UNIVERSITY OF WARMIA AND MAZURY IN OLSZTYN
The Faculty of Technical Sciences
POLAND, 10-957 Olsztyn, M. Oczapowskiego 11
tel.: (48)(89) 5-23-32-40, fax: (48)(89) 5-23-32-55
URL: http://www.uwm.edu.pl/edu/sobieski/ (in Polish)

Podstawy programowania RAD

Delphi – Elementy Grafiki Komputerowej

Wojciech Sobieski

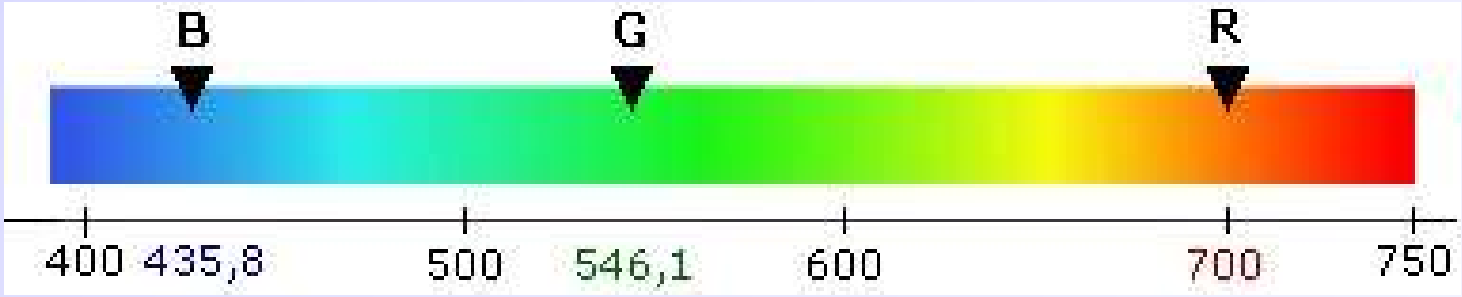
Olsztyn 2004-2011

Światło widzialne

10^{-15}-19^{-11}	promienie gamma
10^{-10}	promienie Roentgena
10^{-8}	ultrafiolet
10^{-6}	promienie widzialne
10^{-2}	podczerwień
10^{0}	fale radiowe krótkie
10^{3}	fale radiowe średnie
10^{6}	fale radiowe długie

Długości fal elektromagnetycznych.

Światło widzialne

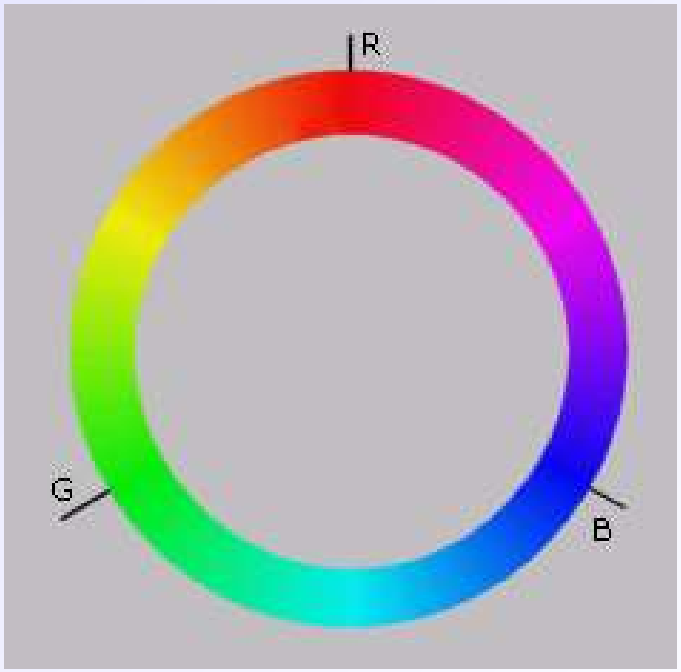


Barwa a długość fal.

Światło widzialne

Na użytek modeli doświadczalnych i matematycznych można traktować światło białe jako sumę pełnego natężenia trzech tylko barw prostych, nazywanych **barwami podstawowymi**.

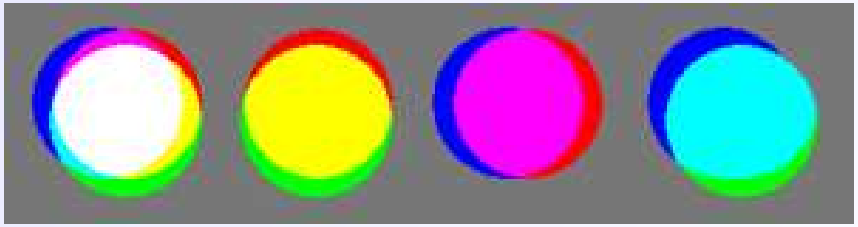
Każda barwa, inna niż biała, może być odwzorowana przez naruszenie tej równowagi, czyli pewną dysproporcję zmieszania trzech barw podstawowych.



Red-**G**reen-**B**lue

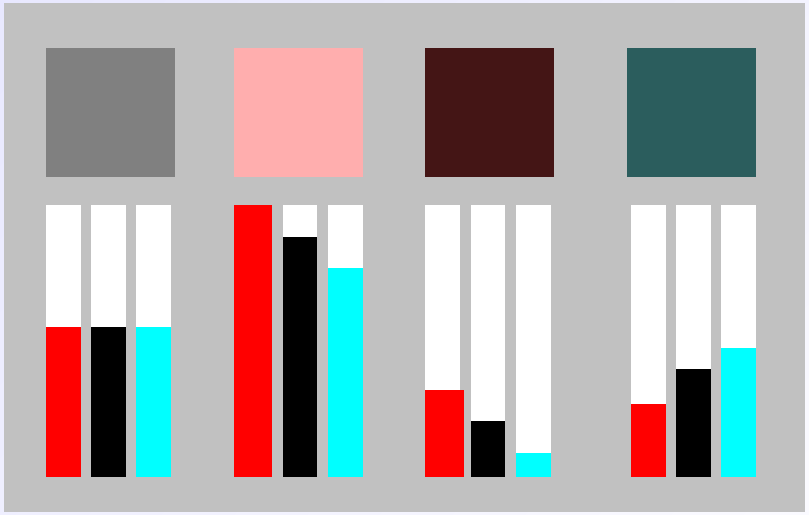
Modele barw

Addytywne mieszanie barw - czyli takie, w którym suma barw dąży do bieli. Odjęcie od światła białego jakiejś barwy tworzy mieszaninę barw z przewagą **barwy przeciwstawnej** do odjętej. Cały zakres barw prostych (widmo) daje się ująć w zamknięty **krąg barw**, w którym każda barwa prosta ma swoją barwę diametralnie przeciwstawną, a jednocześnie każda może być wyrażona przez zmieszanie, w odpowiednich proporcjach, trzech barw podstawowych.



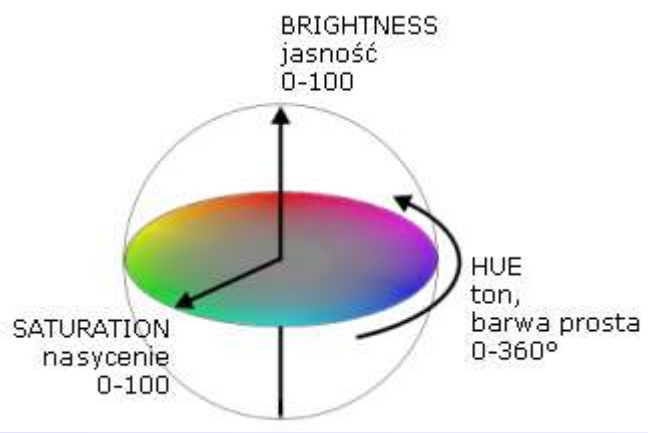
Modele barw

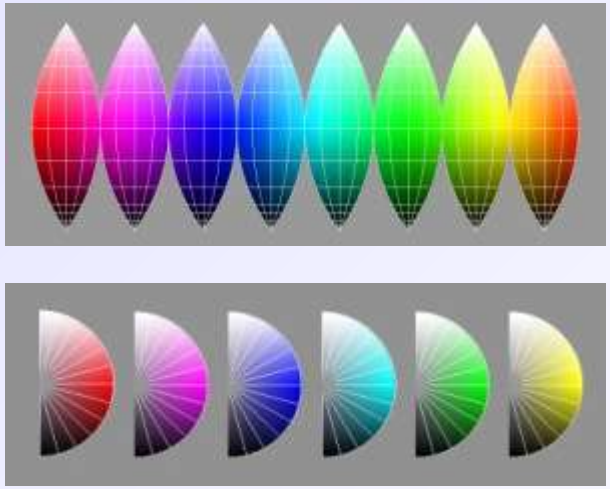
Oko ludzkie ma zazwyczaj do czynienia z mieszaniną fal świetlnych nie tylko o różnych proporcjach występowania każdej długości fali, ale także różnym natężeniu całego światła - podobne wartości względne **R**, **G** i **B** dają inną barwę przy innym poziomie natężenia.



Modele barw

Kula barw (model **HSB**) - przez odwzorowanie przestrzeni barw w postaci kuli uzyskuje się model matematyczny, który może służyć do pomiarów, identyfikowania i syntetyzowania dowolnych barw naturalnych.

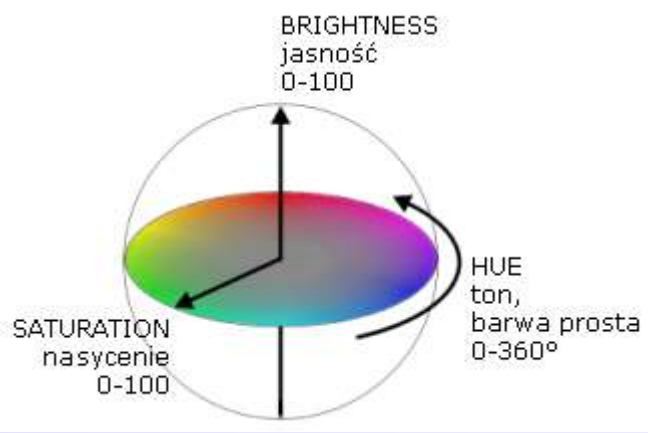




Modele barw

Kula barw (model **HSB**):

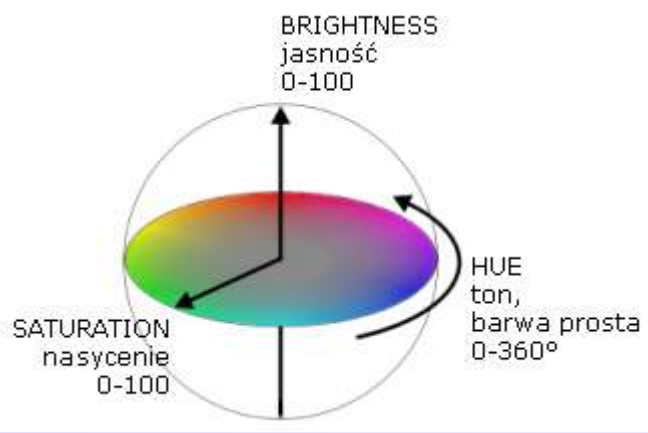
Hue (odcień) - mierzony kątowo, jak na kręgu barw prostych, 0-360°. Wartość "hue" najbliżej odpowiada potocznym określeniom kolorów, gdy nazywamy pewien walor niezależnie od czystości i jasności barwy.



Modele barw

Kula barw (model **HSB**):

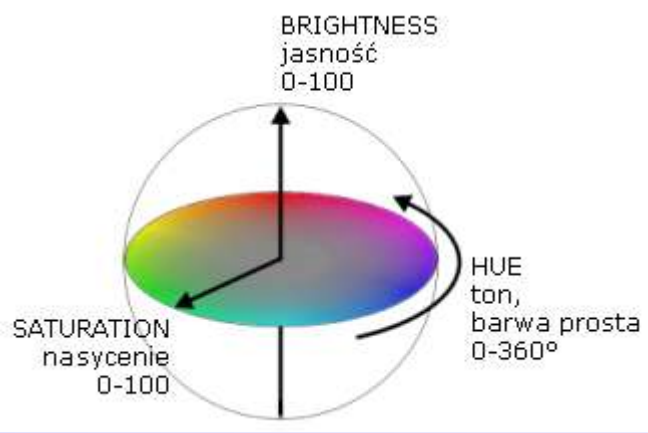
Saturation (nasycenie) - oddalenie od osi szarości, mierzone w skali 0-100.



Modele barw

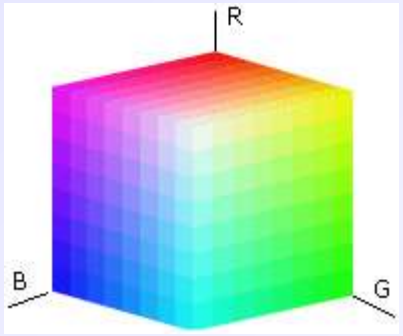
Kula barw (model **HSB**):

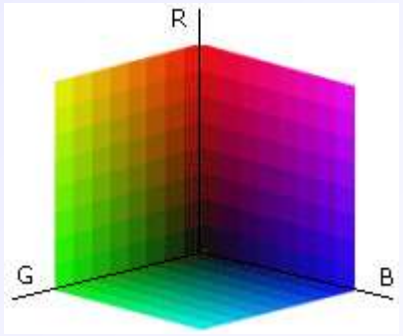
Brightness (jasność) - traktowana jako oddalenie od czerni a zbliżenie do bieli (czyli przyciemnienie lub rozjaśnienie), mierzone w skali 0-100



Modele barw

Sześcian barw (o współrzędnych **RGB**) - ponieważ geometria kuli i wartości kątowe nie zawsze są poręczne do obliczeń, więc czasem lepiej jest przestrzeń barw zamknąć w sześcianie (róg układu współrzędnych oznacza czerń).





Modele barw

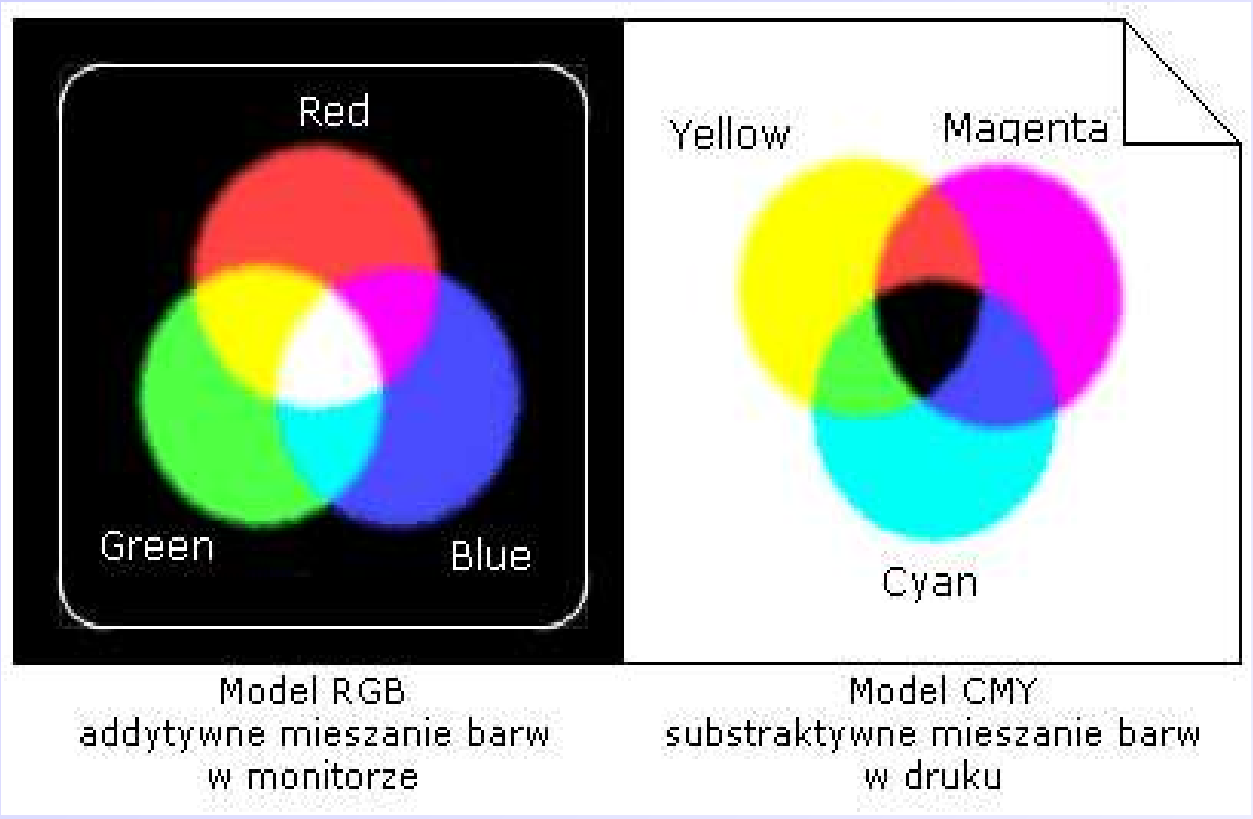
Czerń znajduje się w początku układu, i ma wartości **R**=0, **G**=0, **B**=0, czyli brak światła. Przeciwległy róg to biel, o wartościach **R**=100%, **G**=100%, **B**=100%. Zauważmy, że na krawędziach nie stykających się z punktem czerni i bieli znajdują się barwy proste. Przekątna sześcianu, od punktu czerni do bieli, reprezentuje skalę szarości, czyli wszystkie punkty, dla których **R**=**G**=**B**.

Taki model przestrzeni barw dobrze pasuje do fizycznego przebiegu syntezy barwy w telewizorze i monitorze komputerowym. Generowanie obrazu sprowadza się w ten sposób do prostego (pod względem logicznym) przetworzenia ciągu wyrażeń trzyliczbowych na wartości elektryczne sterujące lampą kineskopową.

Modele barw

Substarktywne mieszanie barw - polega ono na użyciu barwników, czyli substancji wybiórczo odbijających światło białe. Stosuje się barwy: błękitną (**C**yan), różową (**M**agneta), żółtą (**Y**ellow) oraz czarną (Blac**K** - w odmianie modelu, tzw. **CMYK**) Mieszanie barwników, w celu otrzymania pożądanej barwy przedmiotu, jest sztuką najstarszą i - w przeciwieństwie do syntezy trójchromatycznej - intuicyjnie zrozumiałą, choćby z czasów szkolnego malowania akwarelkami. Mieszanina wielu barwników dąży do efektu czerni (w odróżnieniu od syntezy **RGB** gdzie suma kolorów dawała biel).

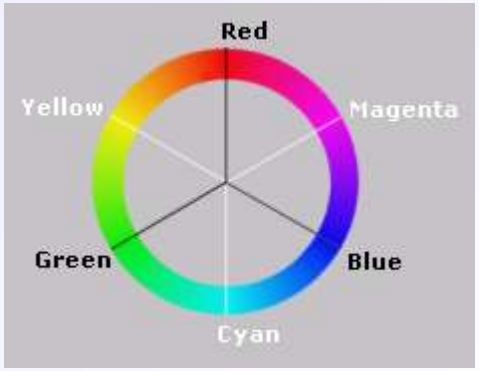
Modele barw



Model RGB
addytywne mieszanie barw
w monitorze

Model CMY
substraktywne mieszanie barw
w druku

Porównanie modeli RGB i CMY.



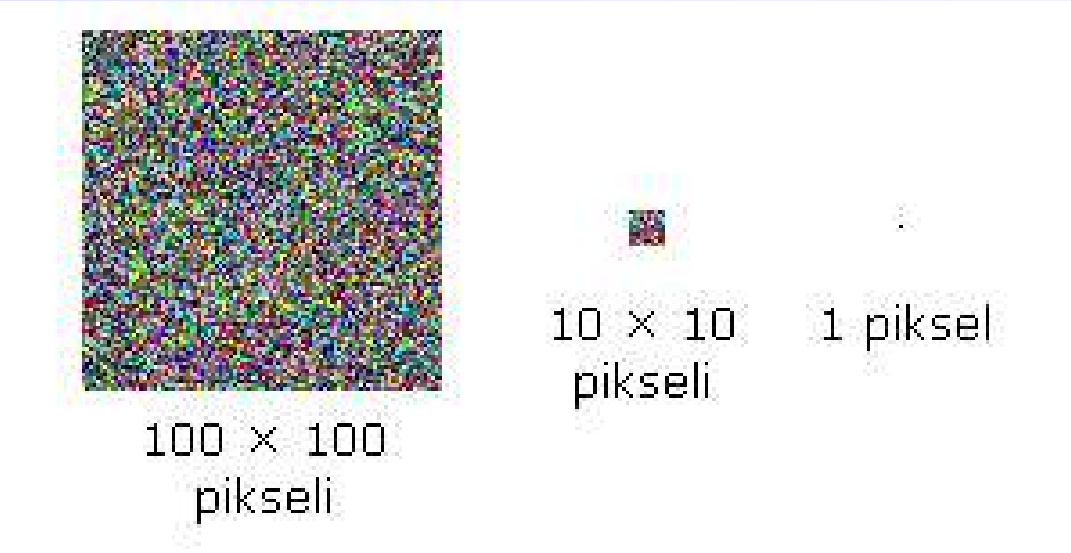
Związek między modelami RGB i CMY.

Wzorniki barw

Matematyczne modele RGB i CMYK służą analizie i syntezie dowolnych barwnych obrazów. Aby je reprodukować na ekranie lub w druku, modele te muszą z założenia ogarniać całą przestrzeń barw widzialnych. Często jednak potrzeby są skromniejsze, gdy chodzi tylko o zdefiniowanie pewnych arbitralnie ustalonych barw. Wtedy mamy do czynienia nie z **modelami przestrzeni barw**, lecz ze **wzornikami barw**.

Piksele

Obraz na monitorze komputera, podobnie jak telewizora, składany jest z pojedynczych punktów świetlnych, zwanych **pikselami**. Ich ilość, wahająca się od kilkuset tysięcy do ponad miliona składa się na rozdzielczość i wielkość obrazu.



Parametry obrazu

Konstrukcja i parametry danego monitora i komputera odpowiadają za przedstawienie obrazu na podstawie cyfrowej informacji o nim. Przy dostatecznych parametrach sprzętowych, obraz zależy wyłącznie od ilości i jakości tej informacji, a zwłaszcza od:

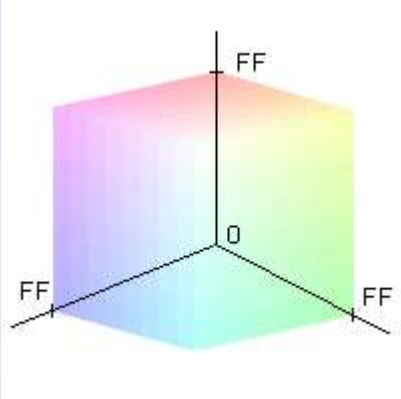
głębi barw mierzonej w bitach na piksel, oznaczającej dokładność informacji o barwie piksela;

sposobu kompresji obrazu czyli algorytmu użytego do skrótowego zapisu barwy wielu sąsiadujących pikseli;

uwzględniania ziarnistości, czyli technik ominięcia lub wykorzystania wpływu ziarnistej struktury pikseli na pożądaną jakość obrazu.

Parametry obrazu

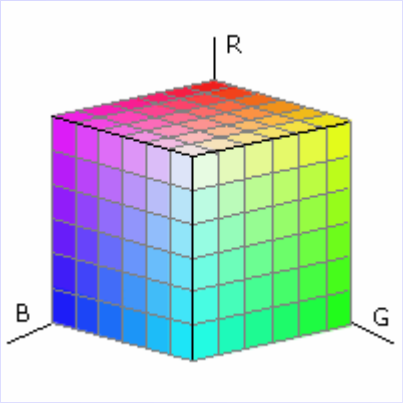
Głębia barw **–** jest to ilość informacji opisującej barwę piksela. Wybór ilości bitów do zapisania barwy to w istocie wybór dokładności odwzorowania przestrzeni barw. Jeśli osie **R**, **G** i **B** podzieli się na 256 wartości (od zera do 255), to otrzyma się do dyspozycji 256×256×256 czyli 16.777.216 barw, co jest o wiele więcej niż rozróżnia ludzkie oko. Do tak precyzyjnego zapisu barw potrzeba 8 bitów dla każdej składowej, czyli łącznie 24 bity na jeden piksel.



Komputer najchętniej obraca liczbami ośmiobitowymi, czyli **bajtami** i ich wielokrotnościami. Dlatego liczba ośmiobitowa, oddająca 256 różnych wartości, jest dla komputerowego przetwarzania równie okrągła i oczywista jak dla nas liczba 100. Z tego względu wartości bajtów wyraża się często w **układzie szesnastkowym**: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.

Parametry obrazu

Wybierając mniejszą głębię barw, ogranicza się możliwą liczbę reprezentujących całą przestrzeń barw i radykalnie zmniejsza się ilość informacji.



Ilość bitów przypisana do każdego piksela obrazu:

1 - 2 kolory,
4 - 16 kolorów,
8 - 256 odcieni szarości lub 256 kolorów,
16 - 65 tys. kolorów,
32 - 16.7 mln. kolorów.

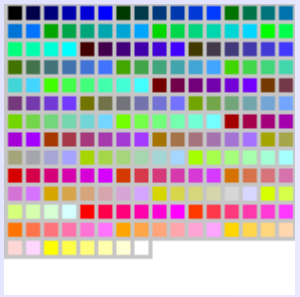
Parametry obrazu

***Paleta barw* –** jest to wzornik barw jednego lub wielu obrazów. Konkretne obrazy zwykle nie zawierają wszystkich barw i wydajniejszy jest zapis obrazu uwzględniający tylko te barwy, które faktycznie w nim występują. Taki zestaw barw nazywamy **paletą optymalną** dla danego obrazu, albo **paletą barw indeksowanych**. Paleta taka może służyć do konstruowania obrazów o podobnej tonacji (wspólnej palecie). Możliwe jest też manipulowanie obrazami przez nadawanie im określonych palet.

Parametry obrazu

Obraz w pełnej przestrzeni barw, najlepsza jakość, objętość: 47 KB.

Obraz w optymalnej dla niego palecie 256 barw, jakość porównywalna z najlepszą, objętość: 17 KB.



Obraz w ogólnej, "bezpiecznej" palecie 216 barw, jakość wyraźnie gorsza, objętość: 17 KB.

Parametry obrazu

Obraz w optymalnej palecie 16 barw, jakość niewiele gorsza od 216 barw, objętość: 8 KB.

Obraz w palecie 16 odcieni szarości, chyba lepiej tak, niż 16 kolorów, objętość: 8 KB.

Rozwiązanie skrajne: 2 barwy, trudno mówić o jakości, raczej o efektach graficznych, objętość: 2,5 KB.

Parametry obrazu

Kompresja - metoda zapisu danych (obrazu) w taki sposób, aby dane te zajmowały mniej pamięci dyskowej.

Parametry obrazu

Algorytmy **bezstratnej** kompresji danych, jak np. **GIF**, stanowią obojętne „opakowanie” wielokrotnego użytku i polegają na szczególnym sposobie oszczędnego zapisu oryginalnej informacji w możliwie małym pliku podobnie jak to robią programy do archiwizacji danych. Oszczędność polega na wyszukaniu regularności w układzie barwnych pikseli. Na przykład zamiast zapisywać wartości **RGB** szeregu pikseli o jednakowej barwie, wystarczy raz zapisać te wartości oraz podać, ilu kolejnych pikseli dotyczą.

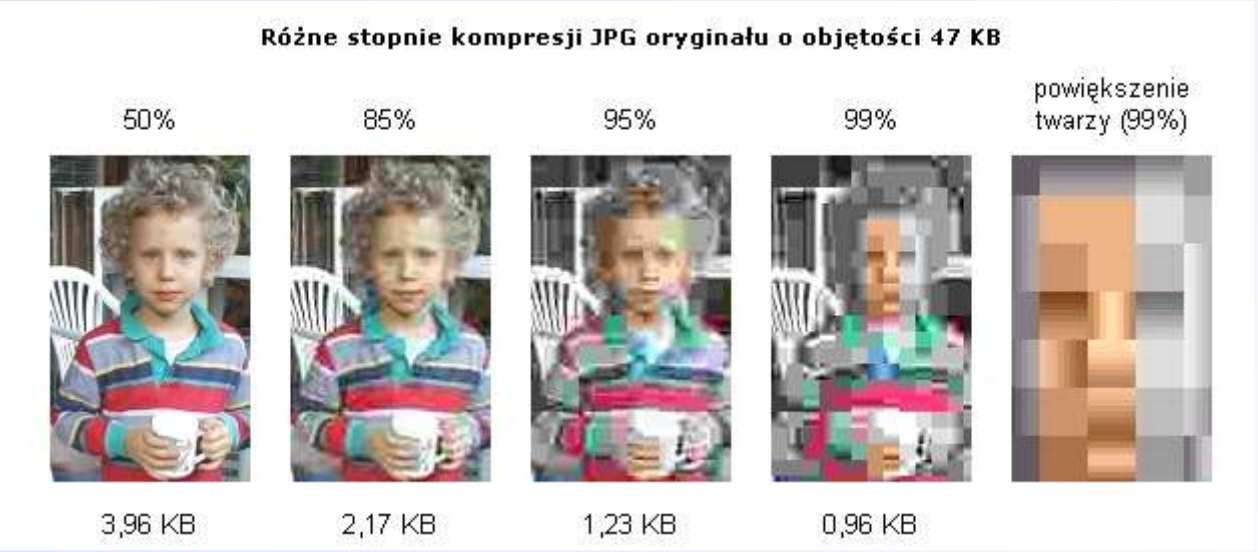
Parametry obrazu

Kompresja **stratna**, jak np. **JPG**, to w istocie stworzenie pochodnego obrazu, który jest uproszczony w tak wyszukany sposób, by dla ludzkiego oka owo uproszczenie było niezauważalne lub nieznaczne, stanowiąc jednak znaczną oszczędność z punktu widzenia komputerowego zapisu.

Na podstawie obrazu skompresowanego metodą stratną nie sposób odtworzyć obrazu oryginalnego. Informacja o szczegółach, choćby nieistotnych, jest tracona. Ta metoda kompresji dobrze służy jako sposób doraźnej prezentacji obrazów, których oryginały przechowujemy w sposób nieskompresowany, lub skompresowany bezstratnie.

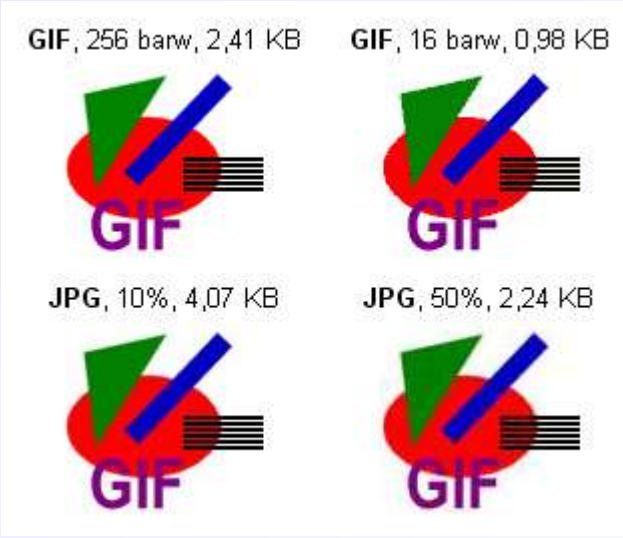
Parametry obrazu

Kompresja **JPG** dzieli obraz na kwadraty różnej wielkości i układu - zależnie od lokalnego zróżnicowania pikseli. W ich ramach dokonuje redukcji do pasm tonalnych lub do jednej barwy. Natężenie tych przekształceń jest regulowane.



Parametry obrazu

Drugi popularny algorytm kompresji, **GIF**, redukuje obraz do palety 256 barw i jest mało efektywny dla obrazów o miękkich przejściach tonalnych, za to dobrze odtwarza ostre kontrasty i jest bezstratny.



Parametry obrazu

Podstawowe formaty grafiki rastrowej:

BMP - standard Windows, brak kompresji, używany np. do tła.
JPG (***J**oint **P**hotographic **E**xperts **G**roup)* - prawie najlepsza kompresja(nieodwracalna) dla wiernego koloru,
GIF (***G**raphics **I**nterchange **P**rogram)* - popularny, dobra kompresja dla koloru 1, 2, 4, 8-bitowego,
TIF (*Tagged **F**ile **I**mage **F**ormat*) - różne warianty, kompresja bez strat LZW, używany przez faksy,
PCX - 256 kolorów, kompresja do 3 razy,
PCD - Kodak, wysoka jakość,
PNG - do WWW, kombinacja **GIF** i **JPG**,
DIB - odmiana BMP,
Inne - **IMG**, **MSP**, **TGA**, **WPG**, **WMF**, **PNT**, **MAC** ...

Parametry obrazu

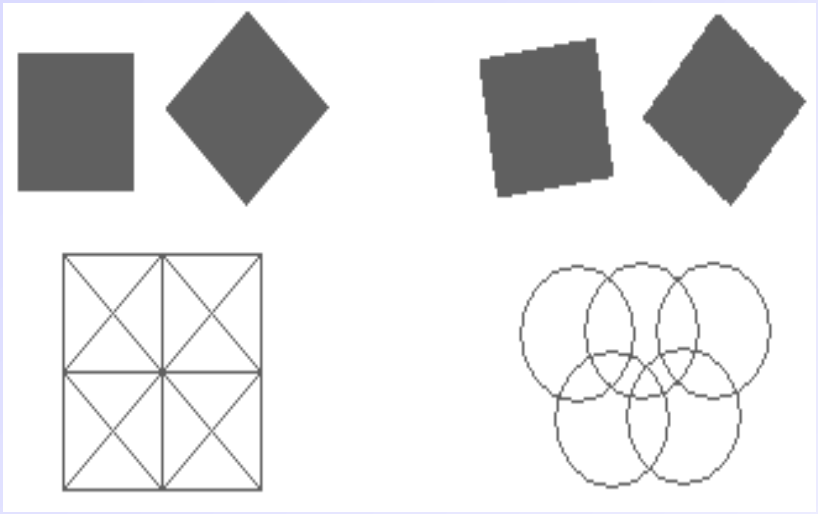
Podstawowe formaty grafiki wektorowej:

WMF (***W**indows **M**eta **F**ile*) – uniwersalny format zapisu wektorowego stosowany w MS Window,
CDR – format stosowany przez aplikacje firmy Autodesk: AutoCAD i in., standard przemysłowy,
EPS, **PS** (*Encapsulated PostScript*) – właściwie język opisu (wyglądu)stron, opracowany przez firmę Adobe, stosowany w zapisie dla celówpoligraficznych,
HPGL – format sterowania ploterami HP,
DXF – powszechnie stosowany w aplikacjach wspomagania projektowania CAD,
WPG – format stosowany przez WordPerfect,
CGM (***C**omputer **G**raphics **M**etafile*) – standard ISO opracowany dla dokumentów elektronicznych, liczne zastosowania przemysłowe.

Parametry obrazu

Ziarnistość obrazu - ze sposobu działania monitora wynika, że wszystko, co daje się na nim przedstawić, jest ostatecznie mapą bitową. Może więc być utrwalone i obrabiane dalej jako mapa bitowa. Także grafika wektorowa (zapisująca obraz jako formuły kreślenia krzywych) ostatecznie interpretowana jest na monitorze w postaci układu pikseli. Ograniczona rozdzielczość obrazu na monitorze, w porównaniu z drukiem lub zwykłym rysunkiem na papierze, przeszkadza w odtworzeniu drobnych i kontrastowych szczegółów źle wpisujących się w rzędy i kolumny pikseli.

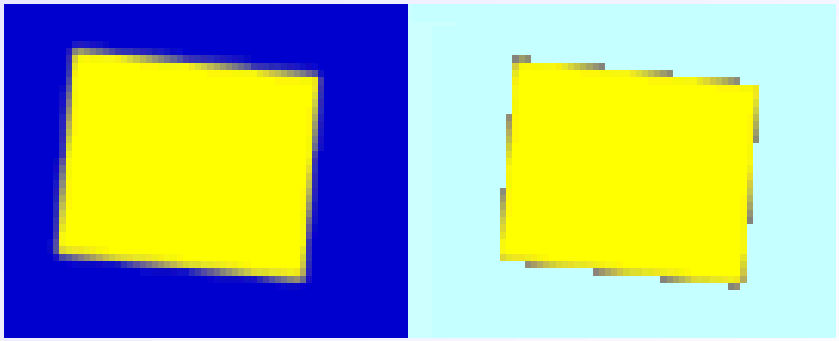
Parametry obrazu



Na rysunku powyżej widać, jak prosta mapa bitowa doskonale oddaje krawędzie pionowe i poziome oraz nachylone 45°. Przy wszystkich innych kątach krawędzi ujawnia się ziarnista struktura pikseli.

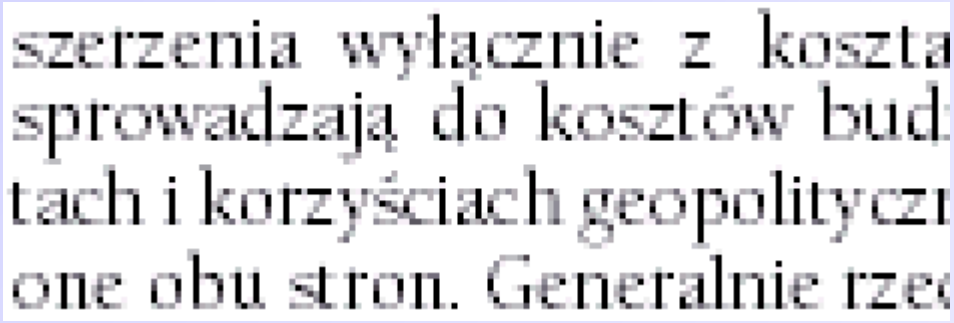
Parametry obrazu

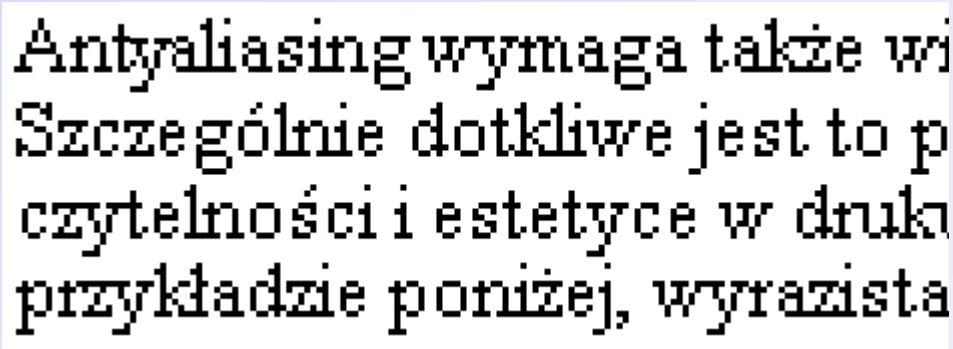
Rozwiązanie tego problemu nazywa się **antyaliasingiem** (wygładzaniem konturów) i polega na tym, że można symulować zabarwienie części granicznego piksela, przez odpowiednie rozmycie barwy całego piksela. Ludzkie oko, które trudno rozróżnia jeden piksel, daje się oszukiwać i ziarnistość osnowy obrazu zostaje ukryta. Antyaliasing barwnych konturów wymaga znacznego powiększenia palety barw. Przy zredukowanej palecie barw działanie antyaliasingu może być bardzo ograniczone.



Parametry obrazu

Antyaliasing najlepiej działa na szeregu wielu pikseli, dlatego nie może pomóc w odtworzeniu niewielkich szczegółów.

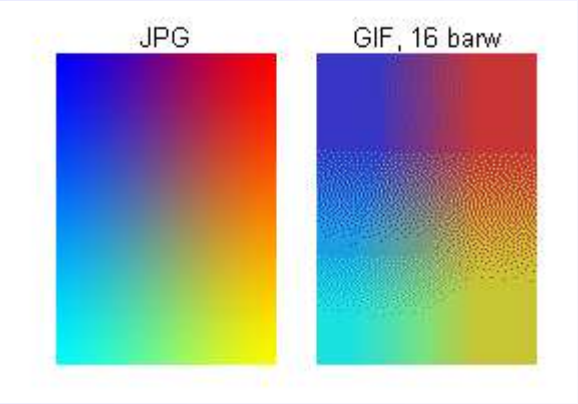




Szczególnie dotkliwe jest to przy lekturze tekstu przygotowanego do druku, a przeniesionego na ekran monitora, jak choćby w popularnej technologii PDF. W celu poprawy jakości stosuje specjalne czcionki ekranowe dające optymalny obraz na niewielu pikselach.

Parametry obrazu

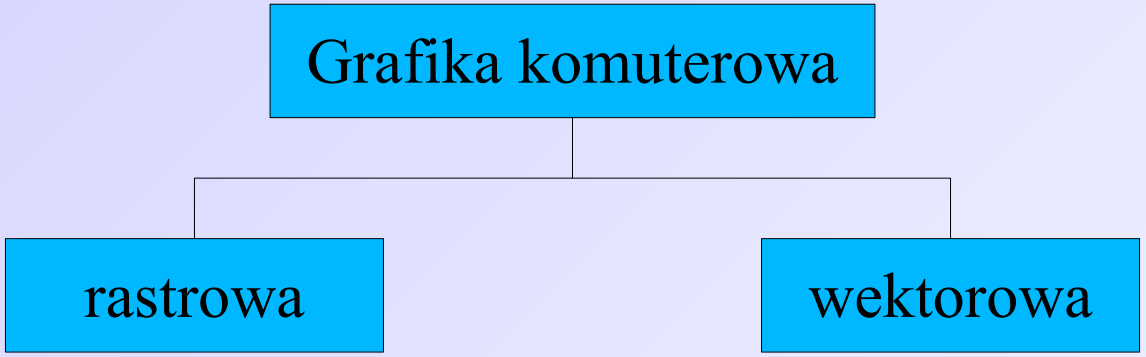
Ziarnistość osnowy obrazu może być wykorzystana do symulowania większej ilości barw niż faktycznie jest w palecie. **Dithering** (roztrząsanie, prószenie), to sposób symulowania przejść tonalnych między barwami przez przemieszanie pojedynczych pikseli barw skrajnych. To właśnie dzięki tej technice większość fotografii z natury daje się przedstawić z ograniczoną paletą barw w kompresji **GIF**.

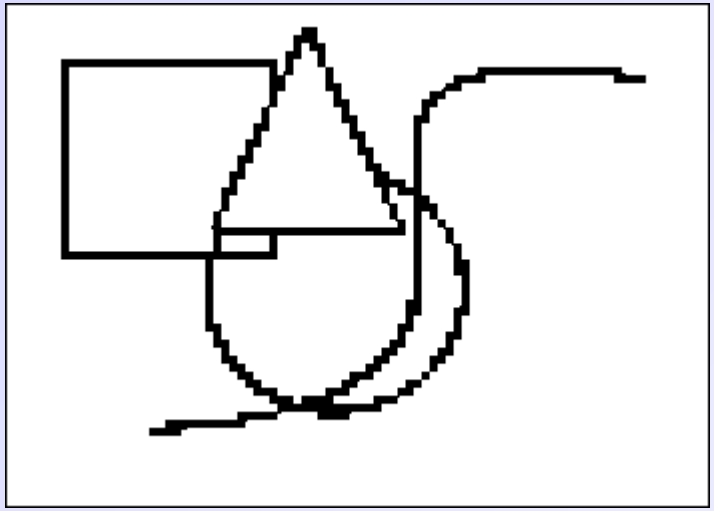


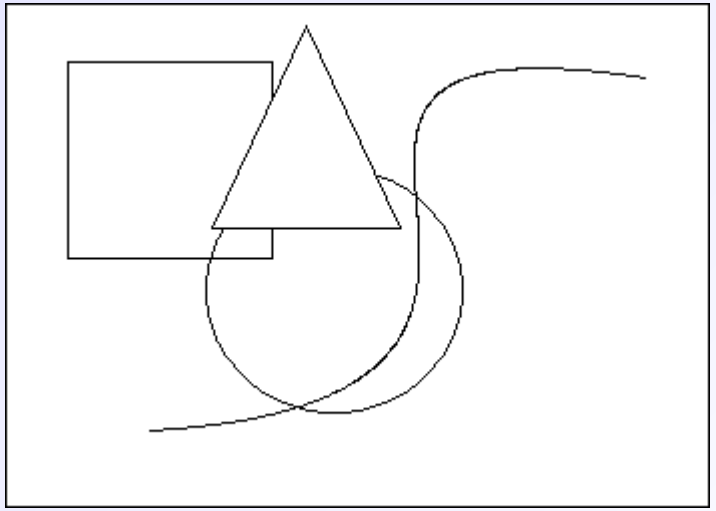
Grafika komputerowa

Grafika komputerowa – jest to obraz tworzony lub poddawany obróbce za pomocą komputera. Istnieje rozróżnienie pomiędzy grafiką rastrową i wektorową. Bitmapy to obrazy będące zbiorami punktów – pikseli w różnych kolorach. Natomiast pliki grafiki wektorowej zawierają nie sam obraz, a instrukcje, które wykorzystywane są przez komputer do budowania grafiki na ekranie.

Grafika komputerowa







Grafika komputerowa

Cechy grafiki wektorowej:

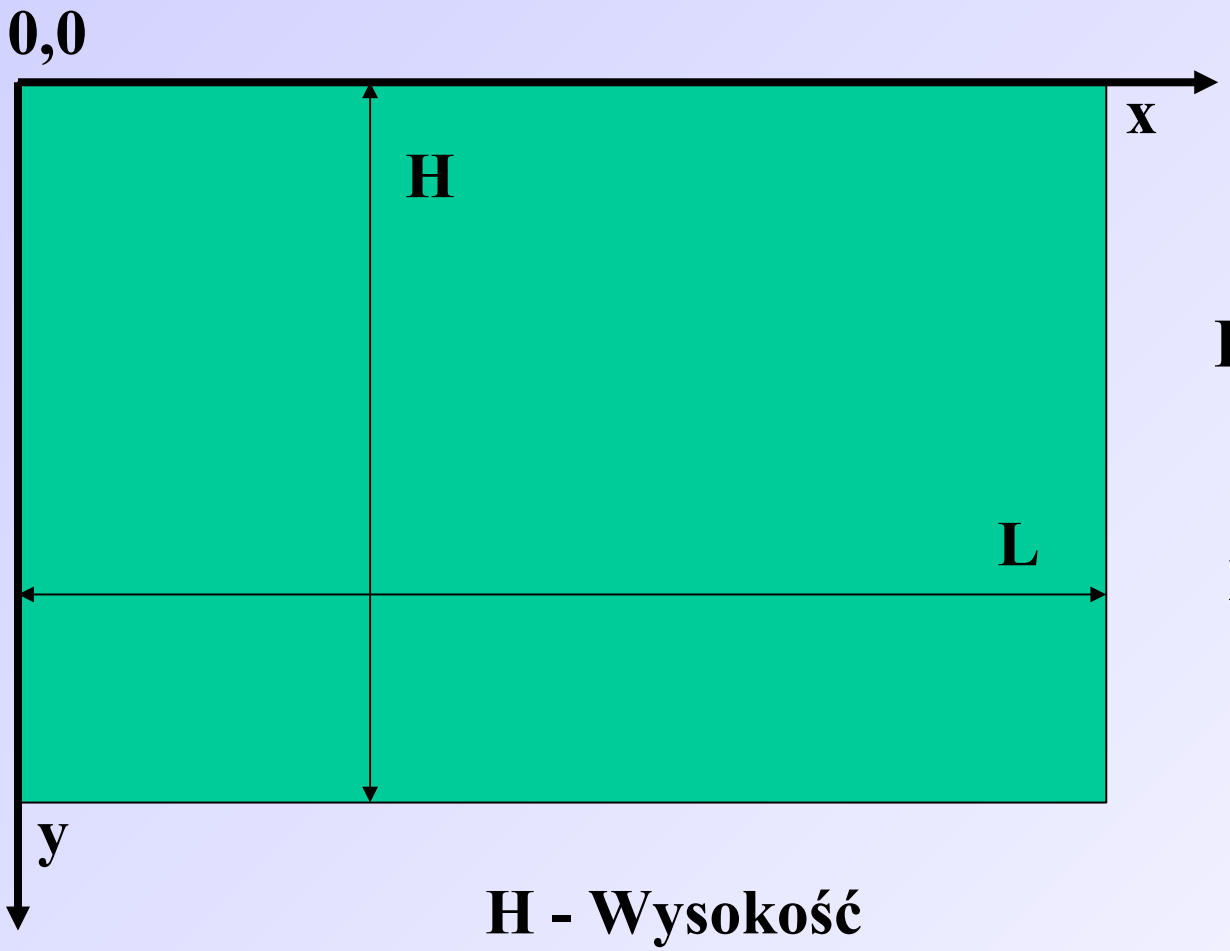
obraz złożony z wektorów (odcinek kodują współrzędne początku, końca i barwa),
mała zajętość pamięci, łatwość modyfikacji, analityczny opis,
konieczność konwersji do urządzeń wyjściowych,
rysunek w formacie wektorowym zajmuje znacznie mniej miejsca, niż w postaci bitmapy, ale zdjęcia lepiej zapisywać jako bitmapy,
grafikę wektorową można przeskalowywać (oraz deformować) bez utraty jakości.

Grafika komputerowa

Cechy grafiki rastrowej:

obraz złożony z kropek (pikseli), zwany bitmapą,
„naturalne” dopasowanie do urządzeń wyjściowych, łatwość implementacji,
barwa każdego piksela kodowana na określonej ilości bitów,
duże zapotrzebowanie na pamięć, trudne modyfikacje, duże moce obliczeniowe do przekształceń,
przy powiększaniu rozmiarów bitmapy jakość się pogarsza.

Transformacje

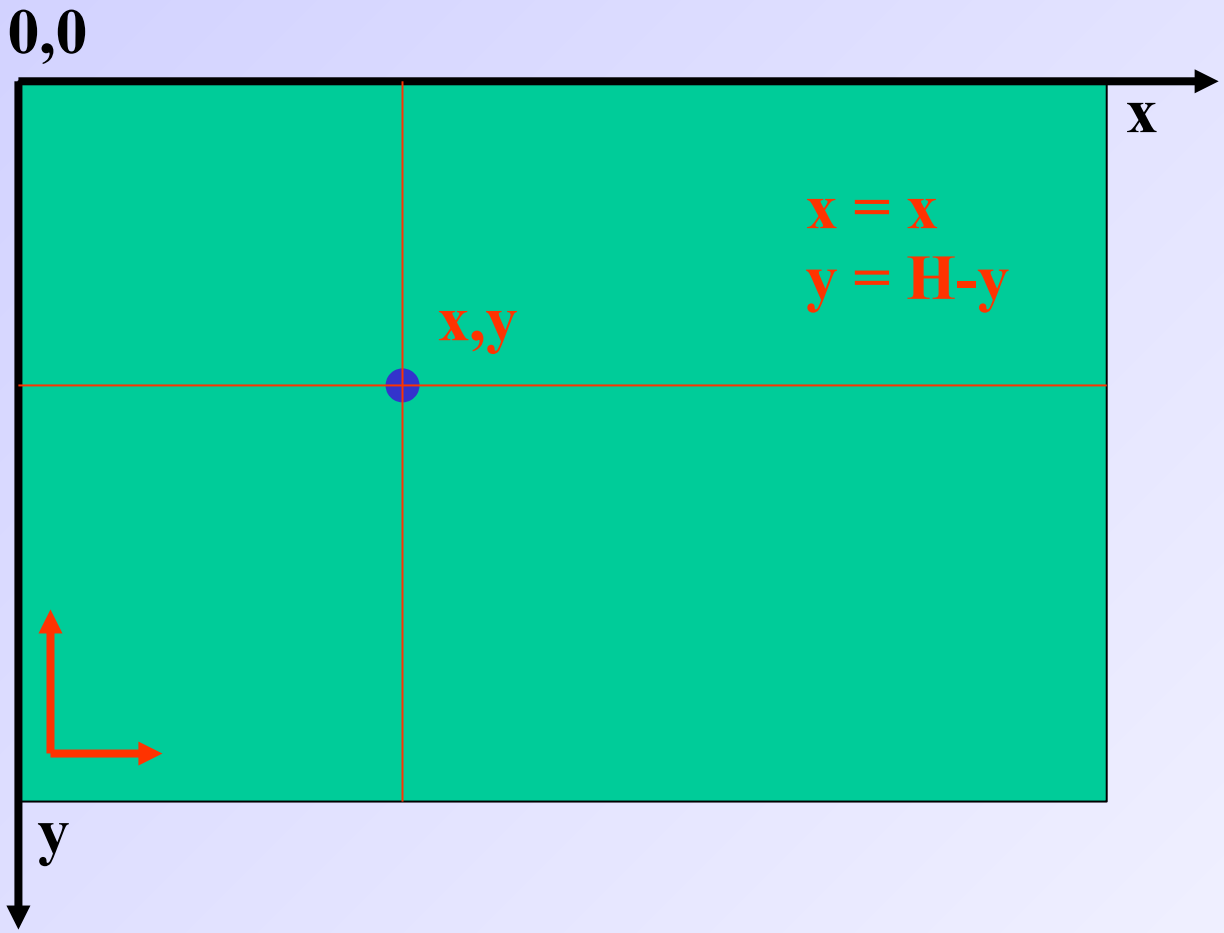


H - Wysokość
L - Szerokość

Podstawowe operacje:

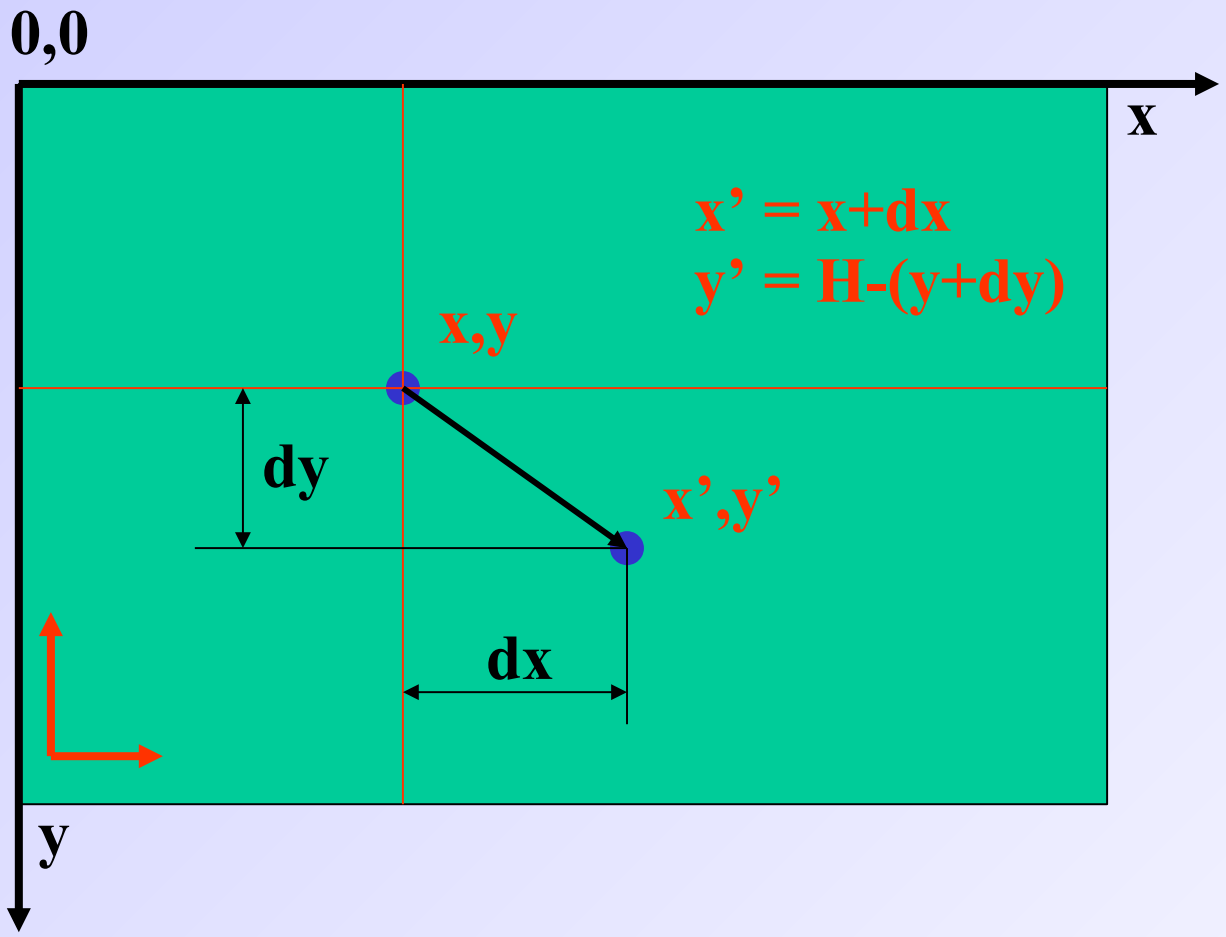
odwrócenie osi pionowej,
przesunięcie,
skalowanie,
obrót.

Transformacje



Odwrócenie osi pionowej.

Transformacje



$$\begin{cases} x' = x + dX \\ y' = y + dY \end{cases}$$

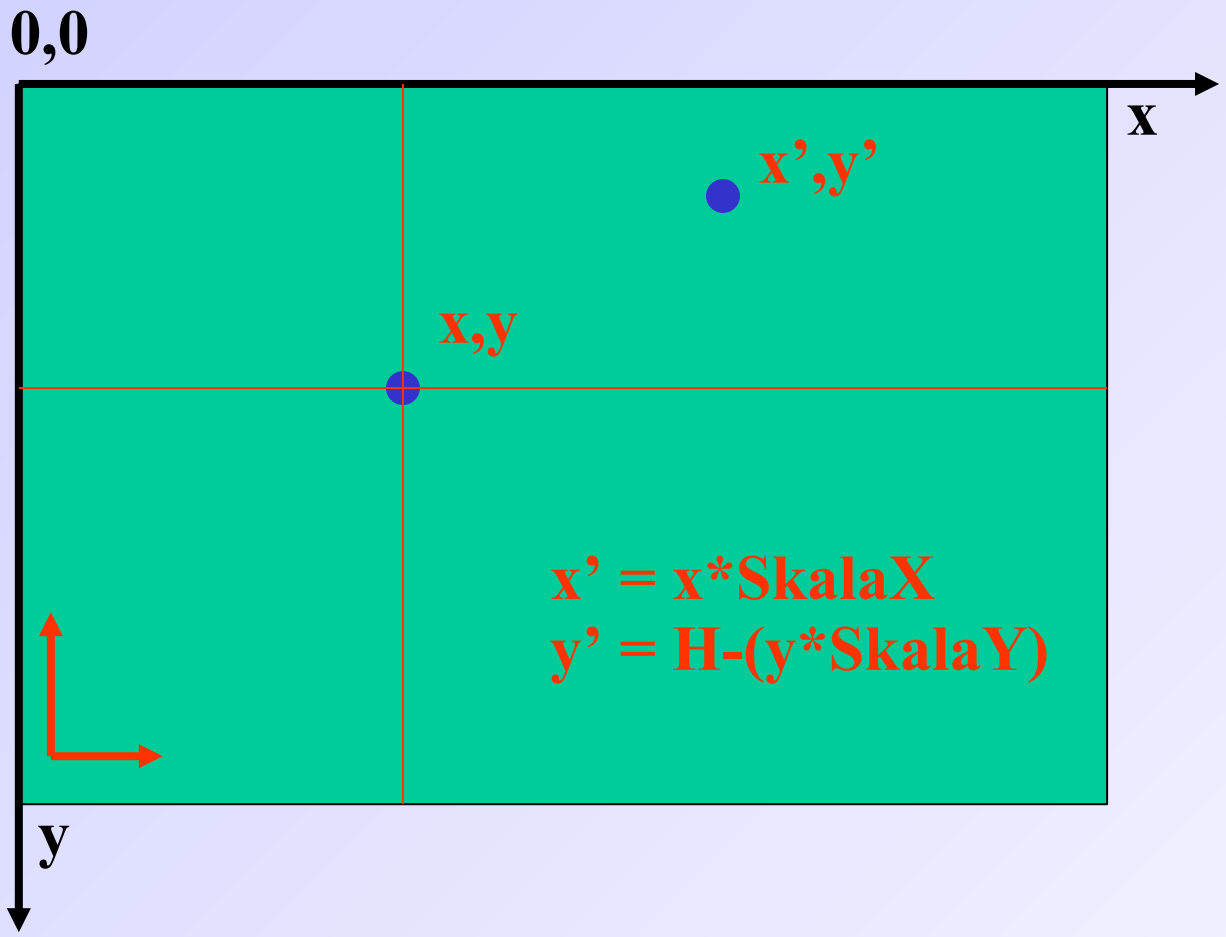
$$P = \begin{bmatrix} x \\ y \end{bmatrix} \qquad P' = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

$$T = \begin{bmatrix} dX \\ dY \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} dX \\ dY \end{bmatrix}$$

Przesunięcie.

Transformacje



$$\begin{cases} x' = x \cdot s_x \\ y' = y \cdot s_y \end{cases}$$

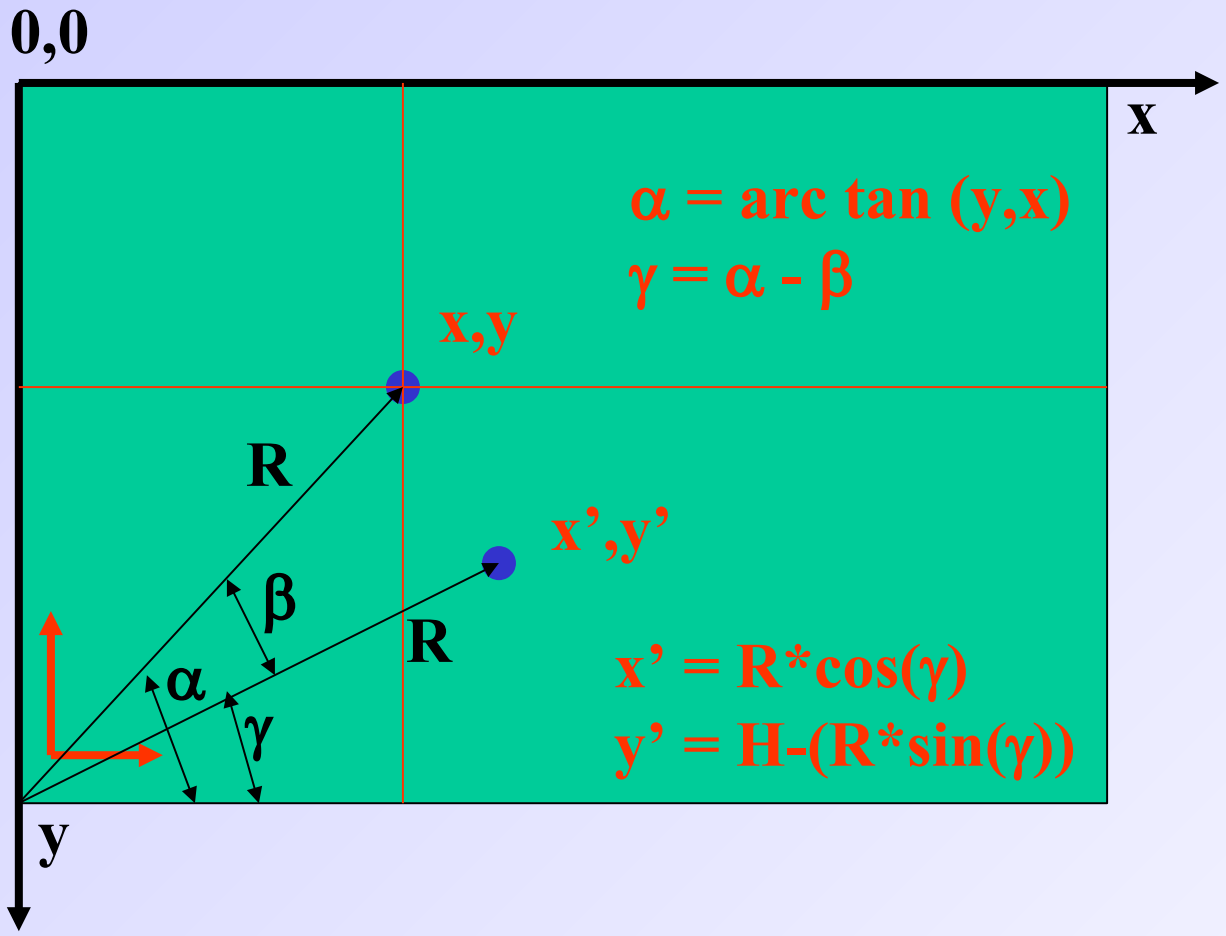
$$P = \begin{bmatrix} x \\ y \end{bmatrix} \qquad P' = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

$$T = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} \cdot \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$$

Skalowanie.

Transformacje



$$\begin{cases} x' = r \cdot \cos(\alpha) \\ y' = r \cdot \sin(\alpha) \end{cases}$$

$$P = \begin{bmatrix} x \\ y \end{bmatrix} \qquad P' = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

$$T = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix}$$

Obrót.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} \cdot \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix}$$

Transformacje

Jeżeli transformacja ma być dokonana względem punktu nie będącego początkiem układu współrzędnych to należy:

zapamiętać współrzędne środka obiektu (x_s,y_s),
przesunąć środek obiektu do punktu (0,0),
dokonać skalowania lub obrotu,
przesunąć środek obiektu do punktu (x_s,y_s).

Implementacja

Definicja zmiennych:

```
Xr : Array of Real;
Yr : Array of Real;

X0 : Integer;
Y0 : Integer;

n : Integer;
```

Implementacja

```
procedure TFormGlowny.Oblicz;
begin
    try
            n:=5;                       - liczba punktów do wyświetlenia
            SetLength(Xr,N+1);          - ustawienie długości tablicy współrzędnych X wszystkich punktów
            SetLength(Yr,N+1);          - ustawienie długości tablicy współrzędnych Y wszystkich punktów
            X0:=Obraz.Width div 2;      - określenie współrzędnej X środka ekranu
            Y0:=Obraz.Height div 2;     - określenie współrzędnej X środka ekranu
            Xr[1]:=X0-100;              - współrzędne prostokąta
            Yr[1]:=Y0-50;               - współrzędne prostokąta
            Xr[2]:=X0+100;              - współrzędne prostokąta
            Yr[2]:=Y0-50;               - współrzędne prostokąta
            Xr[3]:=X0+100;              - współrzędne prostokąta
            Yr[3]:=Y0+50;               - współrzędne prostokąta
            Xr[4]:=X0-100;              - współrzędne prostokąta
            Yr[4]:=Y0+50;               - współrzędne prostokąta
            Xr[5]:=X0-100;              - współrzędne prostokąta
            Yr[5]:=Y0-50;               - współrzędne prostokąta
    except
    end;
end;
```

Implementacja

```
procedure TFormGlowny.Przesuniecie2D(X,Y : Real);
var
  i : Integer;
begin
     try
               X0:=Round(X0+X);              - obliczenie nowego środka prostokąta
               Y0:=Round(Y0+Y);              - obliczenie nowego środka prostokąta
               for i:=1 to N do
                  begin
                  Xr[i]:=Round(Xr[i]+X);     - przesunięcie wszystkich współrzędnych x o wartość X
                  Yr[i]:=Round(Yr[i]+Y);     - przesunięcie wszystkich współrzędnych y o wartość Y
                  end;
     except
     end;
end;
```

Implementacja

```
procedure TFormGlowny.Skalowanie2D(Skala : Real);
var
 i : Integer;
 Temp1 : Real;
 Temp2 : Real;
begin
      try
            Temp1:=X0;                          - zapamiętanie współrzędnej X środka prostokąta
            Temp2:=Y0;                          - zapamiętanie współrzędnej Y środka prostokąta
            Przesuniecie2D(-Temp1,-Temp2);      - przesunięcie do początku układu współrzędnych
            for i:=1 to N do
               begin
               Xr[i]:=Round(Xr[i]*Skala);       - skalowanie współrzędnej X
               Yr[i]:=Round(Yr[i]*Skala);       - skalowanie współrzędnej Y
               end;
            Przesuniecie2D(Temp1,Temp2);        - przesunięcie do położenia pierwotnego
    except
    end;
end;
```

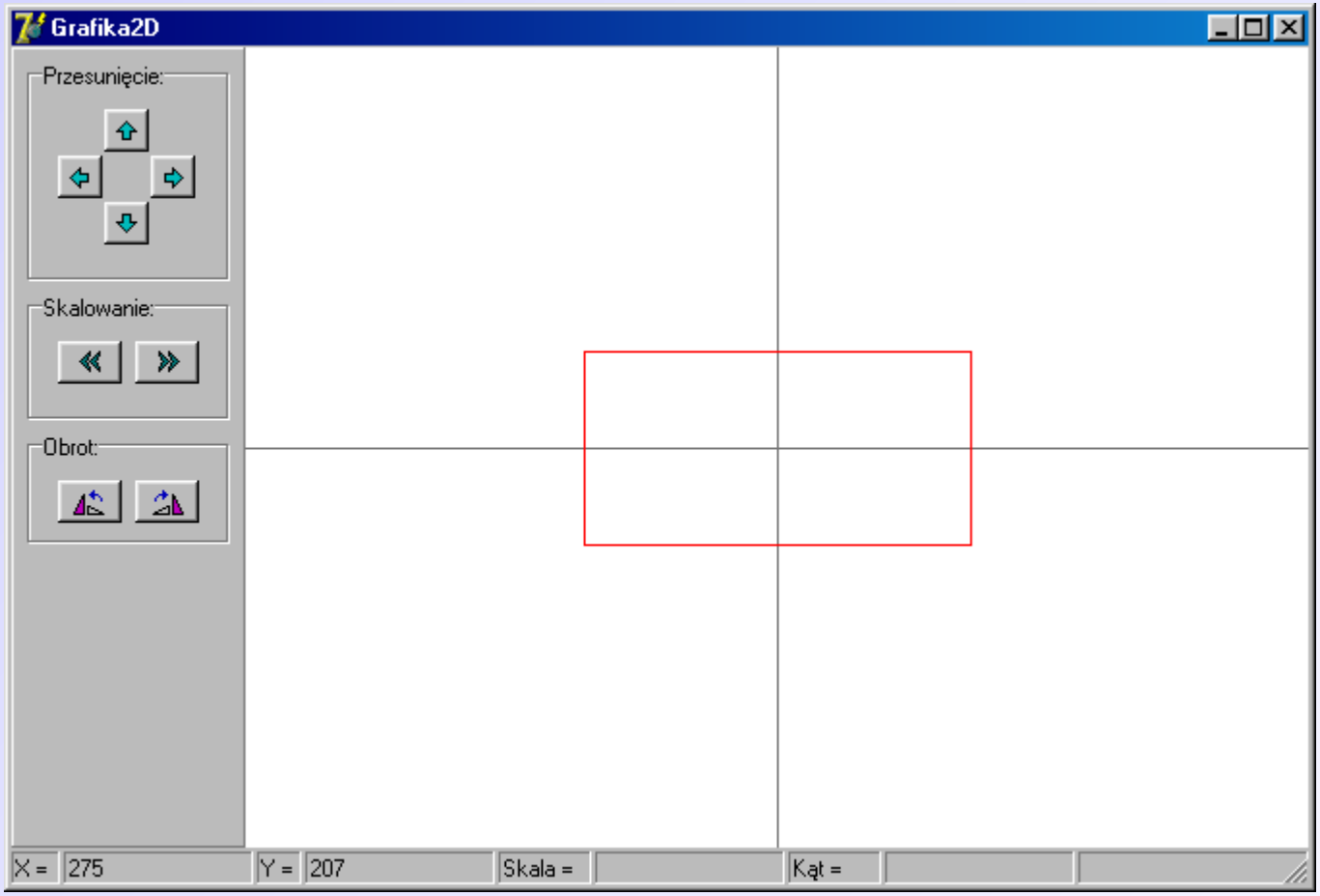
Implementacja

```
procedure TFormGlowny.Obrot2D(Kat : Real);
var
 Alfa : Real;
 R : Real;
 i : Integer;
 Temp1 : Real;
 Temp2 : Real;
begin
    try
            Temp1:=X0;                          - zapamiętanie współrzędnej X środka prostokąta
            Temp2:=Y0;                          - zapamiętanie współrzędnej Y środka prostokąta
            Przesuniecie2D(-Temp1,-Temp2);      - przesunięcie do początku układu współrzędnych
            for i:=1 to N do
               begin                            - obliczenie bieżącego kąta wektora wodzącego punktu
               if Xr[i]>= 0 then Alfa:=ArcTan(Yr[i]/Xr[i]) else Alfa:=ArcTan(Yr[i]/Xr[i])+3.1415926;
               R:=SQRT((Xr[i])*(Xr[i])+(Yr[i])*(Yr[i]));        - obliczenie odległości od środka układu
               Xr[i]:=R*cos(Alfa+Kat);                          - wyznaczenie rzutu R na os X
               Yr[i]:=R*sin(Alfa+Kat);          - wyznaczenie rzutu R na os Y
               end;
               Przesuniecie2D(Temp1,Temp2);     - przesunięcie do położenia pierwotnego
    except
    end;
end;
```

Implementacja

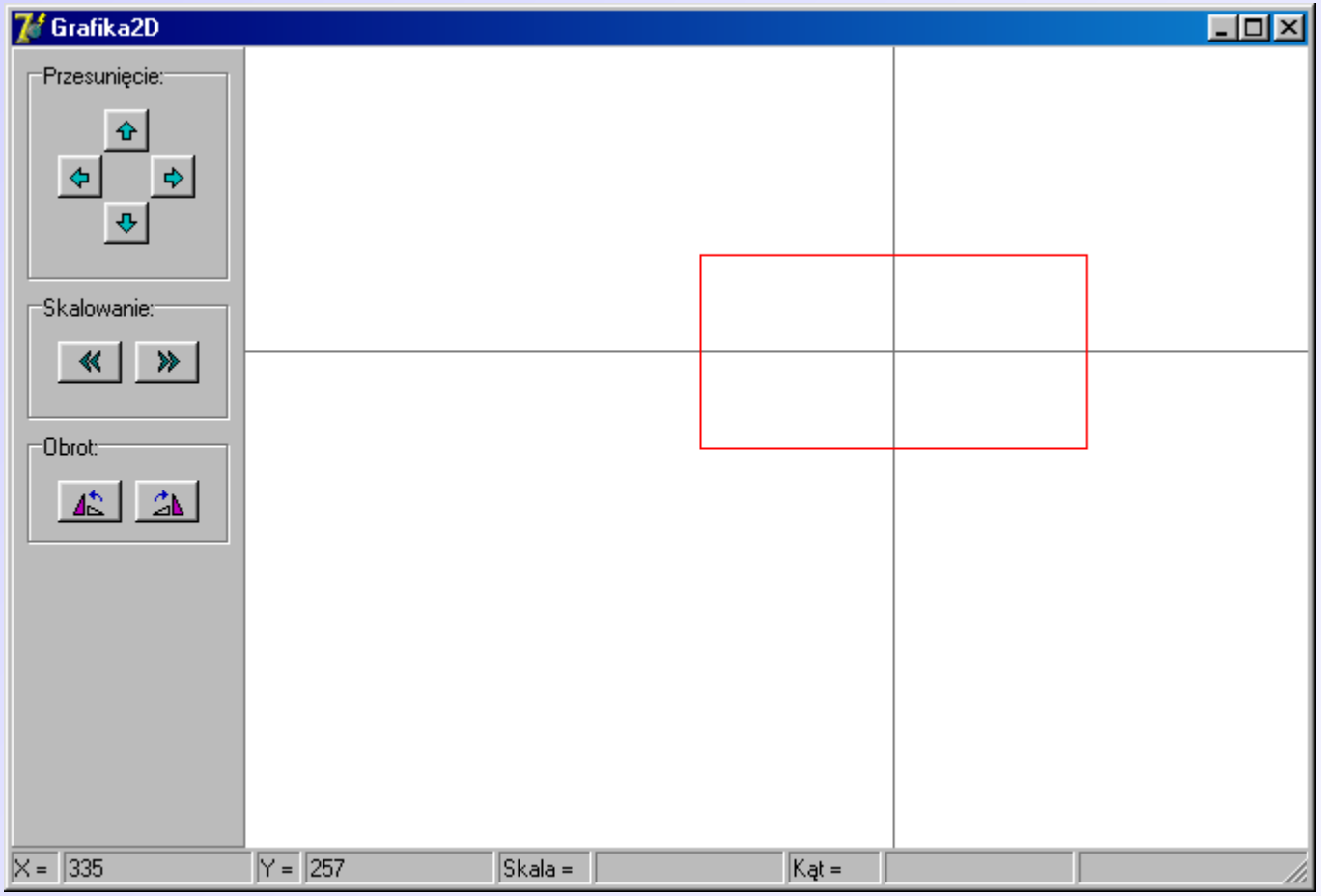
```
procedure TFormGlowny.Rysuj;
var
  i : Integer;
begin
    try
            RysujPustaBitmape;                                  - inicjacja bitmapy
            Obraz.Canvas.Pen.Color:=clGray;                     - określenie koloru wskaźnika środka (x0,y0)
            Obraz.Canvas.MoveTo(X0,0);                          - linie wskazujące punkt (x0,y0)
            Obraz.Canvas.LineTo(X0,Obraz.Height);               - linie wskazujące punkt (x0,y0)
            Obraz.Canvas.MoveTo(0,Obraz.Height-Y0);             - linie wskazujące punkt (x0,y0)
            Obraz.Canvas.LineTo(Obraz.Width,Obraz.Height-Y0);   - linie wskazujące punkt (x0,y0)
            Obraz.Canvas.Pen.Color:=clRed;                      - zmiana koloru linii
            for i:=1 to N do                                    - rysowanie prostokąta
               begin
               if  i = 1 then Obraz.Canvas.MoveTo(Round(Xr[i]),Round(Obraz.Height-Yr[i])) else
                              Obraz.Canvas.LineTo(Round(Xr[i]),Round(Obraz.Height-Yr[i]));
               end;
    except
    end;
end;
```

Implementacja



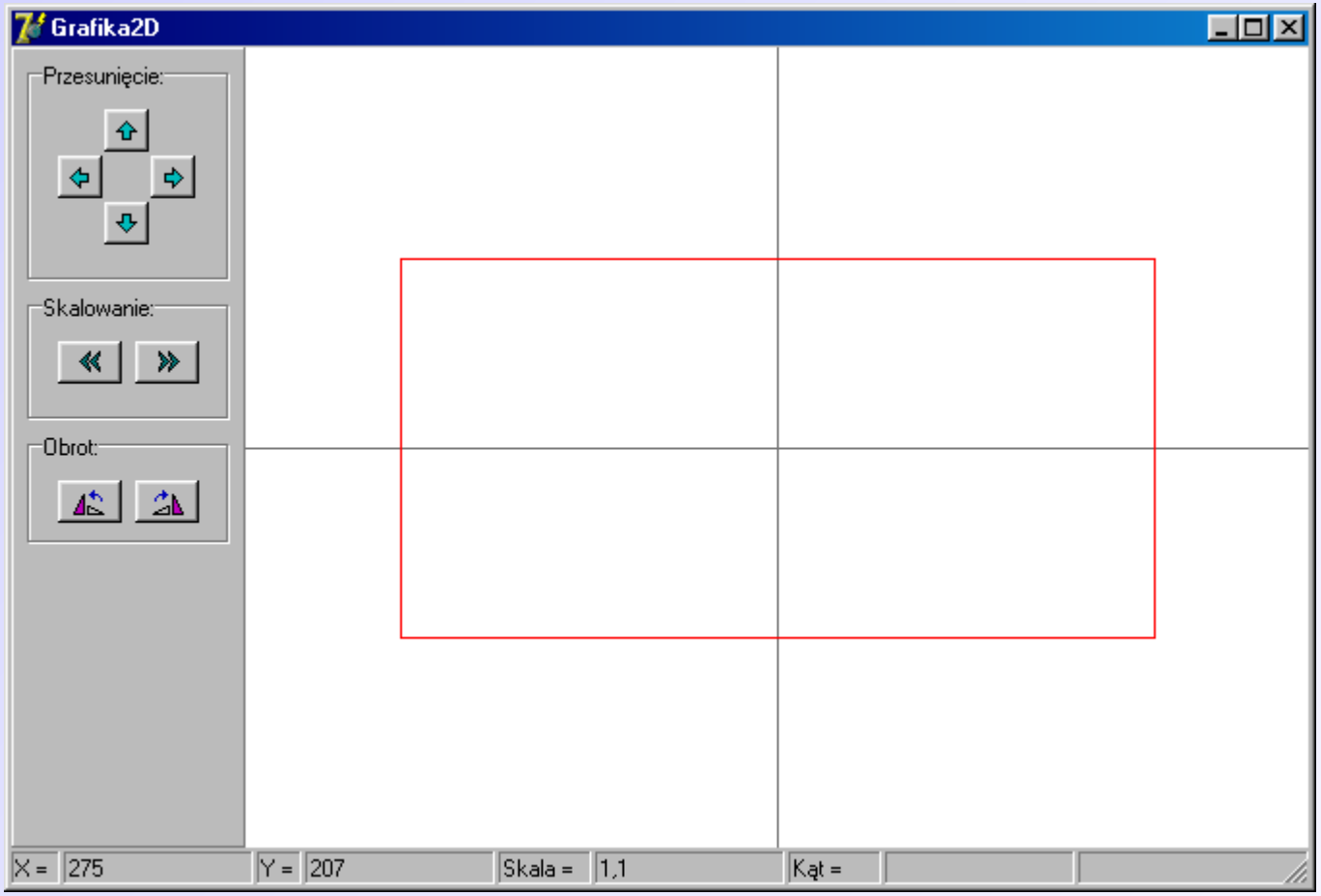
Rysunek podstawowy.

Implementacja



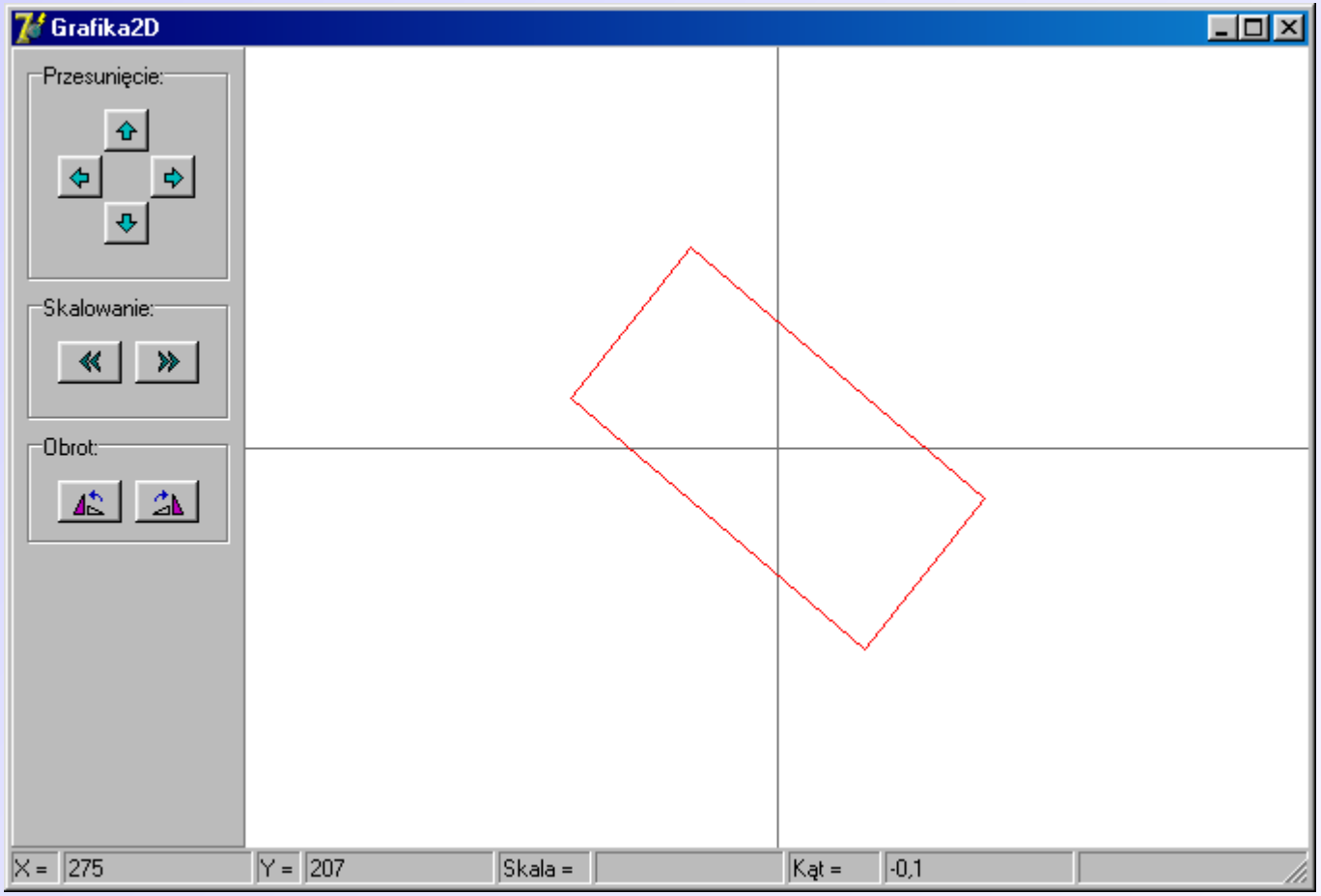
Przesunięcie.

Implementacja



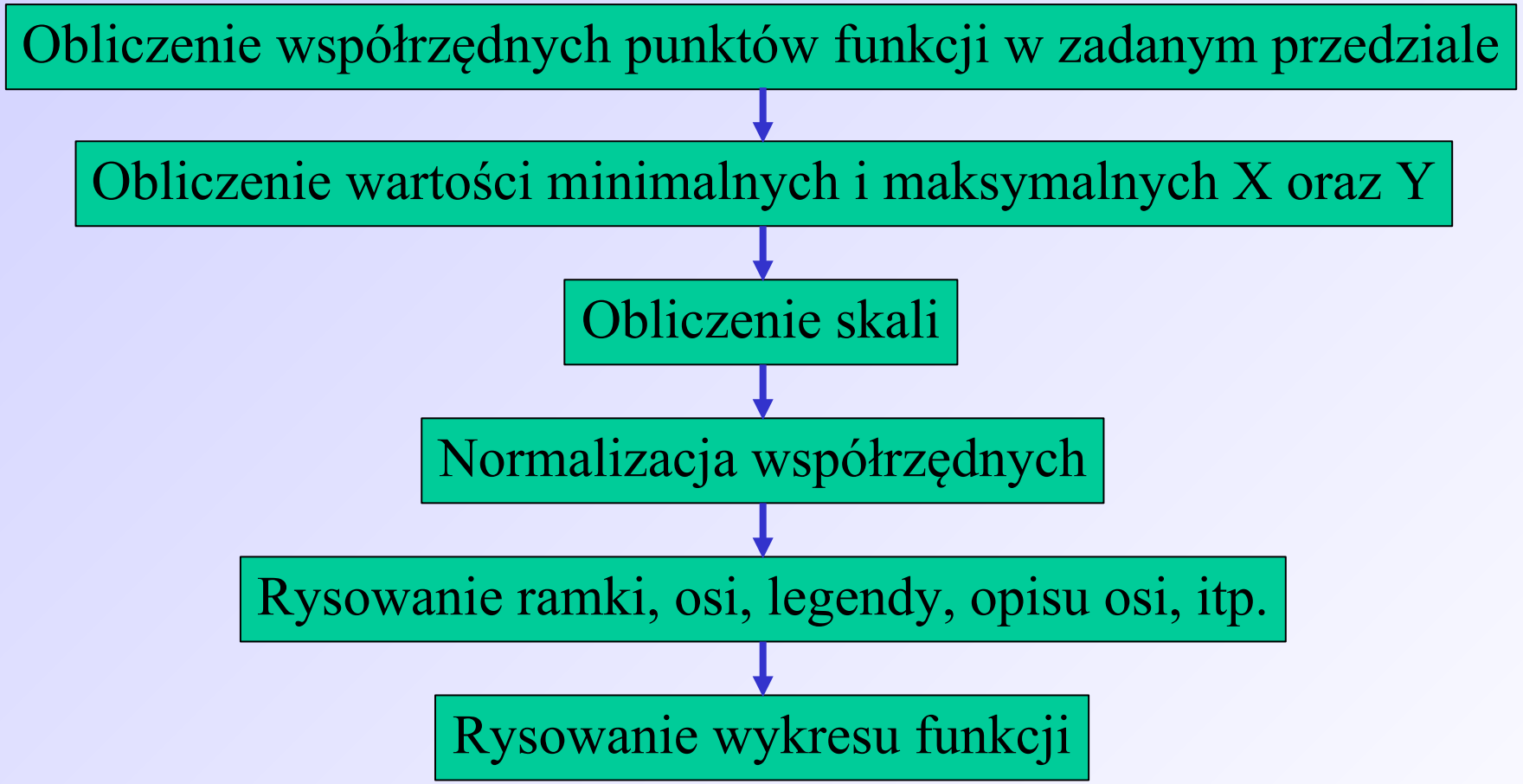
Skalowanie.

Implementacja



Obrót.

Wykresy funkcji



Normalizacja

Normalizacja służy dopasowaniu obiektu do obszaru rysunkowego:

$$SkalaX = \frac{L - 2 * M}{X_{\max} - X_{\min}}$$

$$SkalaY = \frac{H - 2 * M}{Y_{\max} - Y_{\min}}$$

gdzie:

L - szerokość obszaru
H - wysokość obszaru
M - margines

Normalizacja

Normalizacja służy dopasowaniu obiektu do obszaru rysunkowego:

$$X_e = X_r \cdot SkalaX + M - X_{\min} \cdot SkalaX$$

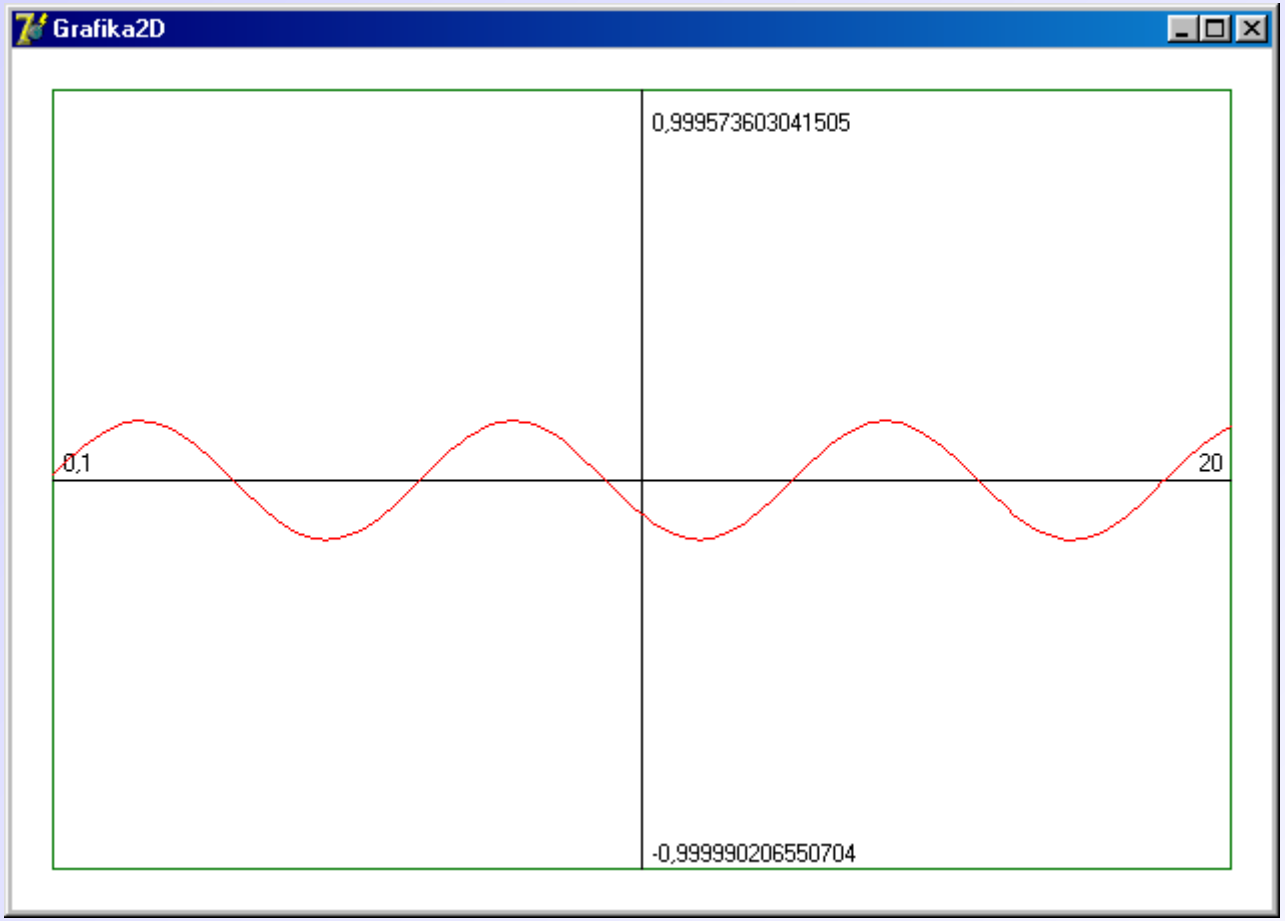
$$Y_e = H - (Y_r \cdot SkalaY + M - Y_{\min} \cdot SkalaY) - (H - 2 \cdot M) + \frac{Y_{\max} - Y_{\min}}{2} \cdot SkalaY$$

Gdzie X_e, Y_e – współrzędne ekranowe.

Implementacja

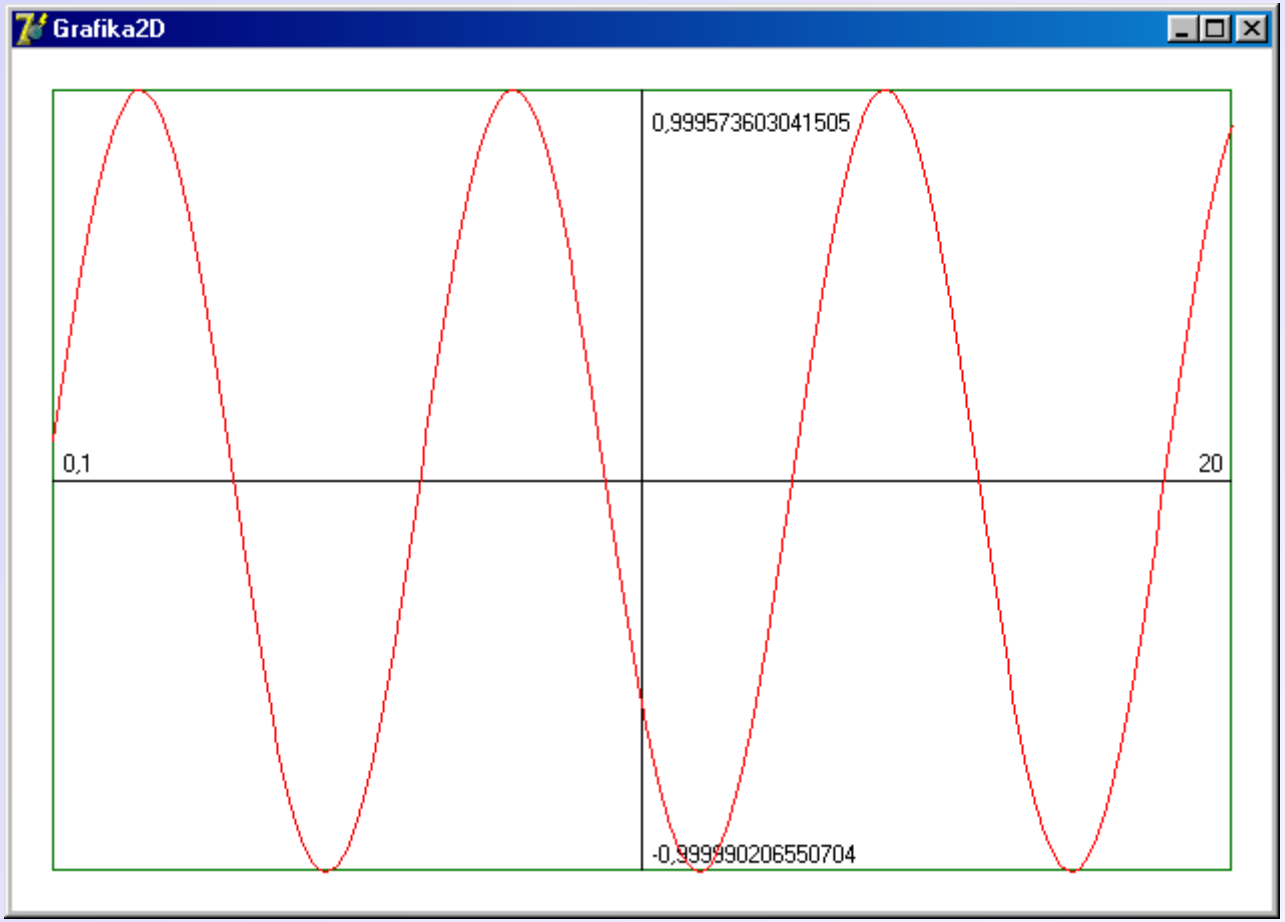
```
for i:=1 to N do
   begin
   if CzyProporcje = True then
     begin
     if SkalaY >= SkalaX then
       begin
       Xe[i]:=Round(Xr[i]*SkalaX+Margines-X_Min*SkalaX);
       Ye[i]:=Round((FormGlowny.Obraz.Height)-(Yr[i]*SkalaX+Margines-Y_Min*SkalaX)-
      ((FormGlowny.Obraz.Height-2*Margines) div 2)+((Y_Max-Y_Min)/2)*SkalaX);
       end
     else
       begin
       Xe[i]:=Round(Xr[i]*SkalaY+Margines-X_Min*SkalaY);
       Ye[i]:=Round((FormGlowny.Obraz.Height)-(Yr[i]*SkalaY+Margines-Y_Min*SkalaY)-
       ((FormGlowny.Obraz.Height-2*Margines) div 2)+((Y_Max-Y_Min)/2)*SkalaX);
       end;
     end
   else
     begin
     Xe[i]:=Round(Xr[i]*SkalaX+Margines-X_Min*SkalaX);
     Ye[i]:=Round((FormGlowny.Obraz.Height)-(Yr[i]*SkalaY+Margines-Y_Min*SkalaY));
     end;
   end;
```

Implementacja



Wykres funkcji – skalowanie proporcjonalne.

Implementacja



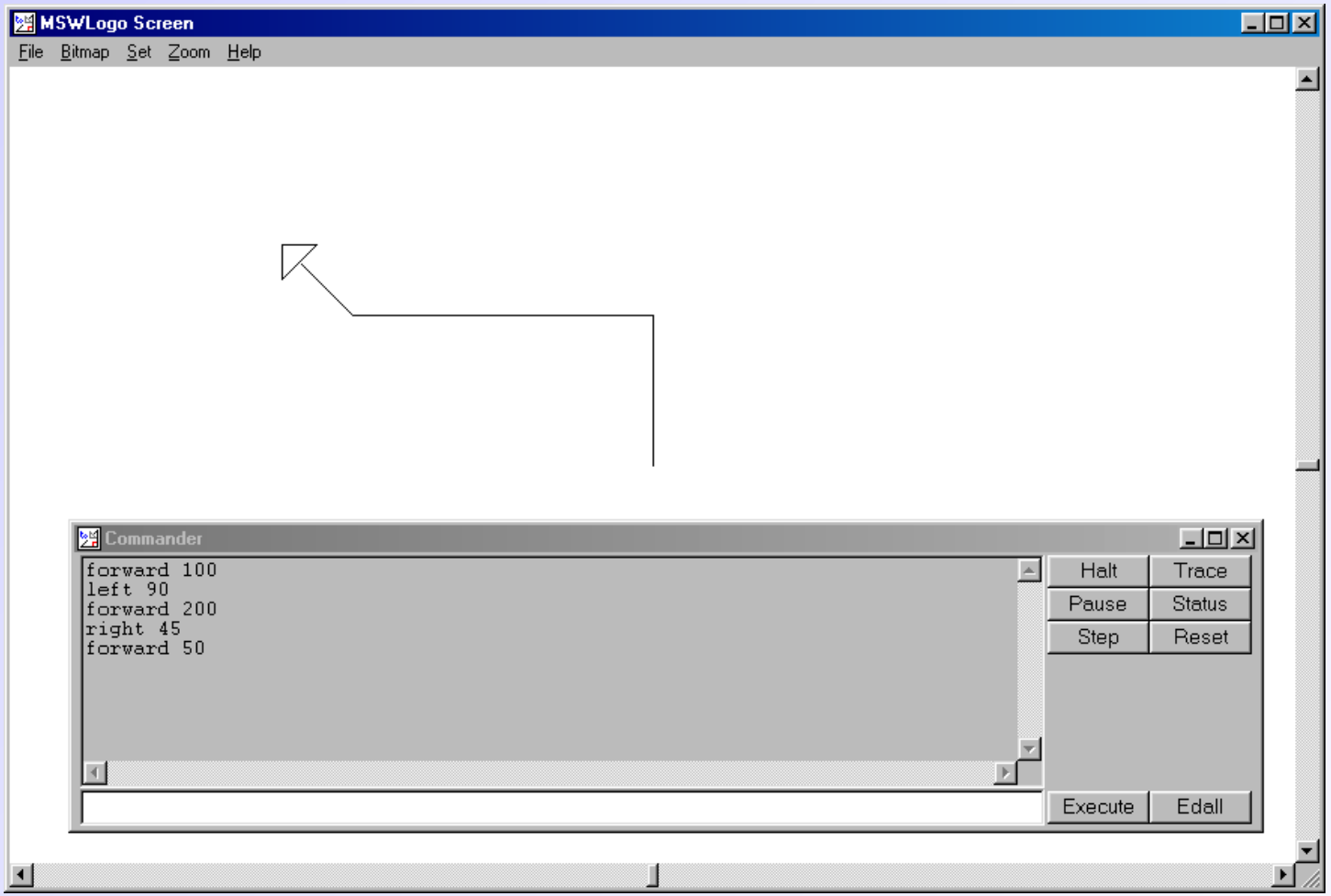
Wykres funkcji – skalowanie nieproporcjonalne.

Grafika żółwia

Grafika „żółwia” jest techniką rysowania krzywych, w której zastosowano pojęcie względnego opisu ruchu oraz rysowania we współrzędnych biegunowych. Proces rysowania obiektów opisany jest tutaj za pomocą kierunku oraz długości nowo powstających odcinków. Takie podejście pozwala na rysowanie dowolnych wielokątów na płaszczyźnie, a także w przestrzeni 2D.

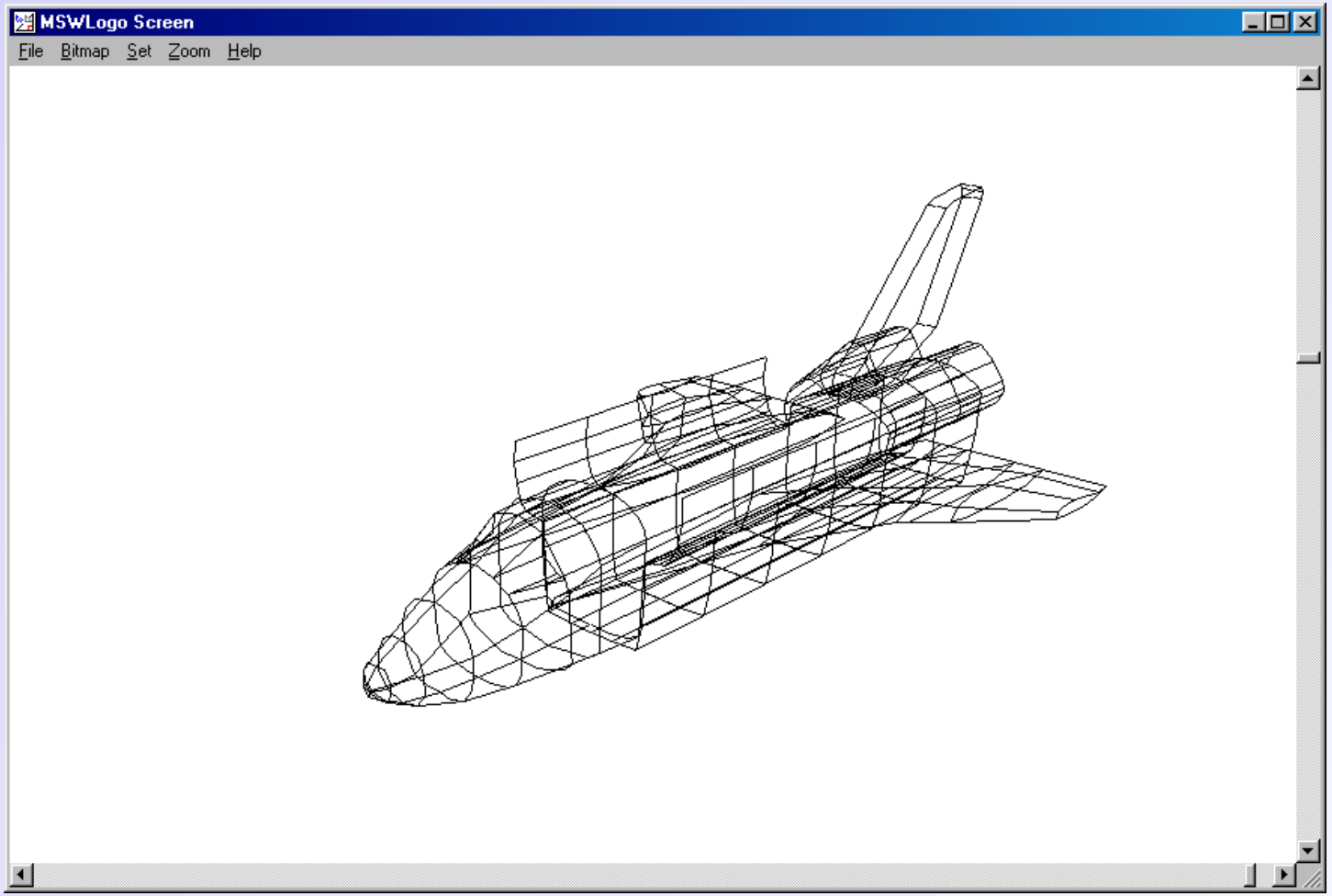
Grafika „żółwia” jest podstawową techniką wykorzystywaną w edukacyjnym języku LOGO.

Grafika żółwia



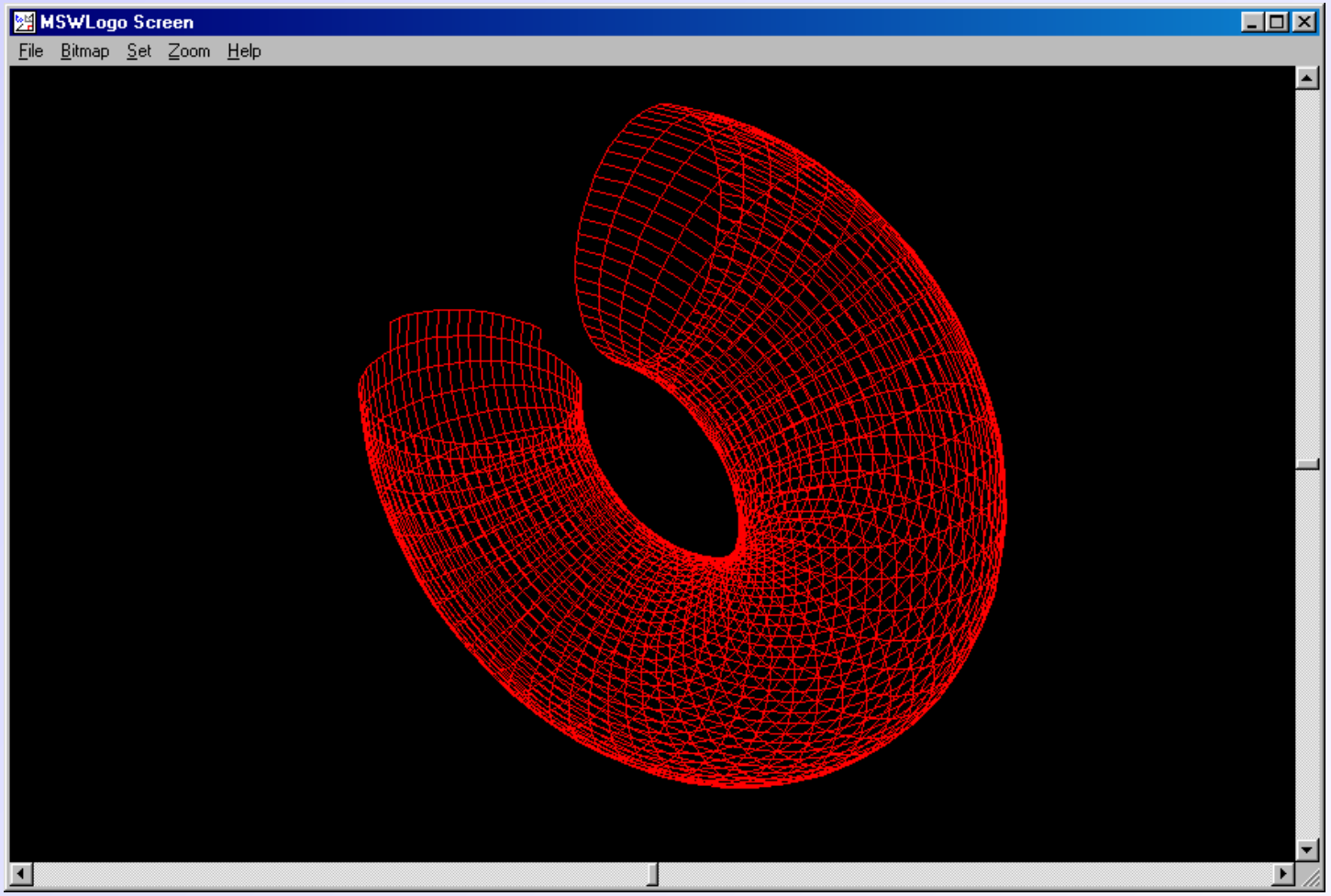
MSW Logo – przykłady grafiki żółwia.

Grafika żółwia



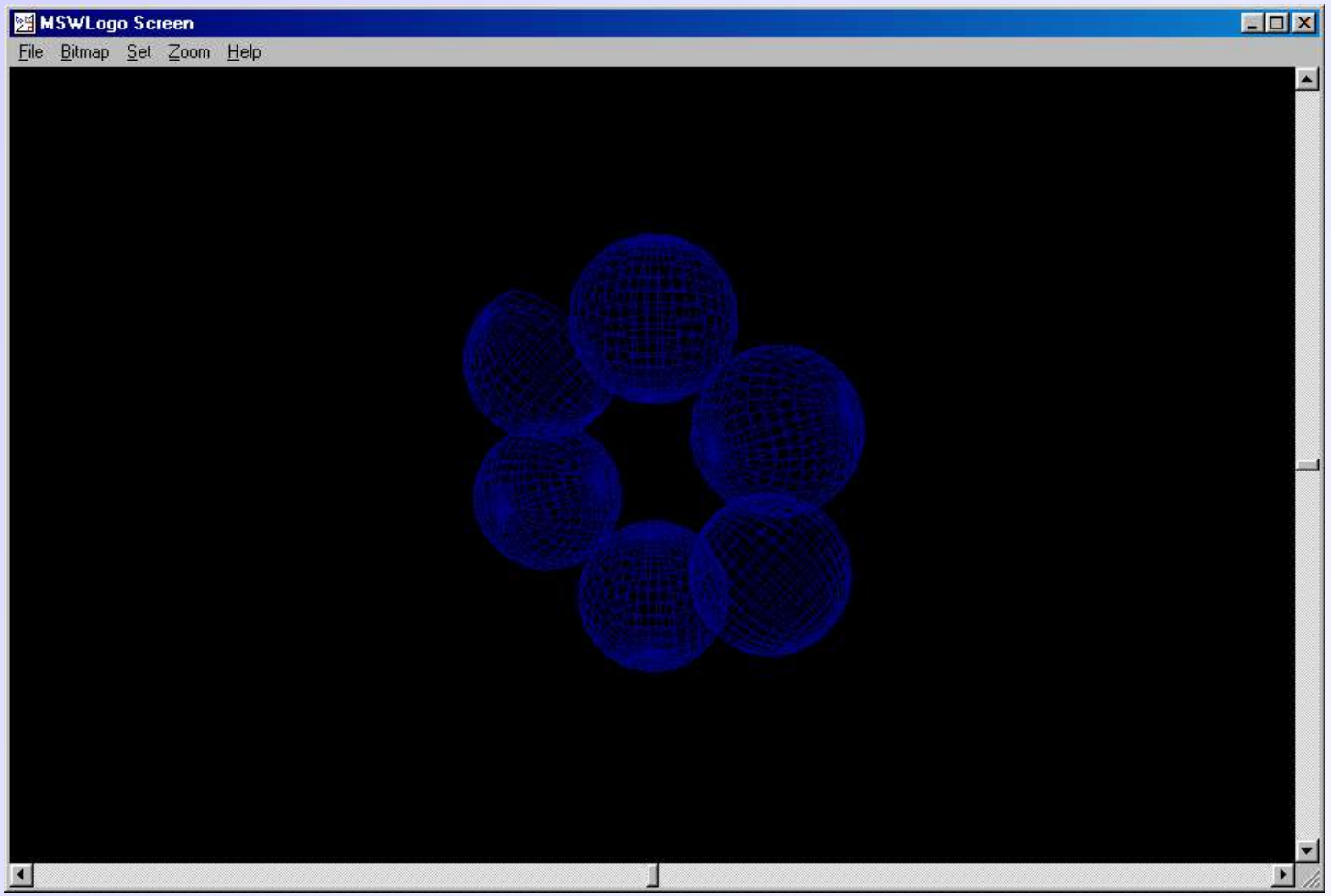
MSW Logo – przykłady grafiki żółwia.

Grafika żółwia



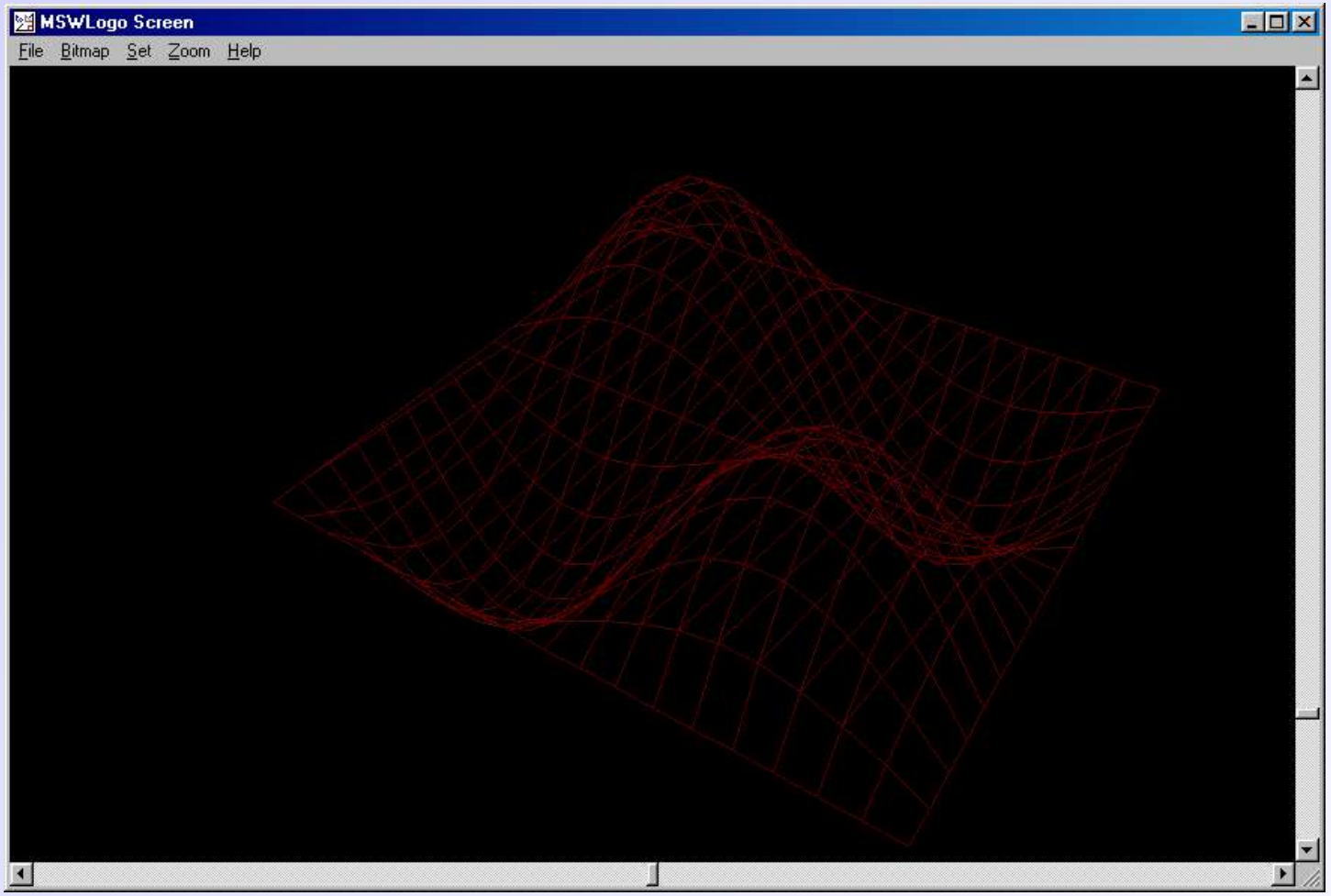
MSW Logo – przykłady grafiki żółwia.

Grafika żółwia



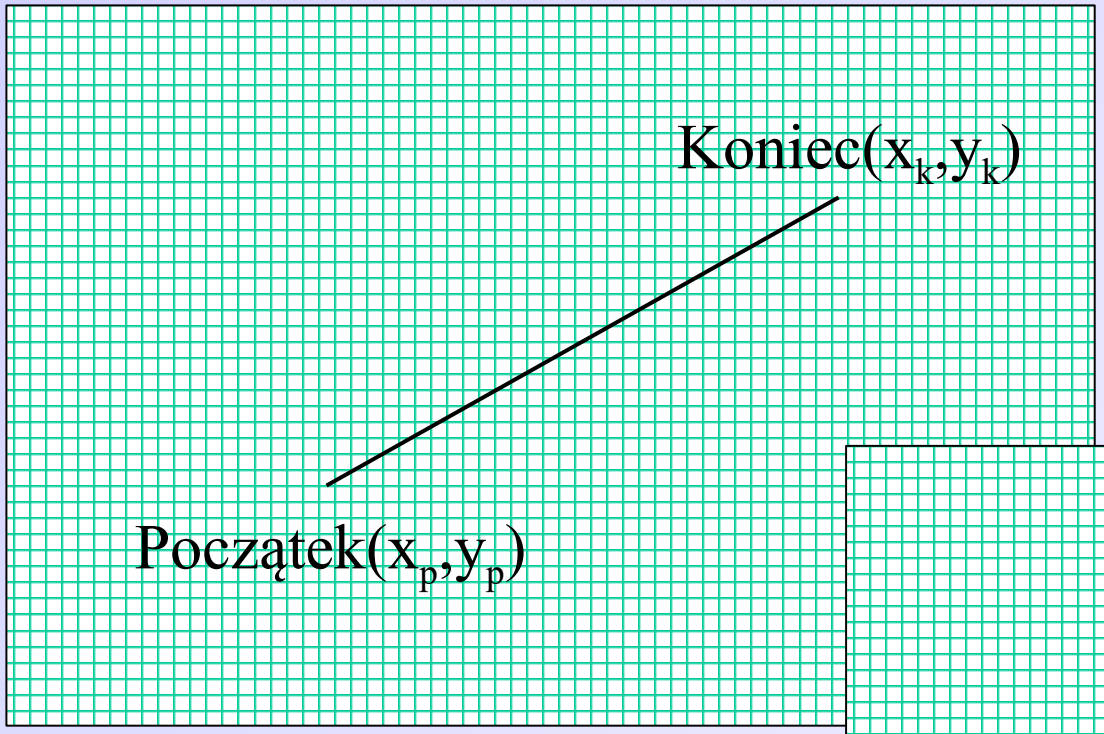
MSW Logo – przykłady grafiki żółwia.

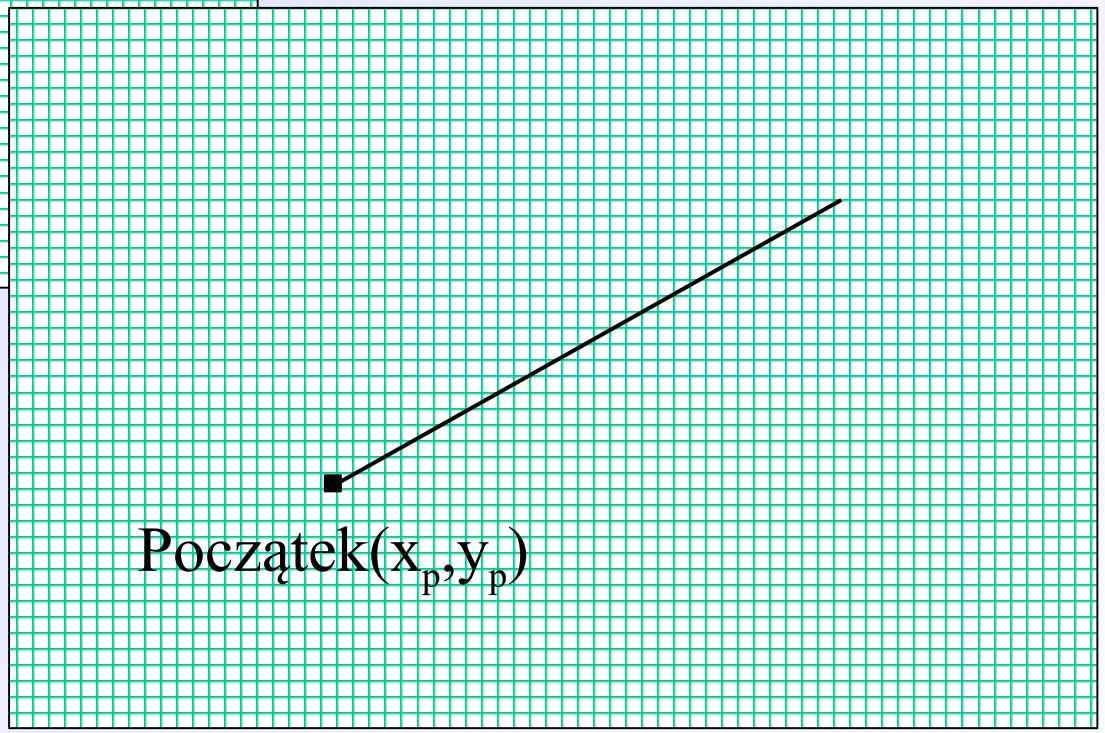
Grafika żółwia



MSW Logo – przykłady grafiki żółwia.

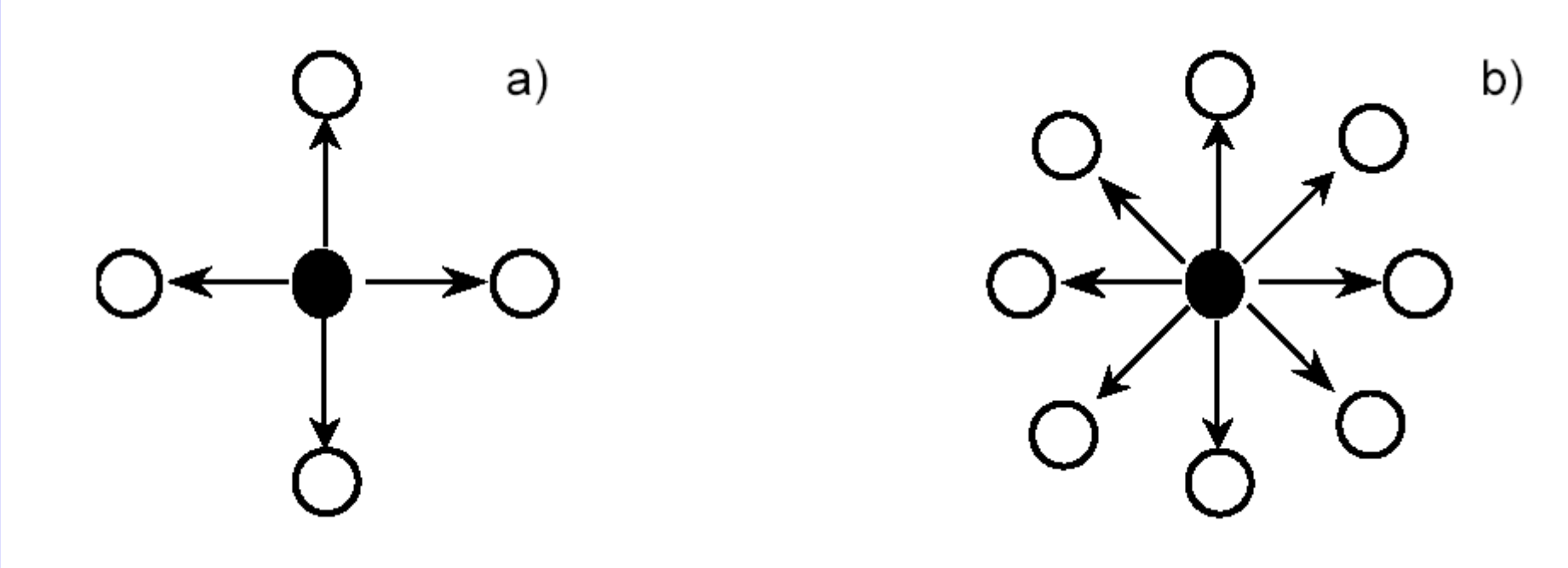
Grafika rastrowa





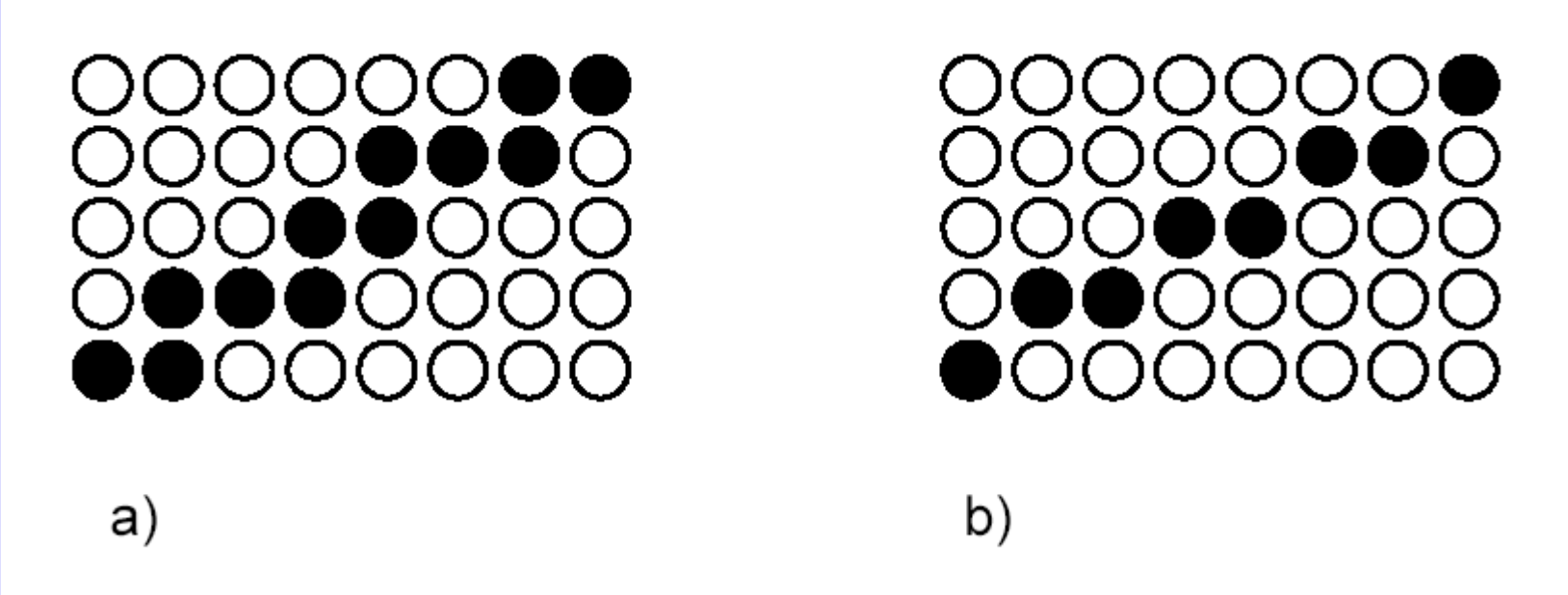
Grafika rastrowa
– rysowanie odcinka.

Grafika rastrowa



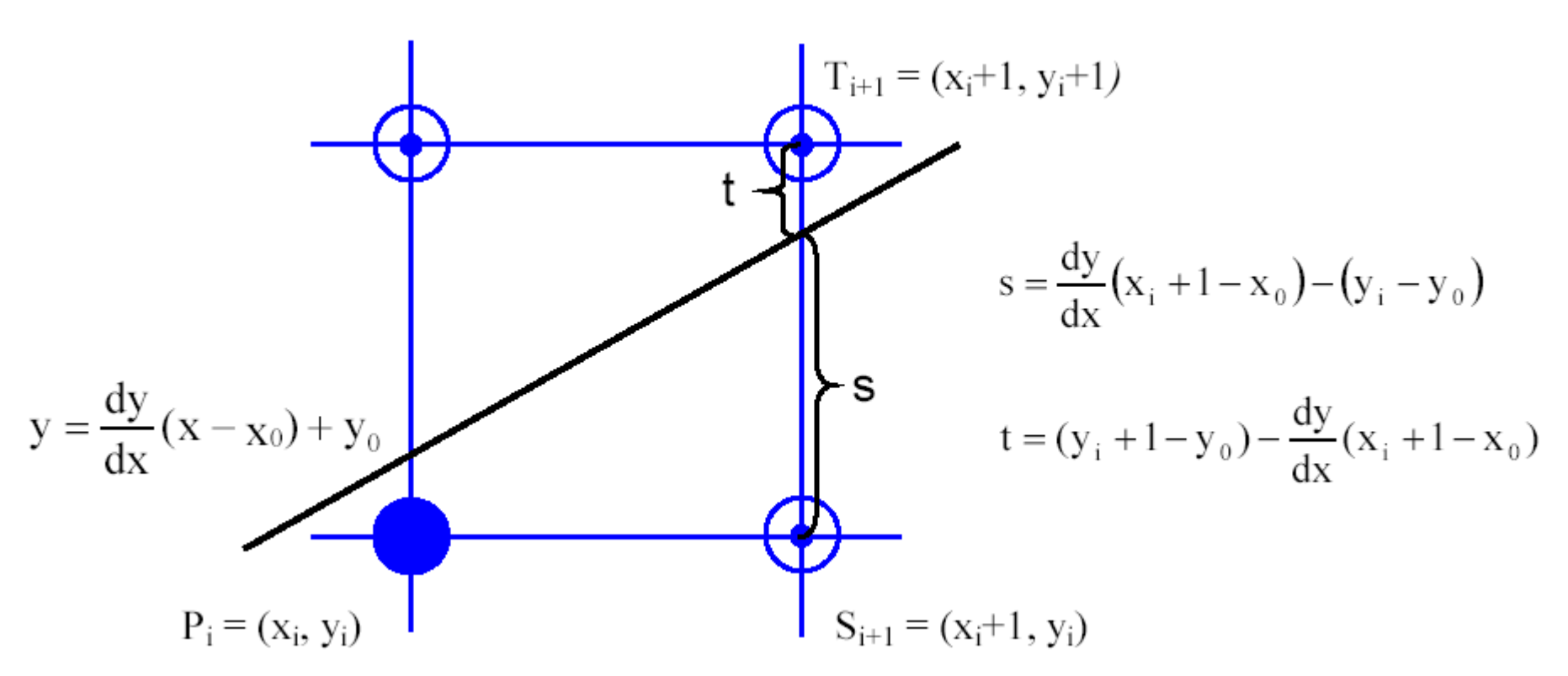
Rysowanie odcinka
a) wybór cztero-pikselowy, b) wybór ośmio-pikselowy.

Grafika rastrowa



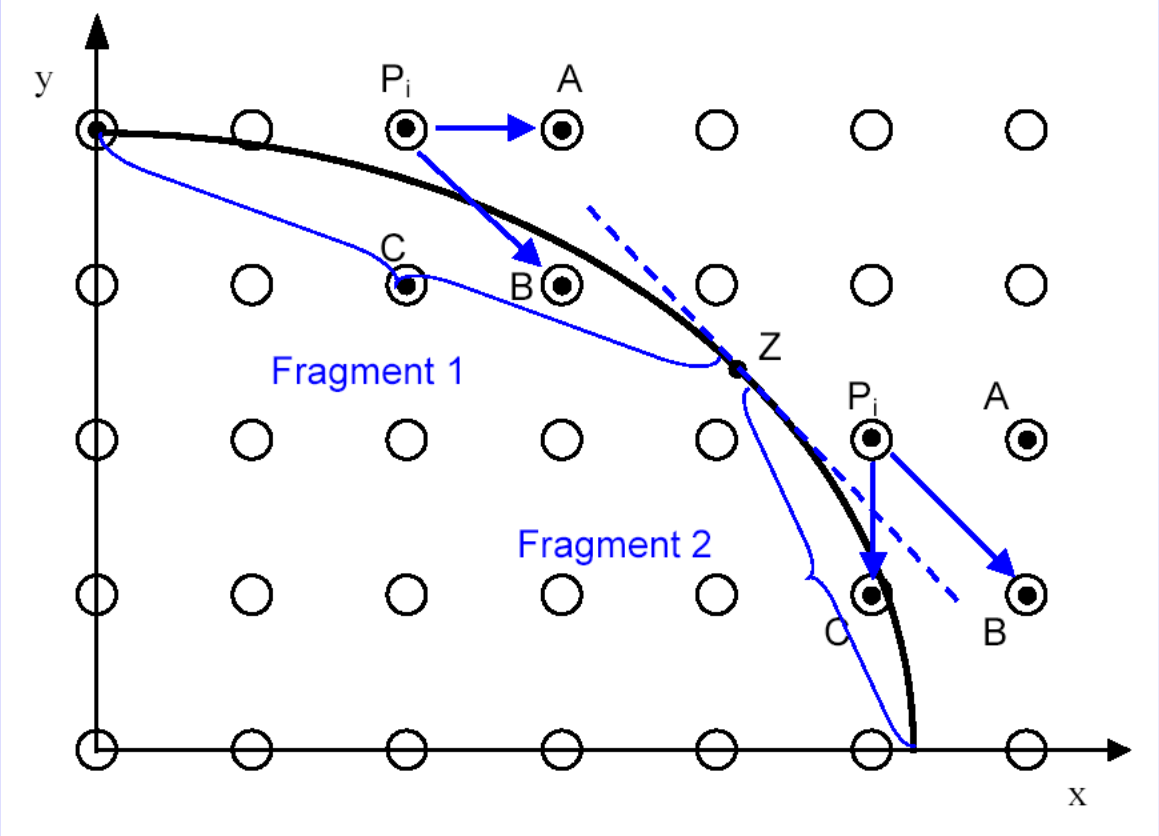
Rysowanie odcinka
a) z wykorzystaniem wyboru cztero-pikselowego,
b) z wykorzystaniem wyboru ośmio-pikselowego.

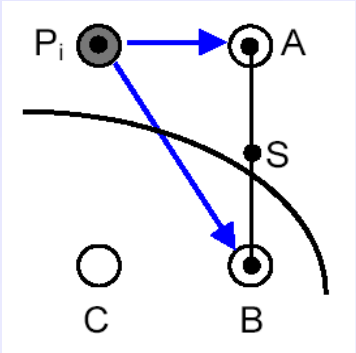
Grafika rastrowa

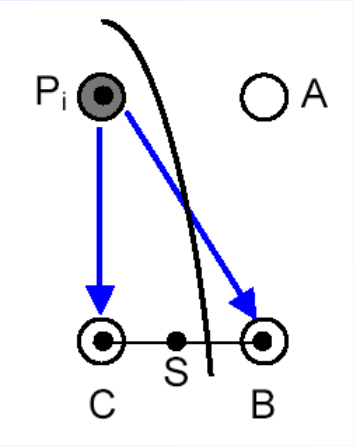


Algorytm Bresenhama.

Grafika rastrowa

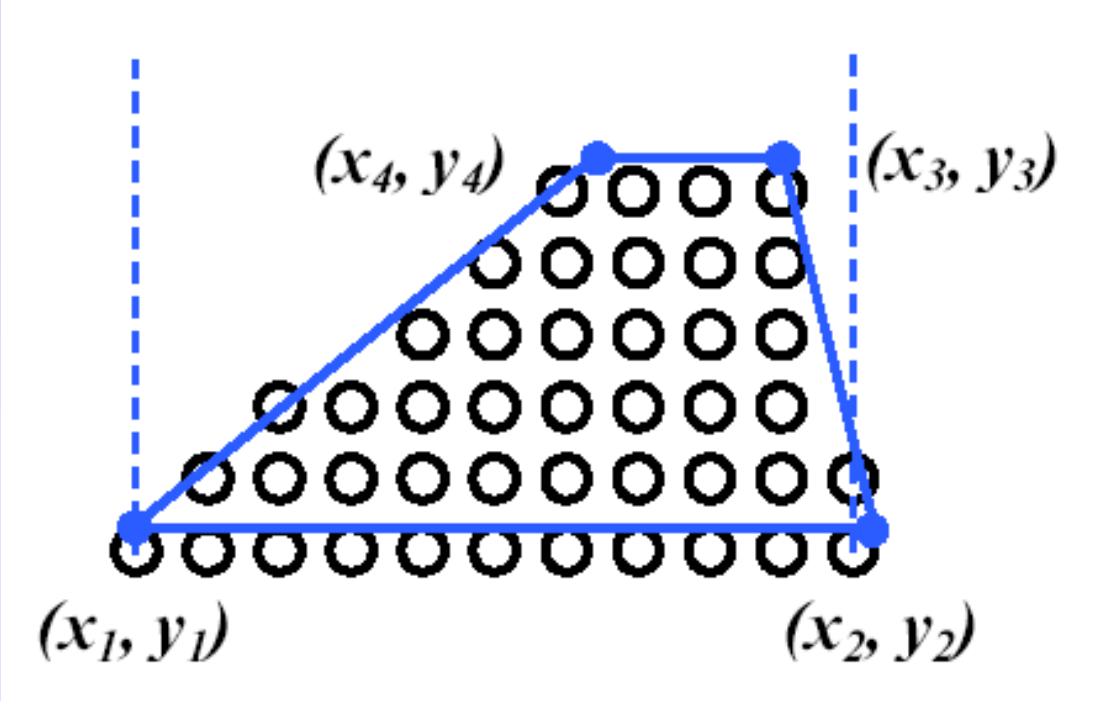






Rysowanie krzywych - algorytm von Akenema.

Grafika rastrowa



Wypełnienie trapezu.

UNIVERSITY OF WARMIA AND MAZURY IN OLSZTYN
The Faculty of Technical Sciences
POLAND, 10-957 Olsztyn, M. Oczapowskiego 11
tel.: (48)(89) 5-23-32-40, fax: (48)(89) 5-23-32-55
URL: http://www.uwm.edu.pl/edu/sobieski/ (in Polish)

Dziękuję za uwagę

Wojciech Sobieski

Olsztyn 2004-2011