

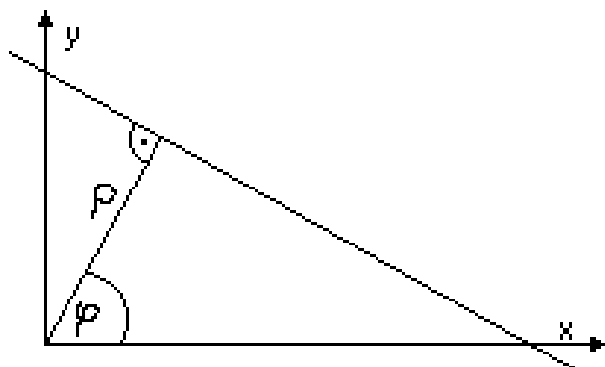
Wyższa Szkoła Informatyki Stosowanej i Zarządzania <b>Algorytmy przetwarzania obrazów</b> Inżynieria oprogramowania grupa ID 304, sem.VI, 2001 r.	Data :  17.06.2001
Projekt : <u>Program detekcji krawędzi z wykorzystaniem transformaty Hough'a</u> Wykonawca : Rafał Kalinowski	Prowadzący: Dr inż. Marek Doros

### Co to jest transformata Hough'a?

W programie tym zastosowano transformatę Hough'a jako metodę detekcji linii prostych. Program przekształca źródłowy obraz w przestrzeń parametrów, zwaną dalej tablicą akumulatorów. Później interpretuje dane zawarte w tej tablicy i przechodzi z powrotem do pierwotnej przestrzeni. Konwersja oparta jest o wzór normalny prostej:

$$x * \cos(\varphi) + y * \sin(\varphi) = \rho$$

Jest to równanie normalne prostej przedstawionej na Rys.1



Rys.1 Równanie normalne prostej

Metoda ta oparta jest na dualności pomiędzy punktami na prostej a parametrami tej krzywej  $(\varphi, \rho)$ .

Właściwości transformaty Hough'a :

- punkt obrazu koresponduje z sinusoidą w przestrzeni parametrów;
- punkt w przestrzeni parametrów koresponduje z linią prostą na obrazie;
- punkty leżące na tej samej prostej w obrazie korespondują z krzywymi (sinusoidami) przechodzącymi przez wspólny punkt w przestrzeni parametrów  $(\varphi, \rho)$ ;

- punkty leżące na tej samej krzywej ( sinusoidzie ) w przestrzeni parametrów korespondują z liniami prostymi przechodzącymi przez ten sam punkt na obrazie.

Ogólnie algorytm oparty o transformatę Hough'a ma dużą złożoność obliczeniową - rzędu:  $O(n^2)$ , gdzie  $n$  - liczba punktów niezerowych obrazu. Jest to oczywiście wadą tego algorytmu. W celu redukcji złożoności obliczeniowej w programie zastosowano dyskretyzację przestrzeni parametrów (  $\varphi$ ,  $\rho$  ) - utworzenie regularnej siatki ich wartości.

Zdyskretyzowana przestrzeń Hough'a będzie w tym przypadku dwuwymiarową tablicą „akumulatorów” zawierających wartości odpowiadające ilości przechodzących przez nie krzywych. Tworzenie tablicy można opisać w dwóch krokach:

Krok 1: Dla każdego punktu  $(x_1, y_1)$  obrazu korespondująca sinusoida ( dana równaniem  $\rho = x_1 \cdot \cos\varphi + y_1 \cdot \sin\varphi$  ) jest wprowadzana do tablicy poprzez powiększenie o 1 wartości oczek siatki ( akumulatorów ) leżących wzdłuż tej krzywej.

Krok 2: Przeszukiwanie tablicy w celu znalezienia akumulatorów o największych wartościach ( miejsca przecięć największej liczby krzywych ). Jeżeli pewne okno w tablicy (  $\varphi$ ,  $\rho$  ) ma wartość  $k$ , oznacza to, że dokładnie  $k$  punktów obrazu leży wzdłuż linii, której parametrami są (  $\varphi_1$ ,  $\rho_1$  ). W tym stwierdzeniu założyliśmy, że błąd kwantyzacji jest pomijalny.

Do zalet transformaty zaliczyć można:

- mała wrażliwość na szum
- mała wrażliwość na nieciągłości krawędzi
- mała wrażliwość na zwielokrotnienie krawędzi

Do wad transformaty zaliczyć można:

- wspomniana już złożoność obliczeniowa, w przypadku zdyskretyzowania tablicy, złożoność ta jest równa  $O(nl)$ , gdzie  $n$  – jest to liczba pikseli obiektów na obrazie, a  $l$  – jest wielkością  $\varphi$  (jeden z wymiarów tablicy akumulatorów)
- słabe działanie, gdy są duże dysproporcje co do długości linii
- słabe działanie, gdy są duże dysproporcje co do grubości linii

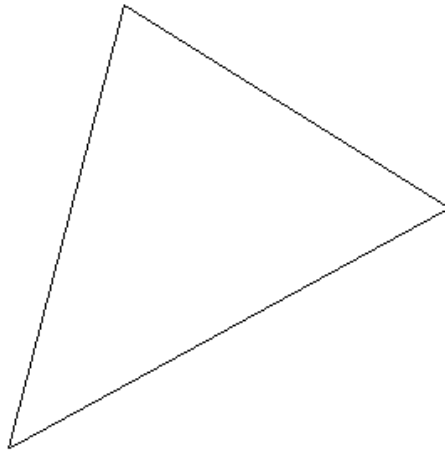
### **Wymagania programu.**

Program „Transformata Hough'a” został napisany w środowisku programistycznym Borland Delphi 5. Wymagania sprzętowe co do programu:

- Windows 9X/NT/2000,
- 32 Mb Ram
- Pentium 166 MMX

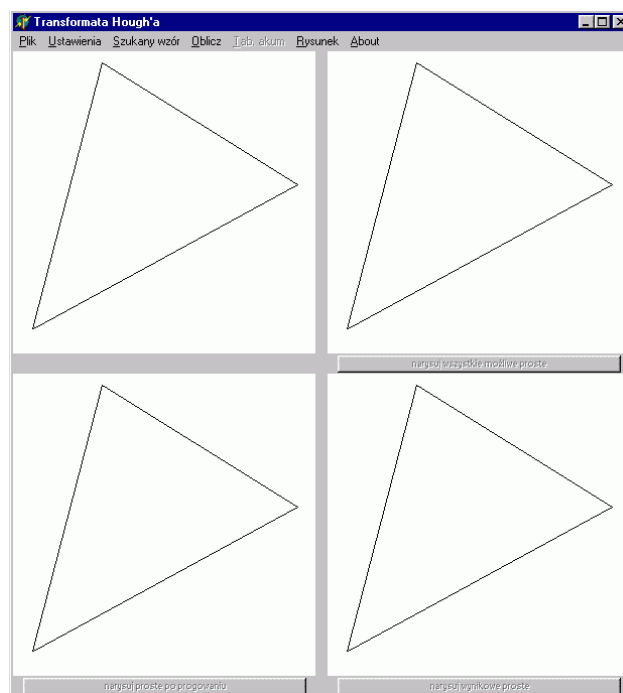
## Obsługa programu.

Przedstawię działanie programu na przykładowym obrazku. Jest to trójkąt

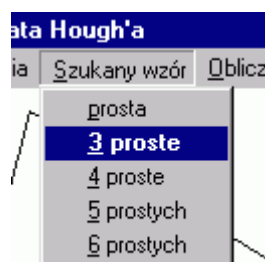


Żeby załadować ten plik do programu wybierz opcję: Plik -> Otwórz... i podaj nazwę pliku. W tym przypadku jest to duzy\_tro\_206\_206\_32.bmp.

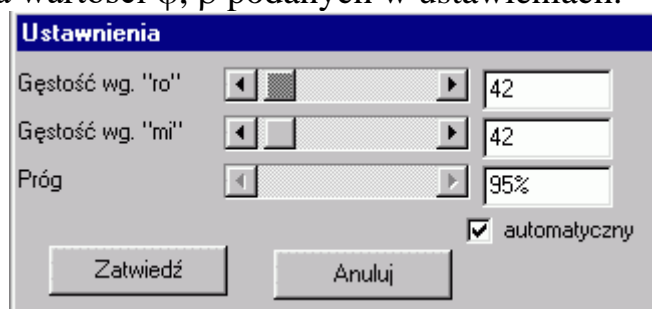
Wygląd programu:



Teraz musimy powiedzieć programowi, że ma szukać trzech linii, czyli trójkąta. Naciśnij Szukany wzór -> 3 proste.

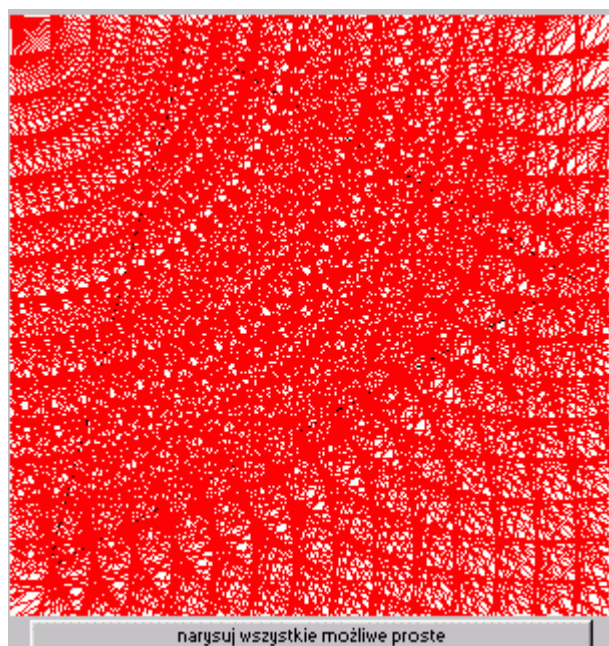


Następnie naciśnij Oblicz. W tym momencie program wyliczył tablicę akumulatorów, dla wartości  $\varphi$ ,  $\rho$  podanych w ustawieniach.



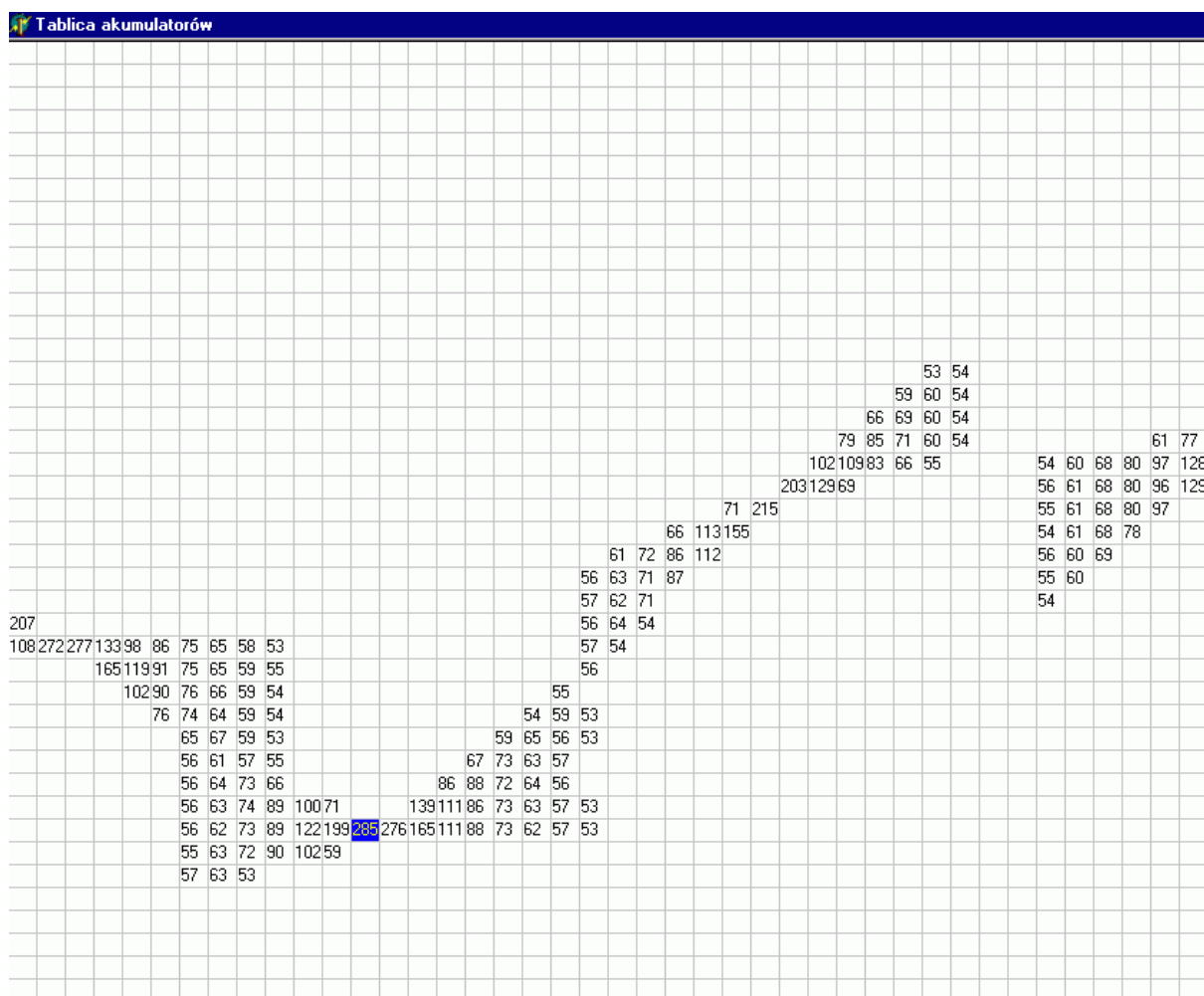
Tablica akumulatorów ma teraz wymiar 42x42. Próg jest wykorzystywany przez program do analizowania tablicy akumulatorów. Jeżeli jest włączony automatyczny próg, wtedy program ustala próg na podstawie średniej wartości tablicy akumulatorów.

Naciśnij teraz 'narysuj wszystkie możliwe proste'. Program narysuje wszystkie proste, które dane są w tablicy akumulatorów jako punkty niezerowe.



Jak widzimy zostało narysowanych bardzo wiele prostych. Jeżeli chcemy zobaczyć tablicę akumulatorów utworzoną w tym czasie naciśnij 'Tab. Akum.'. Pojawi się okienko z pasem sinusoid. Niebieski punkt oznacza maksimum.

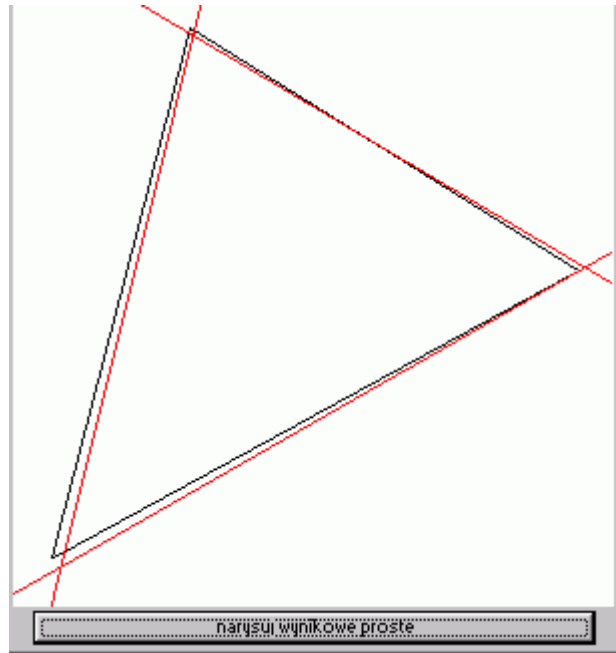




Widzimy, że progowanie usunęło dużą część punktów. Niestety gdy naciśniemy ‘narysuj wynikowe proste’, program stwierdzi, że nie widzi 3 prostych, gdyż w tablicy akumulatorów, nie znajduje 3 osobnych (rozłącznych) obiektów w rozumieniu sąsiedztwa 8-spójnego. Widzi tylko dwa. W takim razie musimy ustawić ręcznie parametry. Ustalmy wartości jak na obrazku:

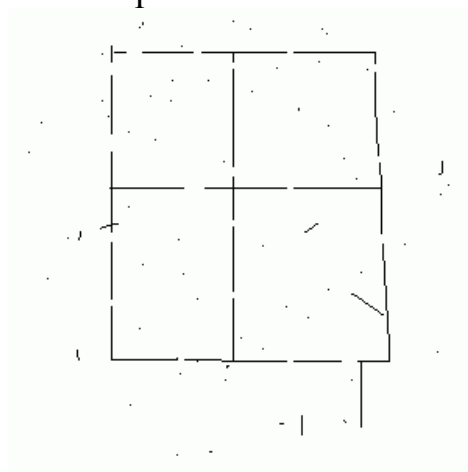
Zwiększyliśmy rozmiary tablicy akumulatorów, co spowoduje dokładniejsze odwzorowanie, oraz ustaliliśmy próg na 32% max. wartości w tablicy akumulatorów.

Powtórzymy jeszcze raz 1. ‘Oblicz’ oraz 2. ‘Narysuj wynikowe proste’. Program znalazł 3 „plamy” w tablicy akumulatorów i wybrał z każdej maksymalną wartość. Punkty te w tablicy akumulatorów są prostymi na obrazie.

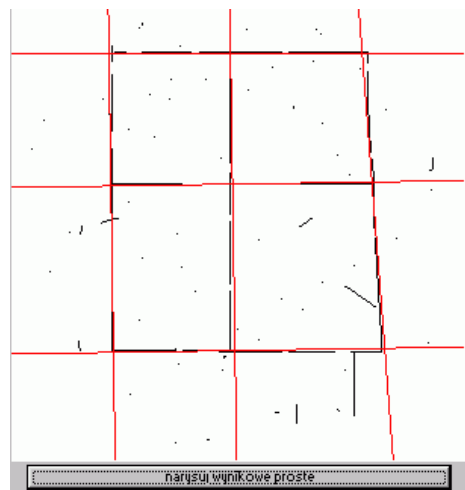


Widzimy, że program znalazł te proste.

A teraz inny przykład. Weźmiemy obraz bardziej nieczytelny i zobaczymy czy program będzie w stanie znaleźć proste.



a teraz wynik:



czyli OK.

**Uwagi końcowe.**

Program daje możliwość odręcznego malowania bezpośrednio na lewym-górnym obrazku. Obrazek ten musi być binarny (black-white). Jeżeli chcesz załadować obrazek z zewnętrznego pliku, to najlepiej gdyby to była bitmapa 300x300 monochromatyczna. Jeżeli załadujesz za duży obrazek to zostanie on przycięty. Jeżeli będzie za mały to będzie analizowany właśnie taki mały. Jeżeli będziesz próbował(a) załadować plik nie monochromatyczny to program zbinaryzuje go wg zasady, jeżeli wartości każdej ze składowych RGB są większe od 144 wtedy piksel będzie biały. W przeciwnym przypadku będzie czarny.