdzić poprawnie rodzaj rozpoznanej bramki. Dla bramki NOT średnia wartość współczynnika Lp1 wynosi 0.1725 natomiast dla bramki OR 0.3861.

W celu rozpoznawania negacji została wprowadzona funkcja, która dodatkowo indeksuje obiekty znajdujące się w prostokącie opisującym bramkę. Przeprowadzone testy dowodzą poprawności tego rozwiązania, gdyż statystycznie bardzo mało było przypadków nie wykrycia symbolu negacji w bramce.

Również metoda rozpoznawania symbolu bramki XOR polegająca na kolejnym przeszukiwaniu otoczeń działa poprawnie i pozwala niemal za każdym razem dobrze rozpoznać bramkę.

Ostatni etap testu programu *bramki* polegał na rozpoznawaniu odręcznie narysowanych schematów. Przykłady pokazują, że pomimo różnej jakości zdjęć, segmentacja pozwala prawidłowo wyodrębnić schemat ze zdjęcia. To również świadczy o poprawności zastosowanych algorytmów oraz metod filtracji i obróbki obrazu. Dodatkową zaletą programu jest jego duża odporność na zakłócenia polegająca na niewrażliwości na obroty bramek o niewielkie kąty oraz na brak idealnej symetrii. Algorytm umożliwia rozpoznawanie bramek obróconych o 90, 180 lub 270 stopni.

3 Możliwości praktycznych zastosowań opracowanych algorytmów oraz propozycje dalszych badań

3.1 Możliwości praktycznych zastosowań

- **Wprowadzanie schematów do programów do analizy i symulacji**

Program z powodzeniem może posłużyć do wprowadzania schematów do programów do analizy i symulacji układów elektrycznych. Wynika to z faktu, że znaczne łatwiej narysować schemat odręcznie na kartce papieru a następnie posłużyć się niniejszym programem do przeniesienia go na komputer, niż za każdym razem wprowadzać schemat za pomocą myszki i klawiatury, szczególnie gdy zależy nam na szybkim uzyskaniu wyniku. Następnie, po przeniesieniu schematu na komputer program może automatycznie wygenerować schemat elektryczny w dowolnym programie do symulacji układów elektrycznych.

- **Elektroniczna archiwizacja dokumentacji**

Algorytmy rozpoznawania układów logicznych z powodzeniem można wykorzystać do przenoszenia dużych i skomplikowanych schematów istniejących tylko w wersji papierowej do wersji elektronicznej w celach serwisowych lub modyfikacji.

3.2 Propozycje dalszego rozszerzania programu

- **Zastosowanie do rozpoznawania symboli elektrycznych**

Jak już wspomniano w poprzednim rozdziale interesującym rozwiązaniem byłaby rozbudowa programu o dalsze obiekty logiczne oraz elektryczne, co w efekcie da bardziej uniwersalny program i pozwoli na rozpoznawanie bardziej skomplikowanych schematów.

- **Generowanie modeli w formacie zgodnym ze specjalistycznymi programami do analizy układów elektrycznych**

Program może zostać rozszerzony o algorytm umożliwiający konwersję rozpoznanych układów elektrycznych do formatu zgodnego ze specjalistycznymi programami do analizy układów elektrycznych (np. OrCad).

Spis rysunków

Spis tabel

Bibliografia

[1] Bołdak Cezary „*Cyfrowe przetwarzanie obrazów*" wykłady Białystok University of Technology 2008,

[2] Cyganek Bogusław „*Komputerowe przetwarzanie obrazów trójwymiarowych"* 2002,

[3] Doros Marek „*Przetwarzanie obrazów*" -WIT, sem.6, 2006/2007,

[4] Gilewska Grażyna „*Przetwarzanie obrazów*" -wykłady Politechnika Białostocka Katedra Telekomunikacji i Aparatury Elektronicznej, http://teleinfo.pb.edu.pl/~gilg/piro.html

[5] Image Processing Toolbox For Use with MATLAB,

[6] *Image Analysisand Recognition- International Conference,* ICIAR 2004 Porto, Portugal, September 29 - October 1, 2004,

[7] Javidi Bahram „*Image Recognition and Classification*" Marcel Dekker 2002,

[8] Jaworska Tatiana „*Przetwarzanie obrazu*"- www.ibspan.waw.pl/~jaworska,

[9] Korzyńska A., Przytulska M. „*Przetwarzanie obrazów*" PJWSTK Warszawa 2005,

[10] Kujawińska M." *Cyfrowe przetwarzanie obrazów*" wykłady. Zakład Inżynierii Fotonicznej na Wydziale Mechatroniki Politechniki Warszawskiej,

[11] Matulewski J. „Matlab"- Materiały do zajęć dydaktycznych na Wydziale Fizyki, Astronomii i Informatyki Stosowanej Uniwersytetu Mikołaja Kopernika w Toruniu, http://www.fizyka.umk.pl/~jacek/dydaktyka/matlab.pdf

[12] Rafael C. Gonzalez, Richard E. Woods „*Digital Image Processing*"- second edition,

[13] Rotter Paweł „*Sterowanie ekspertowe* i *systemy diagnostyczne*" wykłady AGH 2008,

[14] Rotter Paweł „*Zastosowanie metryki Hausdorffa do identyfikacji obiektów obrazu*" - praca magisterska Kraków 1999,

[15] Sing-Tze Bow „*Pattern Recognition Image Preprocessing*" Marcel Dekker 2002,

[16] Strumiłło P. „Przetwarzanie obrazów" wykłady w Instytucie Elektroniki Politechniki Łódzkiej (http://www.eletel.p.lodz.pl/pstrumil/po/poindex.htm),

[17] Szczepański Marek "*Cyfrowe Przetwarzanie Obrazów*"- wstęp do ćwiczenia11,

[18] Tadeusiewicz R. „ *Systemy wizyjne robotów przemysłowych*" WNT Warszawa 1992,

[19] Tadeusiewicz R., Korohoda P. „ *Komputerowa analiza i przetwarzanie obrazów*" Wydawnictwo Fundacji Postępu Telekomunikacji, Kraków 1997,

[20] Tadeusiewicz R., Flasiński M. „*Rozpoznawanie obrazów"*. PWN, Warszawa 1991,

[21] Theodoridis S., Koitroumbas K. „*Pattern Recognition*” -third edition- Elsevier 2006,

[22] Wróbel Zygmunt, Koprowski Robert „ *Praktyka przetwarzania obrazów w programie Matlab*” EXIT Warszawa 2004,

Dodatek A

Opis techniczny programu *bramki*.

M-pliki (skrypty i funkcje) wchodzące w skład programu:

- **a.m** – funkcja wykonuje określoną pętlę składającą się z podfunkcji, tyle razy ile wynosi liczba rozpoznanych bramek.
- **pobranie_obrazu.m** – funkcja wczytuje obraz oraz wykonuje kolejno operacje segmentacji na całym obrazie, znajduje maksymalną ilość bramek znajdujących się na rozpoznawanym obrazie, a także tworzy macierz G o ilości kolumn odpowiadającej maksymalnej ilości rozpoznanych bramek.
- **obcinanie.m** – funkcja przeprowadza operację indeksacji i wycina z całego rozpoznawanego obrazu prostokąt w którym znajduje się bramka o danym indeksie, pozostawiając jedno-pikselowe otoczenie.
- **wykrywanie_negacji.m** – funkcja wykrywa czy dana bramka posiada symbol negacji, informacja ta jest następnie zapisywana do macierzy G. Jeśli aktualna bramka posiada symbol negacji to w wierszu nr 6 w kolumnie odpowiadającej numerowi indeksu przypisanego danej bramce pojawia się liczba 1, jeśli nie to 0.
- **obwod.m** – funkcja znajduje krawędź zewnętrzną danej bramki za pomocą polecenia „edge”, a następnie oblicza obwód bramki. Obwód zostaje zapisany w wierszu 20 macierzy G.
- **pole.m** – funkcja oblicza pole bramki. Pole zostaje zapisane w wierszu 21 macierzy G.
- **moment_zwykly.m** – funkcja oblicza moment zwykły bramki.
- **WBB.m** – funkcja oblicza współczynnik Blaira-Blissa bramki. Współczynnik zostaje zapisany w wierszu 23 macierzy G.
- **moment_centralny_11.m** – funkcja oblicza moment centralny M_{11} bramki.
- **moment_centralny_20.m** – funkcja oblicza moment centralny M_{20} bramki.
- **moment_centralny_02.m** – funkcja oblicza moment centralny M_{02} bramki.
- **wspolczynnik_7.m** – funkcja oblicza współczynnik Lp1 bramki.
- **identyfikacja.m** – funkcja na podstawie określonych progów dla współczynników Malinowskiej i Blaira-Blissa identyfikuje bramkę oraz wpisuje w wierszu 7 macierzy G liczbę 1 - jeśli rozpoznana bramka to NOT, 2 – jeśli rozpoznana bramka to OR oraz 3 – jeśli AND.

- **wspolrzedne_bramek.m** – funkcja szuka współrzędnych wierzchołków prostokąta w których znajduje się bramka z uwzględnieniem jedno-pikselowego otoczenia bramki, a następnie przepisuje te współrzędne do macierzy G, wiersz 2 – współrzędna $i_{\min}$ prostokąta, wiersz 3 – współrzędna $j_{\min}$ prostokąta, wiersz 4 – współrzędna $i_{\max}$ prostokąta, 5 – współrzędna $j_{\max}$ prostokąta.
- **indeksacja_linii** – funkcja czyści wszystkie bramki z rozpoznawanego obrazu i pozostawia tylko połączenia („druty"). Następnie wykonuje operację indeksacji „drutów" i przeszukuje otoczenia bramek w poszukiwaniu połączeń. Jeśli zostanie znalezione jakieś połączenie, to jego indeks zostaje zapisany w macierzy G w wierszach 10, 11 jeśli połączenie znajduje się w otoczeniu lewym, 12, 13 jeśli połączenie znajduje się w otoczeniu prawym, 14, 15 jeśli połączenie znajduje się w otoczeniu górnym oraz 16, 17 jeśli połączenie znajduje się w otoczeniu dolnym.
- **xor1.m** – funkcja przeszukuje otoczenia w poszukiwaniu linii oznaczającej symbol bramki XOR.
- **funkcja_k.m** – funkcja rozpoznaje które połączenia to wejścia, a które to wyjścia oraz generuje odpowiedź programu.

Zmienne globalne w programie

- x – liczba zawierająca liczbę bramek rozpoznanych na schemacie.
- y – zmienna przechowująca indeks bramki, która w danej chwili jest przetwarzana w programie.
- L – zmienna zawierająca obwód danej bramki.
- S – zmienna zawierająca pole danej bramki.
- MZ – zmienna zawierająca moment zwykły bramki.
- W4 – zmienna zawierająca współczynnik Blaira-Blissa.
- M11 – zmienna zawierająca moment centralny M_{11}
- M20 – zmienna zawierająca moment centralny M_{20}
- M02 – zmienna zawierająca moment centralny M_{02}
- W7 – zmienna zawierająca współczynnik $Lp1$
- W3 – zmienna zawierająca współczynnik Malinowskiej

- G – macierz główna zawierająca najważniejsze dane o rozpoznawanym obrazie, jej definicja podana jest w rozdziale 2.1.2.
- L3 – macierz zawierająca obraz binarny, który jest wykorzystywany przy rozpoznawaniu negacji bez zalanych otworów.
- L7 – macierz zawierająca obraz binarny po binaryzacji.
- X1 – macierz zawierająca obraz binarny po zalaniu otworów funkcją „`bwfill`” z połączeniami pomiędzy bramkami
- X2 – macierz zawierająca obraz binarny po zalaniu otworów funkcją „`bwfill`” bez połączeń pomiędzy bramkami
- A4 – macierz zawierająca bramkę wyciętą z całości obrazu z jedno-pikselowym otoczeniem.
- P2 – macierz do których przepisywane są bramki wycięte z całego obrazu i ze zmienionymi indeksami na 1 (jeśli rozpoznawana w danym momencie bramka posiadała wcześniej po operacji indeksacji indeks np. 3).

Dodatek B

Wydruk programu *bramki*.

```
function a

global x;
global y;

pobranie_obrazu
y=0;
for y=1:x;
obcinanie
wykrywanie_negacji
xor1
if y==x
    indeksacja_linii
    funkcja_k
end
end

function pobranie_obrazu
global x;
global L7;
global L3;
global G;
global X1;
global X2;
global X3;

% L1=imread('nowy4.jpg');
cw5
L3=X3;
% L2=rgb2gray(L1);
% L20=imresize(L2,size(L2),'bilinear');
% L3=roicolor(L2,0,150);
%stąd pobieramy obraz L3 do rozpoznawania negacji
L4=bwfill(L3,'holes',8);
L40=bwmorph(L4,'thin',2);
L61=imopen(L40,ones(2,2));
L6=imclose(L61,ones(4,4));
L7=bwlabel(L6);
% figure;imshow(L20);
figure;imshow(L3);
figure;imshow(L4);
figure;imshow(L6);
figure;imshow(L7,[]);
X1=L4;      %przepisuje obraz z macirzy L4 do macierzy pomocniczej X1
X2=L6;      %przepisuje obraz z macirzy L6 do macierzy pomocniczej X2
[k,l]=size(L7);   % max rozmiar obrazu
 max(k);
  max(l);

x=0;                     % algorytm wyszukiwania max
x1=0;                   % ilości bramek
for k=1:max(k)
    for l=1:max(l)
        x1=L7(k,l);
        if x<x1
```

```
            x=x1;
        end
    end
end
G=zeros(20,x);
                        function obcinanie

                        global A1;
                        global A4;
                        global L7;
                        global y;
                        global P2;
                        P1=zeros(30,10);
                        [k,l]=size(L7);       % max rozmiar obrazu algorytm
wyszykiwania obiektów i tworzący macierze zastępcze
                            P1(1,y)=y;
                             P2=zeros(max(k),max(l));
                            for k=1:max(k)
                                for l=1:max(l)
                                    if L7(k,l)==y
%zamienia wartości z funkcji bwlabel(np 3 na 1) "binarnie"
                                     P2(k,l)=1;
                                    end
                                end
                            end
                        wspolrzedne_bramek
                        A1=not(P2);
                % obcina obraz do prostokąta w którym znajduje się bramka
                        % A3=rgb2gray(A1);
                        % A2=(A3>100);
                        A=bwlabel(double(A1));
                        [i,j]=find(A==0);
                         [a,b]=size(A);
                          min(i);
                           min(j);
                            max(i);
                             max(j);
                              A3=A;
                        if min(i)>1
                        for n=1:(min(i)-2);
                            A3(1,:)=[];
                        end
                        end

                        [i,j]=find(A3==0);
                        [a,b]=size(A3);
                        find min(j);
                        min(j);

                        if min(j)>1
                        for n=1:(min(j)-2);
                             A3(:,1)=[];
                        end
                        end

                        [i,j]=find(A3==0);
                        [a,b]=size(A3);
                        find max(i);
                        max(i);
```

```
                        if max(i)<a
                        for n=1:(a-(max(i)+1))
                            A3(a-n+1,:)=[];
                        end
                        end

                        [i,j]=find(A3==0);
                        [a,b]=size(A3);
                        find max(j);
                        max(j);

                        if max(j)<b
                        for n=1:(b-(max(j)+1))
                            A3(:,b-n+1)=[];
                        end
                        %skalowanie do jednakowego rozmiaru
                        A4=imresize(A3,[300 300],'nearest');
                        end
function wykrywanie_negacji
%sprawdza czy bramka jest z negacją i rozpoznaje rodzaj bramki

global L3;
global G;
global y;
global A4;
global S;
global L;
global W3;
global L31;
global x;
global MZ;
global M11;
global M20;
global M02;

    d2=G(2,y);
    d3=G(3,y);
    d4=G(4,y);
    d5=G(5,y);
    d42=d4-d2;
    d53=d5-d3;
    C1=zeros(d42,d53);
    for e=1:(d42+4)
        for e1=1:(d53+4)
            C1(e,e1)=L3((e+d2-2),(e1+d3-2));
        end
    end
L20=not(C1);
L21=imresize(L20,[300 300],'nearest');
L1=C1;
L2=bwfill(L1,'holes',8);
L4=imresize(L2,[300 300],'nearest');
L30=bitand(L4,L21);
L31=bwlabel(L30);

if y==x
 X10=not(L3);
 X11=bwfill(L3,'holes',8);
 X12=bitand(X11,X10);
 X13=bwlabel(X12);
```

```
 X114=label2rgb(X13);
 figure;imshow(X114,[]);
end

[k,l]=size(L31);                    % max rozmiar podobrazu -
 max(k);
  max(l);
 G(6,y)=0;

h=0;   % algorytm wyszukiwania max ilości obiektów jeśli h>2 wyświetla błąd
h1=0;
for k=1:max(k)
    for l=1:max(l)
        h1=L31(k,l);
        if h<h1
            h=h1;
        end
    end
end
if h<3
    for y1=1:h                  %pętla rozpoznawania bramek
   P1=zeros(30,10);
   [k,l]=size(L31);             % max rozmiar obrazu algorytm wyszukiwania
obiektów i tworzący macierze zastępcze
    P1(1,y1)=y1;
     P2=zeros(max(k),max(l));
     S2=0;                  %oblicza pole symbolu negacji w celu identyfikacji
    for k=1:max(k)
        for l=1:max(l)
            if L31(k,l)==y1
%zamienia wartości z funkcji bwlabel(np 3 na 1) "binarnie"
             P2(k,l)=1;
             S2=S2+1;
            end
        end
    end

 if S2<5000
    G(6,y)=1;
else

A1=not(P2);                         %obcina ramke obiektu do min
A=bwlabel(double(A1));
[i,j]=find(A==0);
 [a,b]=size(A);
  min(i);
   min(j);
    max(i);
     max(j);
      A3=A;
if min(i)>1
for n=1:(min(i)-2);
    A3(1,:)=[];
end %for
end %if

[i,j]=find(A3==0);
[a,b]=size(A3);
find min(j);
min(j);
```

```
if min(j)>1
for n=1:(min(j)-2);
     A3(:,1)=[];
end
end

[i,j]=find(A3==0);
[a,b]=size(A3);
find max(i);
max(i);

if max(i)<a
for n=1:(a-(max(i)+1))
    A3(a-n+1,:)=[];
end
end

 [i,j]=find(A3==0);
[a,b]=size(A3);
find max(j);
max(j);

if max(j)<b
for n=1:(b-(max(j)+1))
    A3(:,b-n+1)=[];
end  %for
%skalowanie do jednakowego rozmiaru
A4=imresize(A3,[300 300],'nearest');
% figure;imshow(A4);
pole
pole_cz
WBB
moment_zwykly
moment_centralny_11
moment_centralny_20
moment_centralny_02
obwod
W3=L/(2*sqrt(pi*S))-1;              %współczynnik Malinowskiej
WM=W3
M1=(M20+M02)/(MZ^2)
M2=((M20+M02)^2+4*M11^2)/MZ^4
M7=(M20*M02-M11^2)/MZ^4
wspolczynnik_7
G(25,y)=M1;
G(26,y)=M2;
G(27,y)=M7;
G(24,y)=W3;
identyfikacja
end %if
    end
    end
else
 'blad - przykro mi ale nie umiem rozpoznac Twojego schematu    :-('
end
                        function obwod
                        global A4;
                        global A5;
                        global L;
                        global G;
```

```
                        global y;
                        global R2;
                        B1=A4;
                        B2=edge (B1);
                        R2=B2;
                        B10=not(B2);
                        %figure;imshow(B10,[]);
                        [i,j]=find (B2==1);
                        max(i);
                        max(j);
                        L=0;
                        for i=1:max(i);              %liczy obwód figury
                            for j=1:max(j);
                                if B2(i,j)==1
                              L=L+1;
                                end
                            end
                        end

                        G(20,y)=L;
                        B3=not(B2);
                        A5=B3;

function pole
global A4;
global S;
global G;
global y;

B1=A4;
[i,j]=find (B1==1);
max(i);
max(j);
S=0;
for i=1:max(i);                    %liczy pole figury
    for j=1:max(j);
        if B1(i,j)==1
      S=S+1;
        end
    end
end

G(21,y)=S;
                     function moment_zwykly
                        global A4;
                        global G;
                        global y;
                        global MZ;
                        B1=A4;
                  [i,j]=find (B1==1);
                  max(i);
                  max(j);
                  MZ=0;
                  for i=1:max(i);                    %liczy moment zwykły
                      for j=1:max(j);
                          if B1(i,j)==1
                        MZ=MZ+1;
                          end
                      end
                  end
```

```
function pole_cz
global A4;
global S1;
global G;
global y;
B1=A4;
[i,j]=find (B1==1);
max(i);
max(j);
a=(max(j)/10);
S1=0;
for i=1:max(i);                  %liczy pole figury dla j=1/10 max(j)
    for j=1:a;
        if B1(i,j)==1
      S1=S1+1;
        end
    end
end

G(22,y)=S1;
```

```
function WBB                %współczynnik Blaira-Blissa
global A4;
global S;
global W4;
global G
global y;
global x1;
global y1;

B1=A4;
[i,j]=find (B1==1);
max(i);
max(j);
x2=0;
for i=1:max(i)
    for j=1:max(j)
        if B1(i,j)==1
        x2=x2+i;
        else
        x2=x2+0;
        end
    end
end
y2=0;
for i=1:max(i)
    for j=1:max(j)
        if B1(i,j)==1
        y2=y2+j;
        else
        y2=y2+0;
        end
    end
end
x1=x2/S; %współrzędna x środka ciężkości
y1=y2/S; %współrzędna y środka ciężkości
r=0;
for i=1:max(i)
    for j=1:max(j)
        if B1(i,j)==1
        r=r+((j-y1)*(j-y1)+(i-x1)*(i-x1));
```

```
%r-odległość piksela obiektu od środka ciężkości obiektu
                            else
                            r=r+0;
                            end
                        end
                    end
                    W4=S/(sqrt(2*pi*r));
                    WBB=W4
                    % G(23,y)=(W4*1000000);

function moment_centralny_11
    global A4;
    global G;
    global x1;
    global y1;
    global M11;
    B1=A4;
    [i,j]=find (B1==1);
    max(i);
    max(j);
    M11=0;
    for i=1:max(i)
        for j=1:max(j)
            if B1(i,j)==1
                M11=M11+(i-x1)*(j-y1);
            end
        end
    end

    function moment_centralny_20
    global A4;
    global G;
    global x1;
    global y1;
    global M20;
    B1=A4;
    [i,j]=find (B1==1);
    max(i);
    max(j);
    M20=0;
    for i=1:max(i)
        for j=1:max(j)
            if B1(i,j)==1
                M20=M20+((i-x1)^2);
            end
        end
    end


    function moment_centralny_02
    global A4;
    global G;
    global x1;
    global y1;
    global M02;
    B1=A4;
    [i,j]=find (B1==1);
    max(i);
    max(j);
    M02=0;
    for i=1:max(i)
```

```
        for j=1:max(j)
            if B1(i,j)==1
                M02=M02+((j-y1)^2);
            end
        end
    end

        function wspolczynnik_7
            global x1;
            global y1;
            global R2;
            global W7;
            [i,j]=find (R2==1);
            max(i);
            max(j);
            r=1000;
            R=0;
       for i=1:max(i)
           for j=1:max(j)
              if R2(i,j)==1
              r1=sqrt((j-y1)*(j-y1)+(i-x1)*(i-x1));
%r-odległość piksela obiektu od środka ciężkości obiektu
              if r1<r
                  r=r1;
              end
              if r1>R
                  R=r1;
              end
              end
         end
      end
                  W7=r/R
                    function identyfikacja
                    global W3;
                    global W4;
                    global S1;
                    global G;
                    global y;
                    global W7;

                    a=0;
                    b=0;


                    if W3<0.8
                        a=a+1;
                    else
                        a=a+0;
                    end

                    if W3>1.2
                        b=b+1;
                    else
                        b=b+0;
                    end

                    if W4>0.45
                        a=a+1;
                    else
                        a=a+0;
```

```
                    end

                    if W4<0.4
                        b=b+1;
                    else
                        b=b+0;
                    end

                    if a==2
                        if W7<0.24
                        G(7,y)=1;           %'rozpoznana bramka to ->  NOT'
                        end
                        if W7>0.26
                        G(7,y)=2;           %'rozpoznana bramka to ->  OR'
                        end
                    end

                    if b==2
                        G(7,y)=3;           %'rozpoznana bramka to ->  AND'
                    end
```

```
function wspolrzedne_bramek
global G;
global P2;
global y;
%szuka współrzędnych min prostokąta
%w którym zawiera się bramka
%najpiew po "k" potem "l"
[k,l]=size(P2);
max(k);
max(l);
x1=0;
 x2=0;
  y1=0;
   y2=0;
    e=0;
for k=1:max(k)
    for l=1:max(l)
      if e==0
        if P2(k,l)==1
            x1=k;
             y1=l;
              e=1;
        end
      end
    end
end

%szuka współrzędnych min prostokąta
%w którym zawiera się bramka
%najpiew po "l" potem "k"
e=0;
for l=1:max(l)
    for k=1:max(k)
        if e==0
            if P2(k,l)==1
             x2=k;
              y2=l;
               e=1;
```

```
            end
        end
    end
end

%wpisanie współrzędnych min do macierzy
if x1<=x2
    G(2,y)=x1;
else
    G(2,y)=x2;
end
if y1<=y2
    G(3,y)=y1;
else
    G(3,y)=y2;
end
%koniec wpisywania współrzędnych min do macierzy

% szuka współrzędnych max prostokąta
% w którym zawiera się bramka
% najpierw po "k" potem "l"
%%%%%%%%%%%
x3=0;
 x4=0;
  y3=0;
    y4=0;
     t1=G(2,y);
      t2=G(3,y);
 for k=t1:max(k)
    e=0;
    for l=t2:max(l)
        if e==0
            if P2(k,l)==1
                x3=k;
                 y3=l;
                  e=1;
            end
        end
    end
end
% szuka współrzędnych max prostokąta
% w którym zawiera się bramka
% najpierw po "l" potem "k"
%%%%%%%%%%%
for l=t2:max(l)
    e=0;
    for k=t1:max(k)
      if e==0
         if P2(k,l)==1
        x4=k;
        y4=l;
        e=1;
         end
      end
   end
end

%wpisanie współrzędnych max do macierzy
if x3>=x4
   G(4,y)=x3;
else
```

```
   G(4,y)=x4;
end
if y3>=y4
   G(5,y)=y3;
else
   G(5,y)=y4;
end
                                %koniec wpisywania współrzędnych max do
macierzy
                                %koniec algorytmu szukania współrzędnych

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%

function cw5
global X3;
% L1=imread('wawel.jpg');
% L2=rgb2gray(L1);
% L3=imresize(L2,size(L2)*1.3,'bilinear');
% L4=roicolor(L3,0,120);  %stąd pobieramy obraz L3 do rozpoznawania negacji
% L5=imclose(L4,ones(5,5));
% L6=bwfill(L5,'holes');
% h=[-1,-1,1;-1,-2,1;1,1,1];
% % L7=double(L6);
% % L8=edge(L7);
% L7=filter2(h,L6);
% figure;imshow(L6,[]);



L1=imread('DSC02106.jpg');
L2=rgb2gray(L1);
% L30=histeq(L2);

% prog= graythresh(L2);
% L3 = im2bw(L2,prog);

L3=roicolor(L2,0,170);
L30=bwmorph(L3,'thicken',15);
L4=imclose(L30,ones(6,6));
% L40=edge(L2);
% L5=bwfill(L4,'holes');
% L6=bwmorph(L5,'thin',15);
% L7=imopen(L6,ones(4,4));
L40=bwmorph(L4,'thin',10);
L41=imresize(L40,size(L40)/6,'bilinear');
% figure;imshow(L7);
figure;imshow(L41);
X3=L41;
L42=not(L41);
figure;imshow(L42);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%

function indeksacja_linii
```

```
global X1;
global G;
global x;
global x1;

Y1=X1;
[i,j]=size(Y1);
max(i);
max(j);

for a=1:x       %funkcja czyści wszystkie bramki i pozostawia tylko "druty"

   d2=G(2,a);
   d3=G(3,a);
   d4=G(4,a);
   d5=G(5,a);

  for e=(d2-20):(d4+20)
       for e1=(d3-20):(d5+20)
         Y1(e,e1)=0;
       end
  end
end

Y2=bwlabel(Y1);     %indeksacja "drutów"
% figure;imshow(Y2,[]);

x1=0;
x2=0;

for i=1:max(i)
    for j=1:max(j)
        x2=Y2(i,j);
        if x1<x2
            x1=x2;
        end
    end
end

for a=1:x           %funkcja sprawdza jakie druty znajdują się w otoczeniu
bramki

   d2=G(2,a);
   d3=G(3,a);
   d4=G(4,a);
   d5=G(5,a);

   c=10;
   t=0;
   for e=(d2-21):(d4+21)        %sprawdza lewe otoczenie bramki w
poszukiwaniu połączeń
       for e1=(d3-21):(d3-21)
          b=Y2(e,e1);
          for b1=1:x1
              if b==b1
                 if b==t
                 else
                  G(c,a)=b;
                  c=c+1;
```

```
                    t=b;
                end
            end
        end
    end
    if c>12
       error= 'blad - wykryto więcej niż dwa połączenia w lewym otoczeniu
jednej z bramek';
    end

    c1=12;
    t=0;
    for e=(d2-21):(d4+21)        %sprawdza prawe otoczenie bramki w
poszukiwaniu połączeń
        for e1=(d5+21):(d5+21)
            b=Y2(e,e1);
            for b1=1:x1
                if b==b1
                    if b==t
                    else
                    G(c1,a)=b;
                    c1=c1+1;
                    t=b;
                end
            end
        end
    end
    if c1>14
       error= 'blad - wykryto więcej niż dwa połączenia w lewym otoczeniu
jednej z bramek'
    end

    c1=14;
    t=0;
    for e=(d2-21):(d2-21)        %sprawdza górne otoczenie bramki w
poszukiwaniu połączeń
        for e1=(d3-21):(d5+21)
            b=Y2(e,e1);
            for b1=1:x1
                if b==b1
                    if b==t
                    else
                    G(c1,a)=b;
                    c1=c1+1;
                    t=b;
                end
            end
        end
    end
    if c1>16
       error= 'blad - wykryto więcej niż dwa połączenia w lewym otoczeniu
jednej z bramek'
    end

    c1=16;
    t=0;
    for e=(d4+21):(d4+21)        %sprawdza dolne otoczenie bramki w
poszukiwaniu połączeń
```

```
        for e1=(d3-21):(d5+21)
           b=Y2(e,e1);
           for b1=1:x1
               if b==b1
                   if b==t
                   else
                   G(c1,a)=b;
                   c1=c1+1;
                   t=b;
                  end
               end
           end
        end
    end
    if c1>18
       error= 'blad - wykryto więcej niż dwa połączenia w lewym otoczeniu
jednej z bramek'
    end

end
G
figure;imshow(Y2,[]);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%

function funkcja_k

global G;
global x;
global kod1;
global kod2;
global y;
global X1;

for i=48:48
    char(i);
end
a=0;
b=0;
c=0;
d=0;

for y=1:x
    c1=0;
    d1=0;
%-------------------------------------------------------------------------
    a1=G(10,y);     %rozpoznaje które połączenia to wejscia a które wyjścia
dla otoczenia lewego
    b1=G(11,y);

    if a1==0
        if b1==0
        end
    end

    if a1>0
```

```
        if b1==0
            if c1==0
             c1=a1;
            else
             d1=a1;
            end
        end
    end

    if a1==0
          if b1>0
              if c1==0
               c1=b1;
              else
               d1=b1;
              end
          end
   end

    if a1>0
        if b1>0
            a=a1;
            b=b1;
        end
    end
%-------------------------------------------------------------------------
 a1=G(12,y);     %rozpoznaje które połączenia to wejscia a które wyjścia
dla otoczenia prawego
 b1=G(13,y);

    if a1==0
        if b1==0
        end
    end

    if a1>0
        if b1==0
            if c1==0
             c1=a1;
            else
             d1=a1;
            end
        end
    end

    if a1==0
          if b1>0
              if c1==0
               c1=b1;
              else
               d1=b1;
              end
          end
   end

    if a1>0
        if b1>0
            a=a1;
            b=b1;
        end
```

```
    end
%-------------------------------------------------------------------------
 a1=G(14,y);     %rozpoznaje które połączenia to wejscia a które wyjścia
dla otoczenia gorne
 b1=G(15,y);

    if a1==0
        if b1==0
        end
    end

    if a1>0
        if b1==0
            if c1==0
             c1=a1;
            else
             d1=a1;
            end
        end
    end

    if a1==0
          if b1>0
              if c1==0
               c1=b1;
              else
               d1=b1;
              end
          end
   end

    if a1>0
        if b1>0
            a=a1;
            b=b1;
        end
    end
%-------------------------------------------------------------------------
 a1=G(16,y);     %rozpoznaje które połączenia to wejscia a które wyjścia
dla otoczenia
 b1=G(17,y);

    if a1==0
        if b1==0
        end
    end

    if a1>0
        if b1==0
            if c1==0
             c1=a1;
            else
             d1=a1;
            end
        end
    end

    if a1==0
          if b1>0
              if c1==0
```

```
                c1=b1;
               else
                d1=b1;
               end
          end
    end

     if a1>0
         if b1>0
             a=a1;
             b=b1;
         end
     end
%-------------------------------------------------------------------------
   c=c1;
   d=d1;

     if a==0                    %przypisuje symbol A do zmiennej
       a=char(48);
     else
       a=char(64+a);
     end

     if b==0
       b=char(48);
     else
       b=char(64+b);
     end

     if c==0
       c=char(48);
     else
       c=char(64+c);
     end

     if d==0
       d=char(48);
     else
       d=char(64+d);
     end

     if c1>0
         if d1==0
     kod1=[a,char(44) b,char(44) c];
     rozpoznanie
     kod=[kod2, char(40) kod1 char(41)]
         end
     end

      if c1>0
         if d1>0
     kod1=[c,char(44) d];
     rozpoznanie
     kod=[kod2, char(40) kod1 char(41)]
         end
     end


     if y==x
         X2=not(X1);
```

```
         figure;imshow(X2);
        for y=1:x
            rozpoznanie
            d=G(2,y);
            e=G(3,y);
            text(e,(d-10),kod2);
        end
     end

end

function rozpoznanie

    global G;
    global y;
    global kod2;

   if G(7,y)==1
     kod2='NOT';
   end

   if G(7,y)==2
       if G(6,y)==1
           if G(8,y)==0
            kod2='NOR';
           end
       end
   end

   if G(7,y)==2
       if G(6,y)==0
           if G(8,y)==0
           kod2='OR';
           end
       end
   end

     if G(7,y)==2
       if G(6,y)==1
           if G(8,y)==1
               kod2='XNOR';
           end
       end
     end

   if G(7,y)==2
       if G(6,y)==0
           if G(8,y)==1
           kod2='XOR';
           end
       end
   end


   if G(7,y)==3
       if G(6,y)==1
           kod2='NAND';
       else
           kod2='AND';
```

```
        end
    end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function xor1
global X1;
global X2;
global G;
global y;

X20=imclose(X2,ones(5,5));
X21=bwmorph(X20,'thicken',10);
X3=not(X21);
X4=bitand(X1,X3);  %różnica logiczna obrazów
i1=G(2,y);          %przepisuje pierwszą współrzędną pionową z macierzy
głównej G
j1=G(3,y);          %przepisuje drugą współrzędną poziomą z macierzy
głównej G
i2=G(4,y);          %przepisuje trzecią współrzędną pionową z macierzy
głównej G
j2=G(5,y);         %przepisuje czwartą współrzędną poziomą z macierzy
głównej G

for i=1:(i2-i1+40)
    for j=1:(j2-j1+40)
        X5(i,j)=X4((i+(i1-20)),(j+(j1-20)));
    end
end

[k,l]=size(X5);
max(k);
max(l);
b=0;
for k=1:max(k)            %wykrywa ilość pikseli w pionie
    a=0;
    for l=1:max(l)
        if X5(k,l)==1
         a=1;
        end
    end
    if a==1
       b=b+1;
    end
end
if b>40
    G(8,y)=1;
end
% X6=imresize(X5,[300 300],'nearest');
% figure;imshow(X6);
end
```

Dodatek C

Tabela 12. Porównanie wartości współczynników dla bramek NOT, AND i OR

Rodzaj bramki	NOT	AND	OR	NOT	AND	OR	NOT	AND	OR	NOT	AND	OR	NOT	AND	OR	NOT	AND	OR
Nr bramki	M1	M1	M1	M2	M2	M2	M7	M7	M7	W. Malinowskiej	W. Malinowskiej	W. Malinowskiej	W. Blaira-Bliss	W. Blaira-Bliss	W. Blaira-Bliss	Lp1	Lp1	Lp1
1.	0.3637	2.8262	0.4571	0.1325	7.9879	0.2089	0.0282	1.4959	0.0522	0.4787	2.7932	0.5803	0.6615	0.2373	0.5901	0.1045	0.1740	0.3957
2.	0.3692	3.5016	0.5734	0.1363	12.7096	0.3287	0.0293	2.9312	0.0817	0.4912	2.7525	0.7260	0.6566	0.2132	0.5269	0.0944	0.3171	0.3818
3.	0.3640	2.7155	0.5404	0.1328	7.3774	0.2921	0.0285	1.4091	0.0730	0.4522	2.7908	0.6731	0.6612	0.2421	0.5427	0.1201	0.1538	0.3266
4.	0.3727	2.3171	0.4755	0.1389	5.3697	0.2261	0.0294	1.0686	0.0564	0.5163	2.7283	0.5916	0.6535	0.2621	0.5785	0.1516	0.1080	0.3644
5.	0.3749	3.0156	0.5158	0.1406	9.3261	0.2662	0.0299	1.9276	0.0665	0.5126	2.7226	0.6231	0.6515	0.2297	0.5555	0.1668	0.2119	0.3303
6.	0.3634	2.3250	0.5214	0.1321	5.4056	0.2719	0.0283	1.0303	0.0679	0.4830	2.6472	0.6190	0.6618	0.2616	0.5525	0.1520	0.1175	0.4278
7.	0.3576	2.6241	0.5043	0.1280	6.8888	0.2543	0.0275	1.3641	0.0635	0.4540	2.8383	0.6579	0.6671	0.2463	0.5618	0.1366	0.1274	0.3946
8.	0.3666	3.3704	0.5522	0.1344	11.3612	0.3049	0.0286	2.2698	0.0761	0.5415	3.1468	0.6520	0.6589	0.2173	0.5369	0.1588	0.1625	0.4193
9.	0.3594	3.5657	0.5227	0.1291	12.7158	0.2733	0.0275	2.5747	0.0693	0.4864	3.2362	0.6404	0.6655	0.2113	0.5518	0.1295	0.1635	0.3462
10.	0.3601	3.3756	0.5766	0.1297	11.3964	0.3328	0.0275	2.2500	0.0828	0.5450	3.1082	0.6904	0.6648	0.2171	0.5254	0.1039	0.1722	0.2804
11.	0.3673	1.9765	0.5504	0.1349	3.9130	0.3029	0.0291	0.6995	0.0757	0.4698	2.3693	0.6452	0.6583	0.2838	0.5378	0.1744	0.1318	0.3535
12.	0.3656	2.4119	0.4814	0.1337	5.8175	0.2318	0.0287	1.0611	0.0579	0.4875	2.5900	0.5848	0.6598	0.2569	0.5750	0.1678	0.1697	0.4006
13.	0.3717	2.7879	0.5533	0.1384	7.7729	0.3061	0.0312	1.5546	0.0764	0.4546	2.9216	0.6433	0.6543	0.2389	0.5363	0.1414	0.1230	0.3363
14.	0.3688	2.7793	0.5797	0.1360	7.7243	0.3363	0.0293	1.5016	0.0839	0.4912	2.8829	0.6705	0.6569	0.2393	0.5240	0.1136	0.1487	0.3300
15.	0.3625	2.9186	0.5579	0.1314	8.5203	0.3115	0.0281	1.6683	0.0776	0.4912	2.9321	0.6547	0.6626	0.2335	0.5341	0.1399	0.1420	0.3166
16.	0.3630	3.9636	0.5352	0.1318	15.7249	0.2864	0.0284	3.1453	0.0716	0.5301	3.3478	0.6253	0.6622	0.2004	0.5453	0.1613	0.1865	0.3155
17.	0.3737	3.8191	0.6078	0.1396	14.9102	0.3695	0.0302	3.5216	0.0923	0.5493	2.9636	0.7239	0.6526	0.2041	0.5117	0.1190	0.2762	0.2947
18.	0.3618	2.1715	0.5479	0.1316	4.7152	0.3002	0.0282	0.8485	0.0747	0.4843	2.4479	0.6134	0.6633	0.2707	0.5390	0.0731	0.1526	0.3185
19.	0.3679	1.9548	0.5562	0.1353	3.8214	0.3095	0.0289	0.6834	0.0773	0.5026	2.3141	0.6348	0.6578	0.2853	0.5349	0.1676	0.1411	0.3826
20.	0.3648	2.2184	0.5223	0.1331	4.9281	0.2730	0.0289	0.9047	0.0681	0.5032	2.5358	0.6101	0.6605	0.2679	0.5520	0.1205	0.1296	0.3915