Politechnika Krakowska
Wydział Inżynierii Elektrycznej i Komputerowej
Katedra Informatyki Technicznej

Instrukcja do ćwiczeń laboratoryjnych
z przedmiotu:

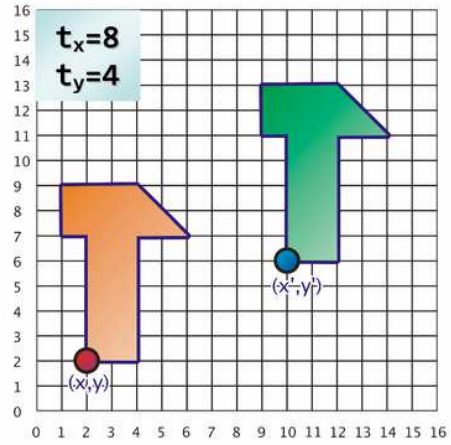
Grafika Komputerowa

Transformacje, wizualizacja i antyaliasing

Transformacje elementarne

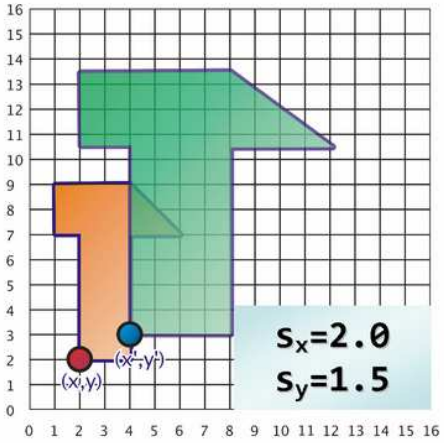
Przesunięcie (translation):

$$x' = x + t_x$$
$$y' = y + t_y$$



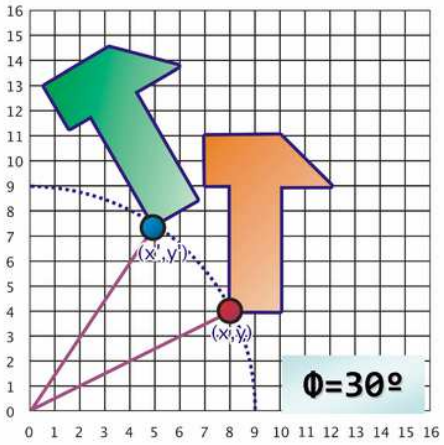
Zmiana skali (scaling):

$$x' = x \bullet s_x$$
$$y' = y \bullet s_y$$



Obrót wokół środka układu współrzędnych (rotation):

$$x' = x \cos\phi - y \sin\phi$$
$$y' = y \cos\phi + x \sin\phi$$



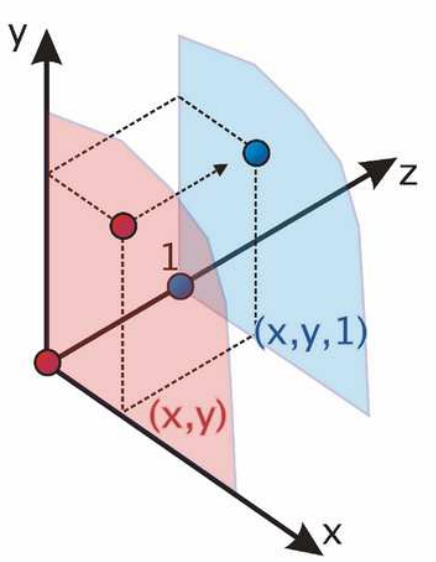
Transformacje elementarne

współrzędne przed transformacją:

$$(x, y) \rightarrow (x, y,1) \rightarrow [x \quad y \quad 1]$$

współrzędne po transformacji:

$$(x', y') \rightarrow (x', y',1) \rightarrow [x' \quad y' \quad 1]$$



Związek pomiędzy określonymi wcześniej wektorami można zapisać w postaci:

$$[x' \quad y' \quad 1] = [x \quad y \quad 1]\begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix}$$

Przesunięcie (translation):

równania:

$$x' = x + t_x$$
$$y' = y + t_y$$

zastępujemy równaniem:

$$[x' \quad y' \quad 1] = [x \quad y \quad 1]\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$$

transformatę skalowania opisuje macierz:

$$T(t_x, t_y) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$$

Zmiana skali (scaling):

równania:

$$x' = x \bullet s_x$$
$$y' = y \bullet s_y$$

zastępujemy równaniem:

$$[x' \quad y' \quad 1] = [x \quad y \quad 1]\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

transformatę przesunięcia opisuje macierz przesunięcia:

$$S(s_x, s_y) = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Obrót wokół środka układu współrzędnych (rotation):

równania:

$$x' = x\cos\phi - y\sin\phi$$
$$y' = y\cos\phi + x\sin\phi$$

zastępujemy równaniem:

$$[x' \quad y' \quad 1] = [x \quad y \quad 1]\begin{bmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

obrót opisuje macierz:

$$R(\phi) = \begin{bmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Wniosek:

- Po wprowadzeniu współrzędnych jednorodnych wszystkie trzy transformacje elementarne zostały opisane w ten sam sposób
- Współrzędne punktu po wykonaniu transformacji można wyznaczyć mnożąc wektor opisujący współrzędne punktu przed transformacją przez odpowiednią macierz

$$[x' \quad y' \quad 1] = [x \quad y \quad 1] \cdot M$$

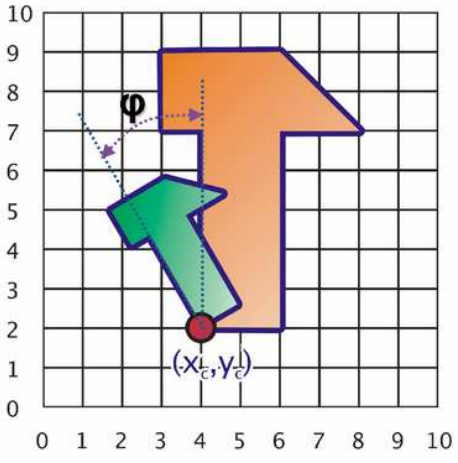
gdzie:

$$M = T, S \text{ lub } R$$

Składanie transformacji

Przykładowe zadanie:

- Obrócić obiekt wokół punktu (x_c, y_c) o kąt φ, pomniejszając go dwukrotnie
- Opisać określoną wyżej transformację za pomocą wzoru



Poszukiwana transformata zostanie wyznaczona w kilku krokach:

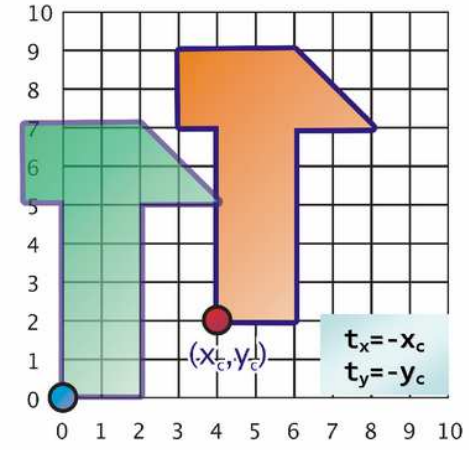
Krok 1

Przesunięcie obiektu tak, aby punkt (x_c, y_c) znalazł się w punkcie (0,0)

$$\mathbf{p} = [x \quad y \quad 1]$$

$$\mathbf{p}' = [x' \quad y' \quad 1]$$

$$\mathbf{p}' = \mathbf{p} \cdot T(-x_c, -y_c)$$



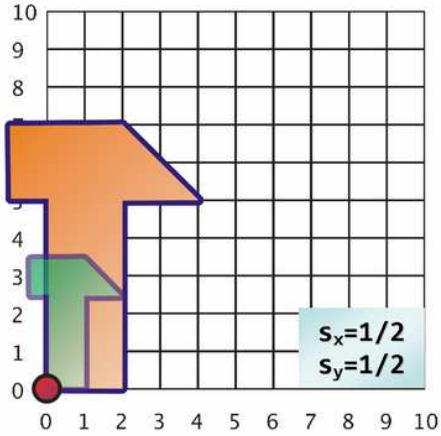
Krok 2

Przeskalować obiekt z parametrami skalowania $s_x=1/2$, $s_y=1/2$

$$\mathbf{p} = [x \quad y \quad 1]$$

$$\mathbf{p}' = [x' \quad y' \quad 1]$$

$$\mathbf{p}' = \mathbf{p} \cdot T(-x_c, -y_c) \cdot S(s_x, s_y)$$



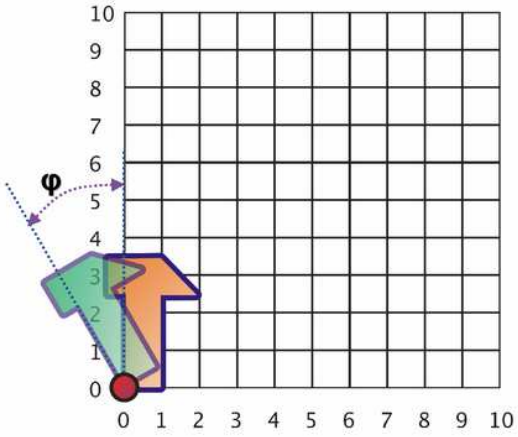
Krok 3

Obrócić obiekt wokół środka układu współrzędnych o kąt φ

$$\mathbf{p} = [x \quad y \quad 1]$$

$$\mathbf{p}' = [x' \quad y' \quad 1]$$

$$\mathbf{p}' = \mathbf{p} \cdot T(-x_c, -y_c) \cdot S(s_x, s_y) \cdot R(\varphi)$$



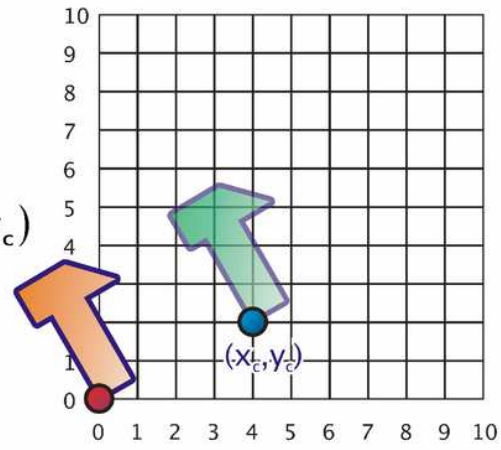
Krok 4

Przesunięcie obiektu tak, aby punkt (0,0) znalazł się w punkcie (x_c, y_c)

$$\mathbf{p} = [x \quad y \quad 1]$$

$$\mathbf{p}' = [x' \quad y' \quad 1]$$

$$\mathbf{p}' = \mathbf{p} \cdot T(-x_c, -y_c) \cdot S(s_x, s_y) \cdot R(\varphi) \cdot T(x_c, y_c)$$



Transformacja została znaleziona.

Jej macierz może zostać wyliczona po pomnożeniu czterech macierzy transformacji elementarnych.

$$\mathbf{p}' = \mathbf{p} \cdot T(-x_c, -y_c) \cdot S(s_x, s_y) \cdot R(\varphi) \cdot T(x_c, y_c)$$

Transformacja ogólniejsza:

$$\mathbf{p}' = \mathbf{p} \cdot T(-x_c, -y_c) \cdot S(s_x, s_y) \cdot R(\varphi) \cdot T(x_d, y_d)$$

Można wykazać, że

$$M = \begin{bmatrix} m_{11} & m_{12} & 0 \\ m_{21} & m_{22} & 0 \\ m_{31} & m_{32} & 1 \end{bmatrix}$$

Obliczanie nowych współrzędnych punktu:

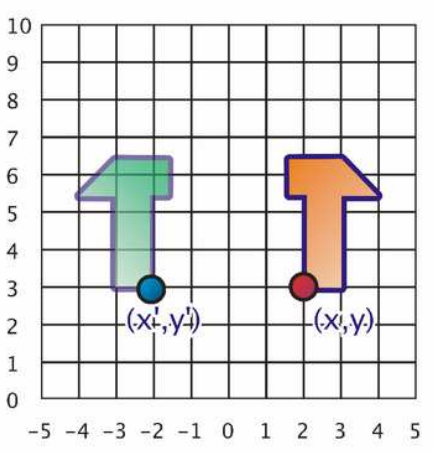
- 4 dodawania zmiennoprzecinkowe
- 4 mnożenia zmiennoprzecinkowe

Inne transformacje

Nie wszystkie transformacje mogą być wyrażone jako złożenia trzech, zdefiniowanych uprzednio transformacji elementarnych.

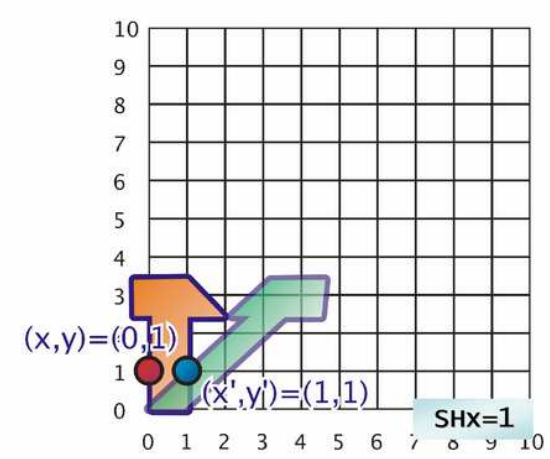
Odbicie (reflection):

$$[x' \quad y' \quad 1] = [x \quad y \quad 1]\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



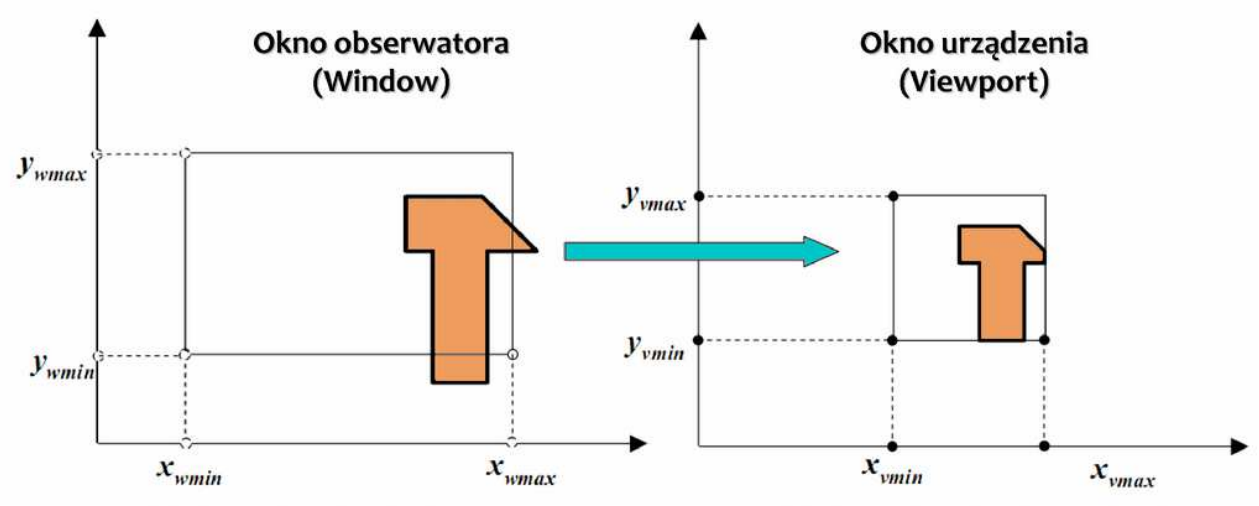
Ścinanie (shear):

$$[x' \quad y' \quad 1] = [x \quad y \quad 1]\begin{bmatrix} 1 & 0 & 0 \\ SHx & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



Wizualizacja

Ogólna procedura wizualizacji:



Algorytm wizualizacji:

1. Zdefiniować obiektu w układzie współrzędnych obserwatora.
2. W układzie współrzędnych obserwatora określić okno obserwatora.
3. W układzie współrzędnych urządzenia określić okno urządzenia.
4. Zmodyfikować opis obiektu usuwając te elementy, które znajdują się poza oknem obserwatora (obcinanie).
5. Przetransformować opis obiektu z wnętrza okna obserwatora do wnętrza okna urządzenia, stosując transformację

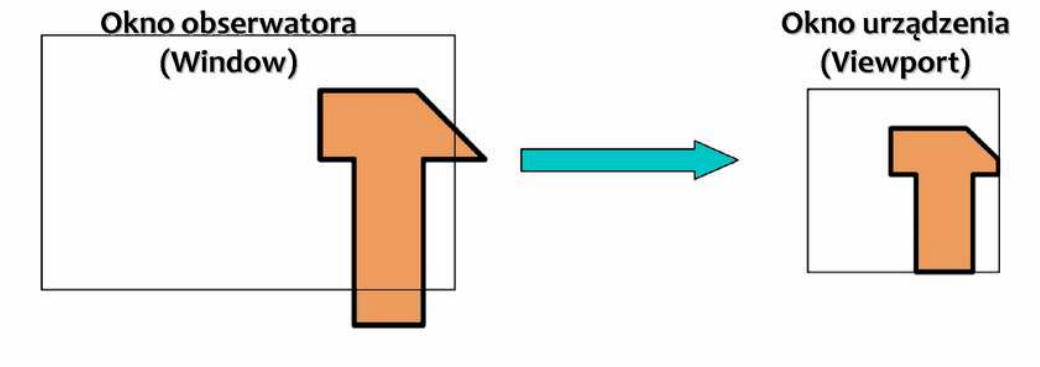
$$p_v = p_w \cdot T(-x_{w\,min}, -y_{w\,min}) \cdot S(s_x, s_y) \cdot T(x_{v\,min}, y_{v\,min})$$

przy czym

$$s_x = \frac{x_{v\,max} - x_{v\,min}}{x_{w\,max} - x_{w\,min}} \qquad s_y = \frac{y_{v\,max} - y_{v\,min}}{y_{w\,max} - y_{w\,min}}$$

6. Narysować obraz obiektu na ekranie.

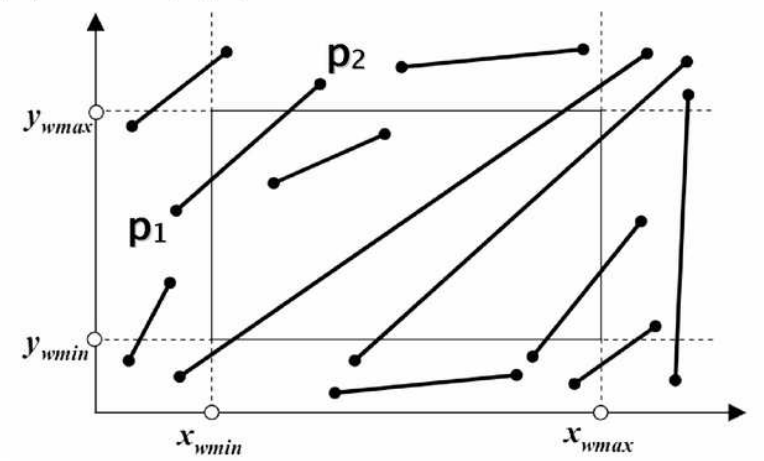
Obcinanie (clipping)



Obcinanie odcinka - algorytm Cohena i Sutherlanda

Założenie:

W przestrzeni obserwatora dany jest zbiór odcinków. Każdy odcinek opisany jest przez punkt początkowy (p1) i końcowy (p2).



Przykładowy układ odcinków w przestrzeni obserwatora

Kodowanie obszarów w przestrzeni obserwatora:

1001	1000	1010
0001	okno obserwatora 0000	0010
0101	0100	0110

bit4, bit3, bit2, bit1

bit1 = 1 – na lewo od okna obserwatora
bit2 = 1 – na prawo od okna obserwatora
bit3 = 1 – w dół od okna obserwatora
bit4 = 1 – w górę od okna obserwatora

Krok 1

Dla każdego punktu końcowego odcinka obliczyć różnice współrzędnych punktu końcowego i granic okna obserwatora

$$\alpha_1 = x - x_{w\,min} \qquad \alpha_2 = x_{w\,max} - x$$

$$\alpha_3 = y - y_{w\,min} \qquad \alpha_4 = y_{w\,max} - y$$

Krok 2

Zakodować wszystkie punkty końcowe odcinków według reguły:

jeżeli $\alpha_i > 0$ to $bit_i = 1$

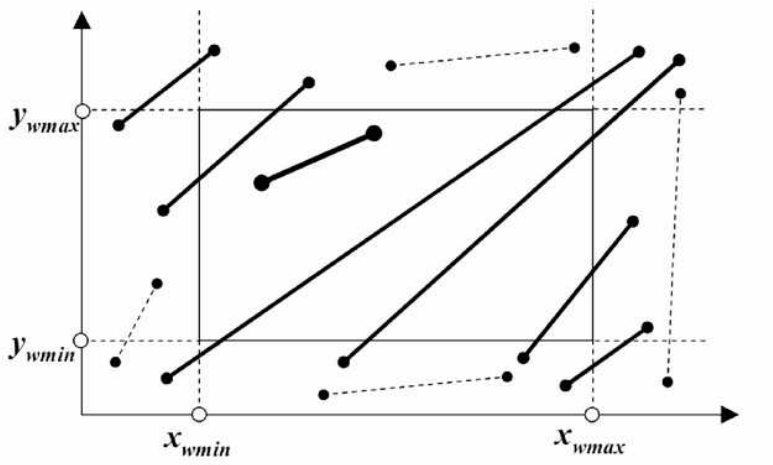
jeżeli $\alpha_i \leq 0$ to $bit_i = 0$

Krok 3

Sprawdzić kody par punktów końcowych dla wszystkich odcinków.

Jeżeli:

1. kod p_1 = kod p_2 = 0000 – odcinek leży całkowicie wewnątrz okna obserwatora.
2. bit_i dla p_1 = bit_i dla p_2 = 1 – odcinek leży całkowicie na zewnątrz okna obserwatora.
   - Pozostawić odcinki leżące wewnątrz okna,
   - Usunąć odcinki leżące na zewnątrz okna,
   - Jeżeli wyczerpano w ten sposób wszystkie odcinki zakończyć algorytm,
   - W przeciwnym przypadku wykonać Krok 4.



Efekt działania algorytmu po Kroku 3

Krok 4a

Dla pozostałych odcinków, których punkty końcowe leżą na lewo lub na prawo od granicy okna obliczyć nowe współrzędne tych punktów według wzorów:

$x = x_{wmin}$ dla punktów leżących na lewo od okna,

$x = x_{wmax}$ dla punktów leżących na prawo od okna,

$$y = y_k + m(x - x_k)$$

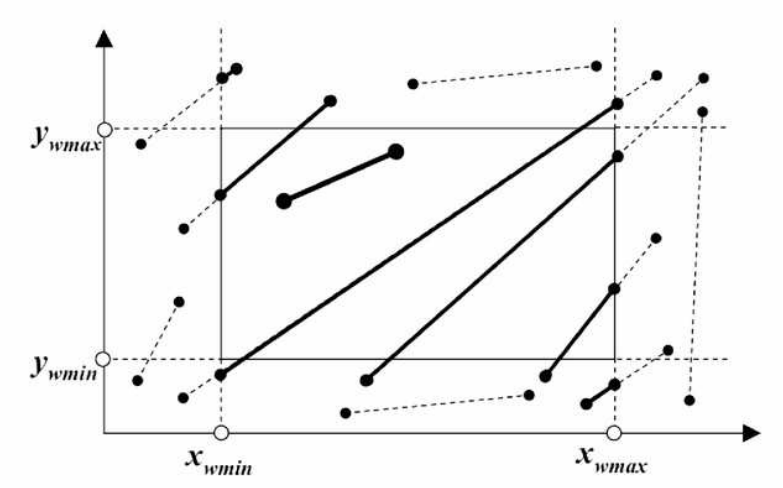
gdzie:

x, y – nowe współrzędne punktu końcowego,

x_k, y_k – poprzednie współrzędne punktu.

Zakodować nowe punkty końcowe według reguły opisanej w Kroku 2.

Powtórzyć Krok 3.



Efekt działania algorytmu po pierwszym przebiegu ponownego Kroku 3

Krok 4b

Dla pozostałych odcinków, których punkty końcowe leżą pod lub nad granicami okna obliczyć nowe współrzędne tych punktów według wzorów:

$y = y_{wmin}$ dla punktów leżących poniżej okna,

$y = y_{wmax}$ dla punktów leżących powyżej okna,

$$x = x_k + \frac{(y - y_k)}{m}$$

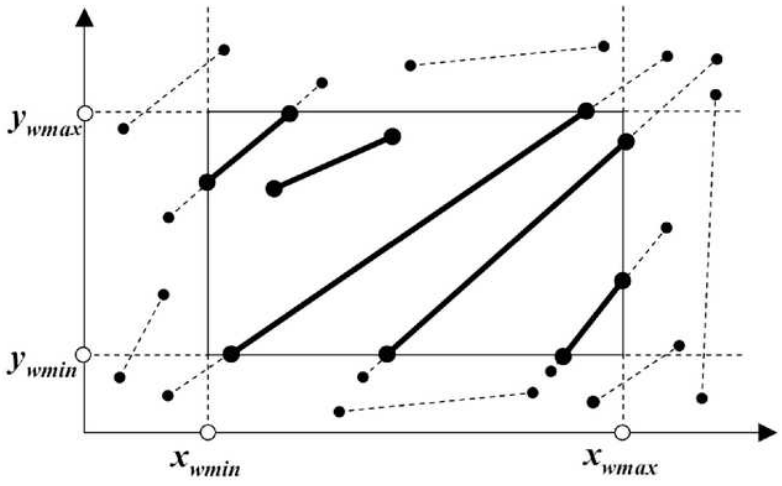
gdzie:

x, y – nowe współrzędne punktu końcowego,

x_k, y_k – poprzednie współrzędne punktu.

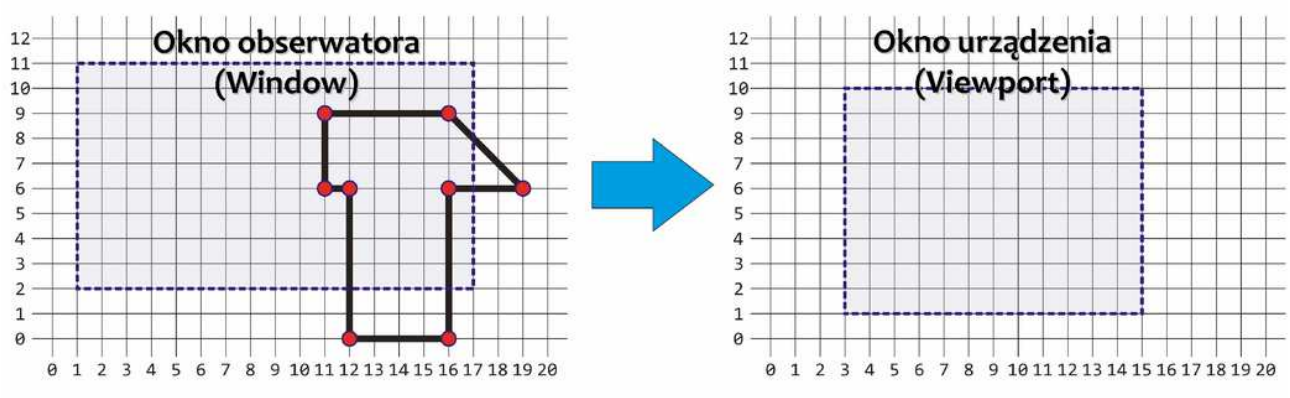
Zakodować nowe punkty końcowe według reguły opisanej w Kroku 2.

Powtórzyć Krok 3.



Końcowy efekt działania algorytmu obcinania

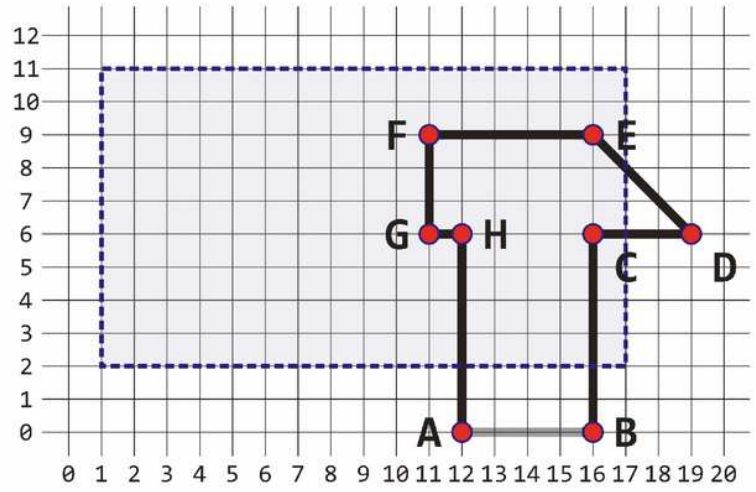
Przykład



$x_{wmin} = 1$
$x_{wmax} = 17$
$y_{wmin} = 2$
$y_{wmax} = 11$

$x_{vmin} = 3$
$x_{vmax} = 15$
$y_{vmin} = 1$
$y_{vmax} = 10$

$$s_x = \frac{x_{v\,max} - x_{v\,min}}{x_{w\,max} - x_{w\,min}} = 0{,}75 \qquad s_y = \frac{y_{v\,max} - y_{v\,min}}{y_{w\,max} - y_{w\,min}} = 1$$



A = 0100 A(12,0)
B = 0100 B(16,0)
C = 0000 C(16,6)
D = 0010 D(19,6)
E = 0000 E(16,9)
F = 0000 F(11,9)
G = 0000 G(11,6)
H = 0000 H(12,6)

$y = y_{wmin}$ dla punktów leżących poniżej okna,
$y = y_{wmax}$ dla punktów leżących powyżej okna,

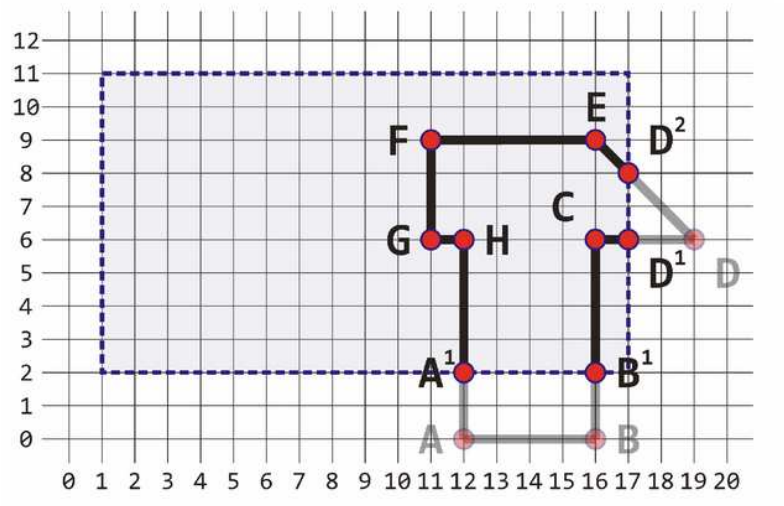
$$y = y_k + m(x - x_k)$$

gdzie:
x, y – nowe współrzędne punktu końcowego,
x_k, y_k – poprzednie współrzędne punktu.

$x = x_{wmin}$ dla punktów leżących po lewej stronie okna,
$x = x_{wmax}$ dla punktów leżących po prawej stronie okna,

$$x = x_k + \frac{(y - y_k)}{m}$$

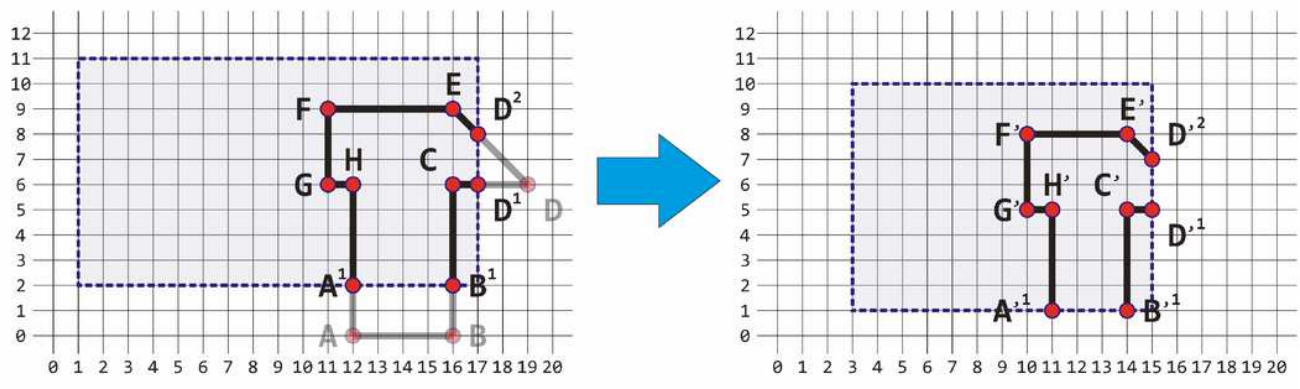
gdzie:
x, y – nowe współrzędne punktu końcowego,
x_k, y_k – poprzednie współrzędne punktu.



A(12,0)→A¹(12,2)
B(16,0)→B²(16,2)
C(16,6)
D(19,6)→D¹(17,6)
→D²(17,8)
E(16,9)
F(11,9)
G(11,6)
H(12,6)

$$p_v = p_w \cdot T(-x_{w\,min}, -y_{w\,min}) \cdot S(s_x, s_y) \cdot T(x_{v\,min}, y_{v\,min})$$

$$s_x = \frac{x_{v\,max} - x_{v\,min}}{x_{w\,max} - x_{w\,min}} = 0{,}75 \qquad s_y = \frac{y_{v\,max} - y_{v\,min}}{y_{w\,max} - y_{w\,min}} = 1$$



Antyaliasing

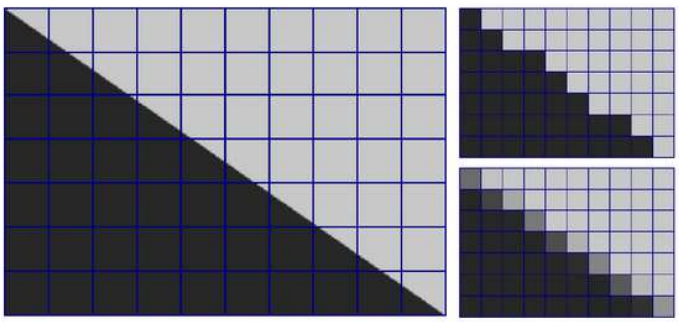
Aliasing to nieodwracalne zniekształcenie sygnału w procesie próbkowania wynikające z niespełnienia warunku Nyquista. Zniekształcenie to objawia się obecnością w sygnale składowych o błędnych częstotliwościach (aliasów).

Aliasing w grafice komputerowej to zjawisko zniekształcenia obrazu w wyniku zbyt małej częstości jego próbkowania w procesie rasteryzacji. Rasteryzacja ta zachodzi najczęściej podczas wyświetlania obrazu na ekranie, który obecnie najczęściej jest ekranem rastrowym, ale może dotyczyć także procesu zamiany modelu opisu obrazu z wektorowego na rastrowy.

Przykładem aliasingu w komputerowej grafice rastrowej jest występowanie „schodków" na liniach ukośnych lub obrzeżach brył w obrazach na monitorach rastrowych.

Aby zminimalizować ten efekt stosuje się tzw. antyaliasing, który powoduje jednak, że obraz staje się w pewnym stopniu nieostry.

Antyaliasing (ang. anti-aliasing) – zespół technik w informatyce służących zmniejszeniu, bądź całkowitej eliminacji aliasingu, czyli artefaktów powstających przy zmniejszaniu rozdzielczości (bądź częstotliwości) obrazu lub innego sygnału.



Wyświetlana bez użycia antyaliasingu biała linia na czarnym tle, będzie się składać z białych pikseli na czarnym tle i przy większości współcześnie używanych rozdzielczości ekranu, będzie w widoczny sposób nierówna – „schodkowana".

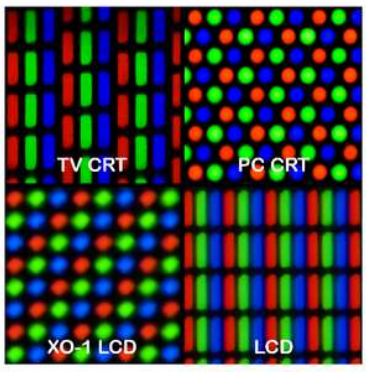
Antyaliasing w tym przypadku będzie polegał na wypełnieniu poszczególnych pikseli proporcjonalnie do odległości idealnej linii przechodzącej przez poszczególne piksele od środków tych pikseli - w takim rozwiązaniu piksel, przez którego środek przechodzi linia będzie biały, a piksel, w którym linia przechodzi w pobliżu jednego z wierzchołków – „prawie" czarny. Taka linia, będzie wydawać się dużo gładsza.

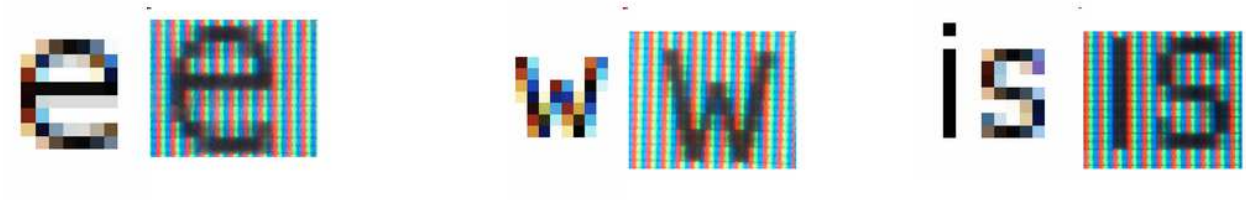
Innym rodzajem antyaliasingu jest **supersampling** (nadpróbkowanie). Jest to rozwiązanie polegające na użyciu metody czołgowej (brute force) do rozwiązania problemu aliasingu. W tej technice, obraz jest renderowany w rozdzielczości odpowiadającej wielokrotności rozdzielczości docelowej i uzyskany, dużo większy obraz jest uśredniany do tej niższej rozdzielczości. W używanych współcześnie układach graficznych technika ta nosi nazwę antyaliasingu pełnoekranowego (FSAA - ang. Full-Screen Anti-Aliasing) obsługiwanego przez wszystkie nowoczesne karty graficzne.

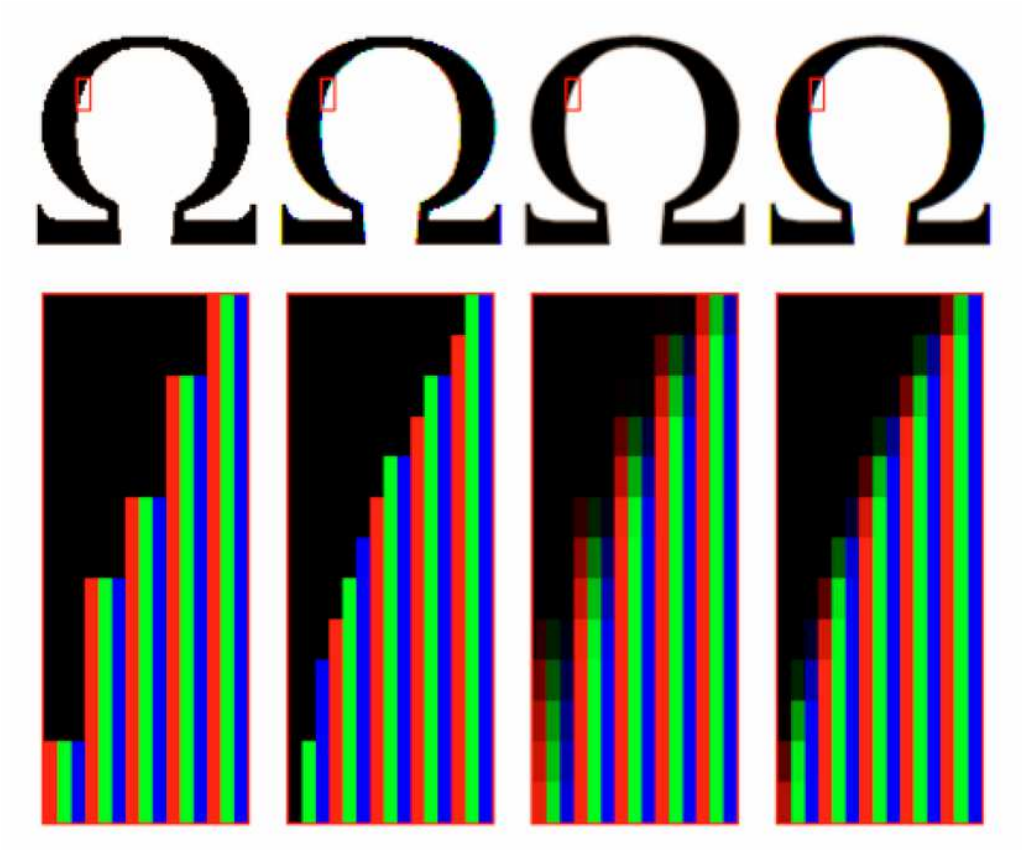
W większości środowisk graficznych antyaliasing jest stosowany do wygładzania krawędzi czcionek ekranowych. Jest to problem znacznie bardziej złożony, niż dla zwykłych linii, ponieważ rozmycia linii, które tworzą glify, mogą powodować nakładania się i zlewanie blisko leżących linii, co może znacząco utrudniać odczytanie tekstu.

Niemożliwe jest też często stosowanie zaawansowanych technik, ze względu na rosnący czas przetwarzania i dużą ilość tekstu. Między innymi z powodu trudności w zastosowaniu antyaliasingu dla fontów nie zaleca się stosowania pochylonego tekstu w rastrze.

Wygładzanie podpikselowe (ang. subpixel hinting) – technologia wygładzania czcionek polegająca na odpowiednim zapalaniu subpikseli. Powoduje ona lekkie rozmycie krawędzi liter, sprawiając, że tekst staje się gładszy.







Renderowanie, od lewej:

1. Bez antyaliasingu
2. Zwykłe renderowanie podpikselowe
3. Zwykły antyaliasing
4. Renderowanie podpikselowe z antyaliasingiem