

Jan Zabrodzki

GRAFIKA KOMPUTEROWA

Wykład 1: Informacje wstępne

Informacje wstępne

Grafika komputerowa jest obecnie obszerną dziedziną wiedzy, która obejmuje wiele zagadnień natury teoretycznej i algorytmicznej i która znajduje praktyczne zastosowania niemal wszędzie tam gdzie są stosowane komputery. Spośród najbardziej rozpowszechnionych zastosowań można wymienić następujące: realizacja interfejsów graficznych w komputerach, wizualizacja danych (poczynając od prostych wykresów słupkowych czy kołowych, a kończąc na zaawansowanych metodach wykorzystywanych w symulacjach naukowych), tworzenie wszelkiego rodzaju ilustracji dwuwymiarowych (2D) oraz scen trójwymiarowych (3D), tworzenie obrazów fotorealistycznych, wizualizacja w systemach wspomagania projektowania, zastosowania multimedialne, wizualizacja w systemach sztucznej rzeczywistości, sztuka komputerowa itd.

Podstawowym zadaniem grafiki komputerowej jest tworzenie obrazów, które są wyświetlane na ekranie komputera i mogą być reprodukowane innymi metodami, na przykład za pomocą drukarek bądź urządzeń poligraficznych. Współczesne metody wyświetlania obrazów wymagają by obraz przed wyświetleniem był przedstawiony w postaci matrycy pikseli. Oznacza to, że obrazy są generowane na bazie skończonej liczby punktów barwnych. Pociąga to za sobą konieczność korzystania z odpowiednich algorytmów, które pozwalają tworzyć obrazy przy takim ograniczeniu.

Tworzone obrazy w większości przypadków powinny być podobne do obrazów, które oglądamy w życiu codziennym. Tworzone obrazy powinny być zgodne z naszą intuicją. Powinny one prawidłowo odzwierciedlać obiekty widoczne dla obserwatora przy uwzględnieniu warunków oświetlenia.

Grafika komputerowa jest silnie powiązana z przetwarzaniem obrazów. O ile w grafice chodzi o tworzenie nowych obrazów, to w przetwarzaniu obrazów istniejący obraz jest poddawany odpowiednim operacjom, które pozwalają uzyskać inną postać tego obrazu. Metody przetwarzania obrazów często wspomagają metody grafiki komputerowej, ułatwiając uzyskanie pożądanego obrazu końcowego.

Poszczególne wykłady składające się na niniejszy kurs pozwalają poznać kolejno najistotniejsze zagadnienia związane z grafiką komputerową. Na początku przedstawiono podstawowe problemy dla techniki rastrowej. Omówiono metody cyfrowej reprezentacji obrazu (wykład II) oraz metody wyświetlania obrazów na monitorach CRT oraz LCD (wykład III). Następne dwa wykłady (IV i V) poświęcono problemowi barwy w grafice komputerowej. Wyjaśniono podstawowe pojęcia oraz problemy związane z percepcją barwy. Omówiono również najważniejsze modele barw stosowane w grafice komputerowej. Te pierwsze cztery wykłady stanowią

bazę, której poznanie i zrozumienie jest konieczne dla każdego, kto chce zajmować się grafiką komputerową.

Kolejne trzy wykłady poświęcono grafice 2D, a więc metodom tworzenia obrazów dwuwymiarowych. Wykład VI prezentuje podstawowe algorytmy rysowania odcinków i okręgów oraz do wypełniania figur itd. na bazie rastra. W wykładzie VII przedstawiono wybrane metody istotne z punktu widzenia grafiki wektorowej, w tym przekształcenia geometryczne oraz metody obcinania. Omówiono również krzywe Béziera. Z kolei wykład VIII pozwala zapoznać się z wybranymi narzędziami jakie spotyka się w programach graficznych bitmapowych i wektorowych.

Wykłady od IX do XII poświęcono grafice 3D. Pierwszy z tych wykładów omawia metody modelowania obiektów 3D oraz scen zawierających te obiekty, natomiast pozostałe trzy wykłady poświęcono metodom renderingu, a więc metodom, które umożliwiają wyświetlenie opracowanych modeli obiektów lub scen na ekranie, z uwzględnieniem efektów związanych z oświetleniem. W wykładach X i XI omówiono zagadnienia rzutowania, eliminowania elementów niewidocznych oraz cieniowania. Natomiast wykład XII poświęcono metodzie śledzenia promieni. Metoda ta pozwala tworzyć obrazy o bardzo dobrej jakości.

Wszystkie wcześniej omawiane metody umożliwiały tworzenie pojedynczych obrazów. Wykład XIII prezentuje zagadnienia związane z tworzeniem sekwencji obrazów. Omówiono typowe problemy spotykane w animacji komputerowej.

Wykład XIV prezentuje podstawowe metody z zakresu przetwarzaniu obrazów.

Utworzone obrazy przechowywane są w plikach o specjalnych formatach umożliwiających najczęściej stosowanie kompresji. Wybrane formaty plików oraz metody kompresji obrazów prezentuje wykład XV.

Ostatni wykład XVI zawiera podsumowanie kursu. Zamieszczono tam również krótki przegląd bogatego oprogramowania graficznego. Podano także przykładowe tematy egzaminacyjne.

Po każdym wykładzie podano kilka pytań i problemów do samodzielnego rozwiązania. Zachęcam do rozwiązywania tych problemów.

Materiał podzielono na wykłady ze względu na poruszaną tematykę. Stąd faktyczny czas potrzebny na zapoznanie się z materiałem zawartym w poszczególnych wykładach może być różny.

W miarę poznawania zagadnień omawianych w poszczególnych wykładach wskazane jest, korzystanie z prostych programów graficznych powszechnie dostępnych. Na przykład można korzystać z programu Paint dostępnego w systemie Windows oraz z możliwości tworzenia rysunków dostępnych w programie Word czy też PowerPoint. Oczywiście można również korzystać z dostępnych w sieci darmowych programów graficznych.

Zainteresowanych pogłębieniem wiadomości zawartych w niniejszym kursie odsyłam do podanej literatury pomocniczej. Istnieje również możliwość korzystania

z konsultacji za pomocą poczty elektronicznej (za wyjątkiem okresów urlopowych) pod adresem jza@ii.pw.edu.pl. Na ten adres proszę przysyłać również wszelkie uwagi dotyczące prezentowanego materiału.

Przedstawiony materiał teoretyczny wyjaśnia podstawowe pojęcia i metody wykorzystywane w grafice komputerowej. Ilustracją do tego będą z pewnością zajęcia laboratoryjne, w ramach których można będzie praktycznie poznać trzy profesjonalne programy graficzne.

Wykład 2: Technika rastrowa

Obraz, podobnie jak każda inna informacja w komputerze, musi być przedstawiony w postaci cyfrowej. W wykładzie wyjaśniono sposób reprezentacji obrazu w postaci cyfrowej oraz omówiono problemy związane ze stosowaniem takiej reprezentacji.

1. Cyfrowa reprezentacja obrazu

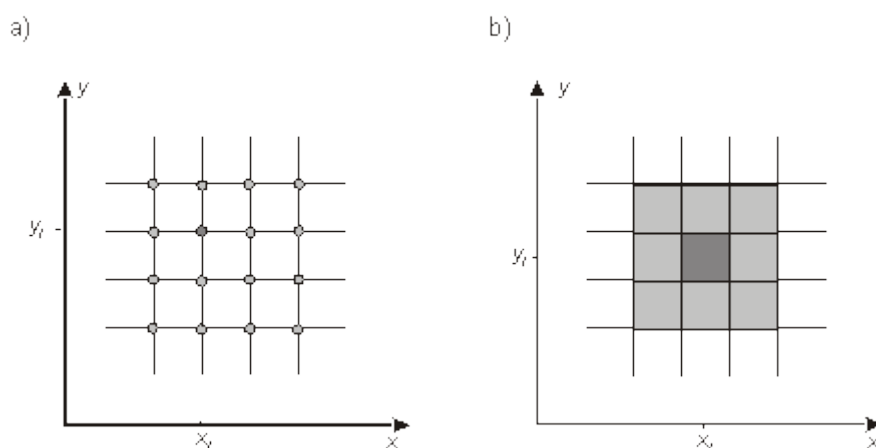
W otaczającym nas świecie obserwujemy obrazy ciągłe składające się z nieskończenie wielu punktów, przy czym każdy punkt może być reprezentowany za pomocą nieskończenie wielu wartości. Każda próba reprezentowania takich obrazów oznacza konieczność ograniczenia zarówno liczby modelowanych punktów jak i liczby wartości jakie można przypisać poszczególnym punktom. W szczególności z sytuacją taką mamy do czynienia w systemach cyfrowych, w których każdy parametr może przyjmować wartości jedynie ze skończonego zbioru wartości. W konsekwencji, w odniesieniu do obrazów powstają następujące problemy: jak rzeczywisty obraz reprezentować w postaci cyfrowej oraz jak tworzyć obraz metodami cyfrowymi żeby możliwie dobrze odzwierciedlał obraz rzeczywisty. W obu przypadkach konieczne jest korzystanie z odpowiedniej reprezentacji cyfrowej obrazu.

Konwersja obrazów rzeczywistych na postać cyfrową polega na próbkowaniu obrazu i kwantowaniu próbek. Natomiast konwersja odwrotna następuje praktycznie dopiero w czasie obserwacji przez człowieka obrazu wyświetlanego na przykład na monitorze - wykorzystuje się tu właściwości systemu wzrokowego człowieka pozwalające odbierać informację dyskretną jako informację ciągłą (oczywiście przy dostatecznie dobrej wizualizacji obrazów).

Proces próbkowania obrazu można wyjaśnić poglądowo w następujący sposób. Wyobraźmy sobie, że na zdjęcie czarno-białe (przyjmujemy tu upraszczające założenie, że zdjęcie dobrej jakości odzwierciedla obraz rzeczywisty) nanosimy gęstą siatkę prostokątną. Każdy węzeł siatki odpowiada jednemu punktowi obrazu - jest próbka obrazu. Zbiór tak określonych próbek stanowi pewne przybliżenie obrazu. Każda próbka obrazu może przyjmować dowolną wartość z określonego przedziału jasności (odcieni szarości). Z kolei ten przedział wartości jest dzielony na pewną liczbę podprzedziałów i poszczególnym podprzedziałom przypisuje się

różne kody cyfrowe. Stąd każdy kod cyfrowy reprezentuje określoną jasność (odcień szarości).

Obok opisanej wyżej reprezentacji obrazu, w której piksele są przedstawiane jako węzły siatki, w praktyce spotykana jest również inna reprezentacja obrazu, w której poszczególne piksele są traktowane jako obszary o określonej powierzchni i są reprezentowane przez oczka siatki prostokątnej. Gdyby odwoływać się do procesu próbkowania obrazu ciągłego, to można przyjąć, że każdemu pikselowi jest teraz przypisana uśredniona barwa w obszarze oczka siatki prostokątnej. Na rysunku II.1 pokazane są obie reprezentacje. Formalnie można zwrócić uwagę na przesunięcie współrzędnych poszczególnych pikseli o $\frac{1}{2}$ skoku siatki.



Rys. II.1 Dwie równoważne reprezentacje pikseli: a) punktowa, b) powierzchniowa

W efekcie nasze zdjęcie zostało zastąpione zbiorem próbek reprezentowanych w postaci cyfrowej. Jest to uporządkowany zbiór próbek o określonych jasnościach. W podobny sposób można przedstawić również zdjęcie kolorowe. Poszczególne próbki są określane w grafice komputerowej jako piksele. (Zauważmy, że operację próbkowania realizują na przykład skanery oraz aparaty cyfrowe.)

Jeżeli z naszą hipotetyczną siatką prostokątną zwiążemy układ współrzędnych prostokątnych x, y , to położenie każdego piksela p można opisać podając odpowiednie współrzędne x_p, y_p . Ostatecznie, obraz jest reprezentowany przez prostokątną macierz $m \times n$ pikseli, gdzie m i n reprezentują liczby pikseli, odpowiednio w poziomie i w pionie. Liczby te określają rozdzielczość pikselową obrazu, odpowiednio poziomą i pionową wyrażoną w pikselach. Tak określona rozdzielczość obrazu jest niezależna od rzeczywistych rozmiarów obrazu - precyzuje jedynie za pomocą ilu pikseli obraz jest reprezentowany.

Zależnie od sposobu reprodukcji obrazu jego rozmiary będą różne. Rozmiary te można określić jeżeli znana jest rozdzielczość liniowa urządzenia reprodukującego, określona przez liczbę podstawowych elementów (linii, pikseli, kropek, próbek) przypadających na określoną jednostkę długości. Na przykład, w odniesieniu do drukarek czy skanerów używa się miary wyrażanej w jednostkach dpi (punktów na cal), a w odniesieniu do druku lpi (linii na cal).

Przykład

Jakie będą rozmiary obrazu o rozdzielczości pikselowej 800 \times 600 reprodukowanego na drukarce o rozdzielczości a) 300 dpi, b) 600 dpi?

W przypadku a) wymiary obrazu będą wynosiły odpowiednio:

$$800/300 = 2,66'' = 6,77 \text{ cm} \text{ i } 600/300 = 2'' = 4,08 \text{ cm}.$$

W przypadku b) otrzymamy odpowiednio:

$$800/600 = 1,33'' = 3,39 \text{ cm} \text{ i } 600/600 = 1'' = 2,54 \text{ cm}.$$

W przypadku urządzeń wyświetlających najczęściej korzysta się z rozdzielczości pikselowej dla charakteryzowania jakości wyświetlanego obrazu. W przypadku gdy znane są rozmiary ekranu monitora możliwe jest również określenie rozdzielczości liniowej w jednostkach takich jak na przykład dpi.

Przykład

Dla ekranu o rozdzielczości 800 \times 600 pikseli i przekątnej 15" określić rozdzielczość liniową w poziomie wyrażoną w dpi. Przyjąć, że stosunek boków ekranu wynosi a : b = 4 : 3.

Z równania Pitagorasa mamy, że $15^2 = a^2 + b^2$. Wprowadzając pomocniczą zmienną x możemy boki kwadratu zapisać w postaci a = 4x oraz b = 3x. Po podstawieniu do równania i rozwiązaniu otrzymujemy, że x = 3". Stąd a = 12" i b = 9". Wobec tego szukana rozdzielczość w poziomie wynosi $800/12'' = 66,6 \text{ dpi}$.

Obraz przedstawiony w postaci matrycy pikseli jest określany często jako mapa pikseli; w praktyce stosowane są również określenia mapa bitowa oraz bitmapa niezależnie od tego ile bitów jest wykorzystywanych do reprezentacji piksela.

Dla zapamiętania obrazu w postaci mapy pikseli potrzebna jest odpowiednia ilość miejsca w pamięci. Dla przykładu, jeżeli obraz ma rozdzielczość pikselową 800 \times 600 i barwa pojedynczego piksela jest reprezentowana za pomocą 24 bitów, to do zapamiętania informacji o obrazie potrzeba $800 \times 600 \times 24$ bitów w pamięci (11,52 Mb lub 1,44 MB).

Przykład

Za pomocą dwóch urządzeń uzyskano obrazy o rozmiarach 2" \times 2". Określić wymaganą pojemność pamięci dla przechowania obrazów jeżeli poszczególne urządzenia miały następujące parametry: a) rozdzielczość liniowa 72 dpi, obraz czarno-biały 1 bit/piksel, b) rozdzielczość liniowa 300 dpi, obraz kolorowy 24 bity/piksel. Zakładamy, że przechowujemy tylko informacje o pikselach obrazu.

W przypadku a) potrzebna jest pamięć o pojemności:

$$(2 \times 72) \times (2 \times 72) \times 1 = 2,592 \text{ kB}$$

W przypadku b) potrzebna jest pamięć o pojemności:

$$(2 \times 300) \times (2 \times 300) \times 24 = 1,08 \text{ MB}$$

Rozdzielczość pikselowa obrazu nie jest cechą stałą obrazu. Obraz może być przedstawiony z różnymi rozdzielczościami zależnie od potrzeb bądź od możliwości urządzeń próbkujących, pamiętających czy wyświetlających. Oczywiście im większa rozdzielczość obrazu tym lepsza aproksymacja rzeczywistego obrazu. Na rysunku II.2 pokazany jest ten sam obraz przy dwóch różnych rozdzielczościach pikselowych.



Rys. II.2. Obraz przedstawiony z różnymi rozdzielczościami pikselowymi

Podobnie jak w przypadku rozdzielczości pikselowej obrazu również liczba bitów poświęcanych na reprezentację barwy pojedynczego piksela nie jest cechą charakterystyczną obrazu. Przy większej liczbie bitów uzyskujemy wierniejsze odzwierciedlenie rzeczywistej barwy. Na rysunku II.3 pokazano ten sam obraz wyświetlony przy dwóch różnych liczbach bitów reprezentujących barwę pojedynczego piksela; w obu przypadkach rozdzielczość pikselowa obrazu jest stała.



Rys. II.3. Obraz przy różnych liczbach bitów na piksel

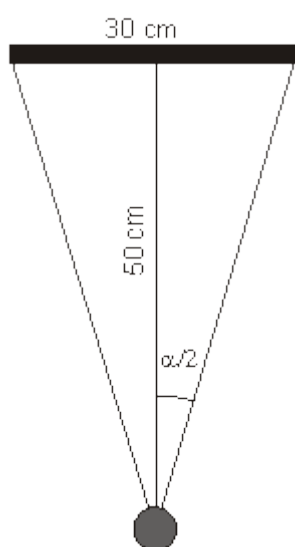
Jednak nadmierne zwiększanie liczby bitów reprezentujących barwę piksela ponownie napotyka barierę sensowności, wynikającą z ograniczonych możliwości postrzegania barw przez człowieka. Na ogół uważa się, że dla potrzeb fotorealistycznych obrazów wystarcza 8 bitów dla zakodowania informacji o luminancji albo o każdej z podstawowych barw. Czasami może występować potrzeba zwiększenia tej wartości do 10 lub 12 bitów.

W praktyce komputerowej są stosowane różne rozdzielczości pikselowe obrazu. Najczęściej stosunek rozdzielczości w poziomie do rozdzielczości w pionie wynosi 4 : 3, czyli tyle ile wynosi typowy stosunek boków w monitorach. Ostatnio najczęściej spotykane wartości rozdzielczości to 640 \times 480, 800 \times 600, 1024 \times

768, 1600 \times 1200. Niemniej spotykane są rozdzielczości o innym stosunku, na przykład 5 : 4 (1280 \times 1024).

W miarę upływu czasu i rozwoju technologii rośnie "typowa" rozdzielczość pikselowa obrazów wyświetlanych na ekranach monitorów - kiedyś było to 320 \times 240; obecnie jest to 1024 \times 768. Można postawić pytanie czy są jakieś granice w tym względzie. Wydaje się, że jedyne sensowne ograniczenie można wiązać ze skończoną zdolnością systemu wzrokowego człowieka do rozróżniania szczegółów. Praktyczna reguła mówi, że człowiek jest w stanie rozróżnić szczegóły widziane w obrębie kąta o wartości około 1 minuty kątowej. Przyjmując, że typowa odległość obserwatora od ekranu monitora wynosi około 0,5 metra otrzymuje się, że maksymalna sensowna rozdzielczość odnośnie do postrzegania drobnych szczegółów jest w przedziale 100-200 punktów (pikseli) na centymetr. (We współczesnych monitorach mamy dostępne rozdzielczości rzędu 30-50 punktów na centymetr.). Zwiększanie rozdzielczości poza ten przedział będzie się miało z celem skoro i tak nie będziemy w stanie rozróżnić drobnych szczegółów. Warto zwrócić uwagę, że praktycznie ta granica została już osiągnięta. Są dostępne monitory, które wyświetlają obrazy o rozdzielczości pikselowej 4096 \times 3072.

Przykład



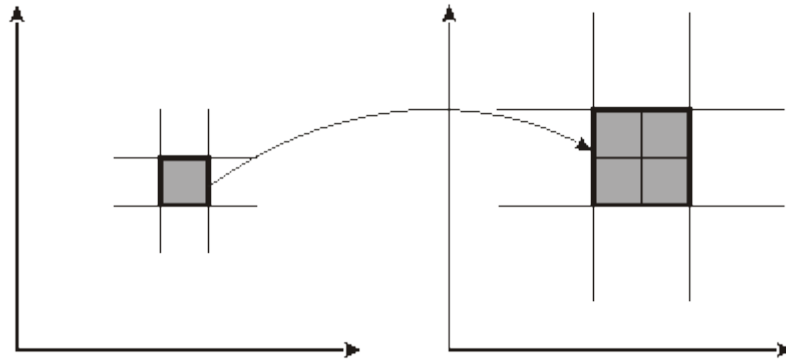
Obserwator ogląda obraz o szerokości 30 cm z odległości 50 cm. Przyjmując, że obserwator jest w stanie rozróżnić szczegóły widziane w obrębie kąta o wartości 1 minuty kątowej (1') określić przybliżoną liczbę pikseli w poziomie, które obserwator jest w stanie rozróżnić.

Korzystając z rysunku pokazanego obok możemy zapisać, że $\tan(\alpha/2) = 0,3$. Stąd $\alpha/2 \approx 17^\circ$ i $\alpha \approx 34^\circ$. Przy założonej zdolności rozróżniania szczegółów i pamiętając, że $1' = 60''$ otrzymujemy szukany wynik: $34 \times 60 = 2040$ pikseli.

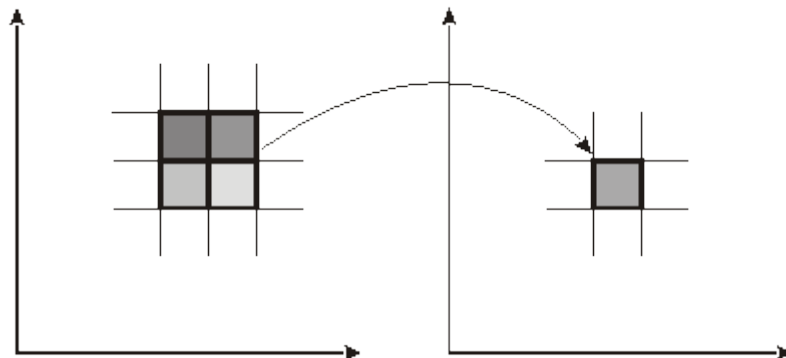
2. Zmiana rozdzielczości obrazu

Czasami powstaje konieczność zmiany rozdzielczości obrazu. Teoretycznie rozdzielczość można zmieniać na zasadzie ponownego próbkowania pierwotnego obrazu. W praktyce jednak na ogół dysponujemy obrazem w postaci cyfrowej o pewnej rozdzielczości pikselowej i na tej podstawie musimy określić nową reprezentację cyfrową obrazu.

Rozwiązanie problemu jest proste jeżeli ograniczymy zmianę rozdzielczości pikselowej jedynie do kilku korzystnych wartości. Na rysunkach II.4 i II.5 pokazano dwa przykłady takich korzystnych sytuacji.



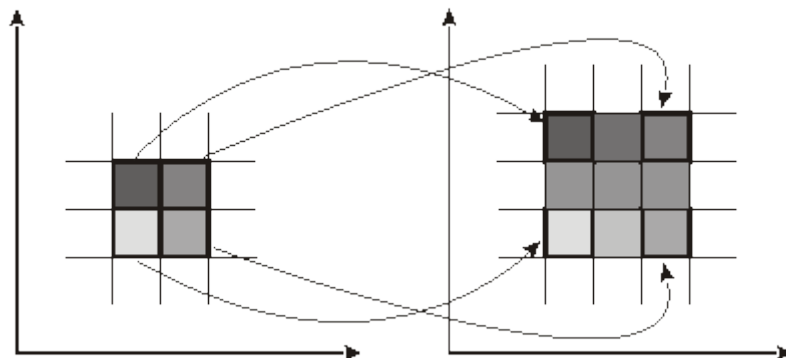
Rys. II.4. Dwukrotne zwiększanie rozdzielczości (w poziomie i w pionie)



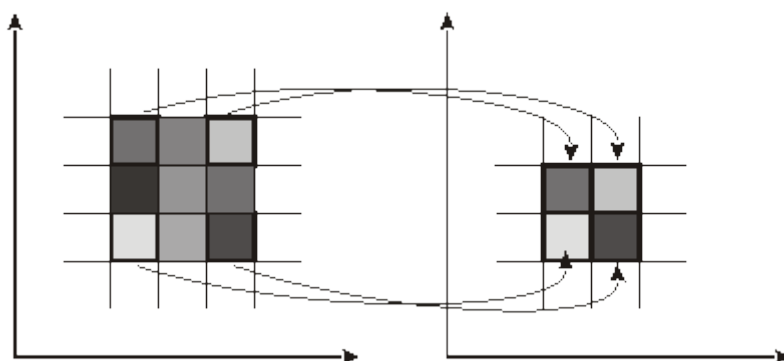
Rys. II.5. Dwukrotne zmniejszanie rozdzielczości (w poziomie i w pionie)

W pierwszym przypadku ma miejsce dwukrotna zmiana rozdzielczości w poziomie i w pionie. Przy zwiększaniu rozdzielczości (rysunek II.4) jeden piksel pierwotny jest zastępowany czterema pikselami o barwie piksela pierwotnego. Przy zmniejszaniu rozdzielczości (rysunek II.5) cztery piksele pierwotne są zastępowane jednym pikselem o średniej barwie pikseli pierwotnych.

W drugim przypadku zmiana rozdzielczości w pionie i w poziomie wynosi $3/2$ albo $2/3$. Przy zwiększaniu rozdzielczości pierwotne piksele są odpowiednio przesuwane (zob. rysunek II.6) natomiast pozostałe piksele (środkowe piksele na rysunku z prawej strony) są wyznaczane na zasadzie interpolacji między pikselami sąsiednimi. Przy zmniejszaniu rozdzielczości można na przykład zachować wybrane piksele tak jak to pokazano na rysunku II.7.

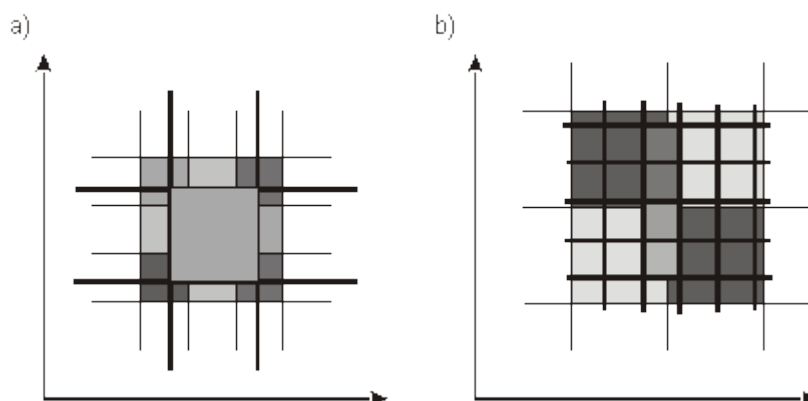


Rys. II.6. Zwiększenie rozdzielczości w stosunku 3/2 (w poziomie i w pionie)



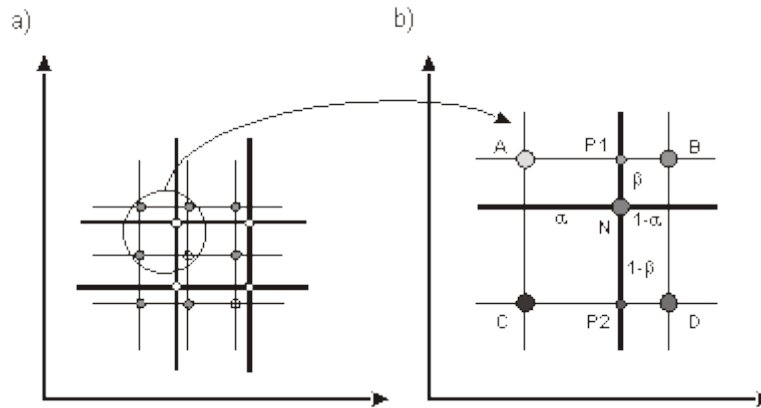
Rys. II.7. Zmniejszenie rozdzielczości w stosunku 2/3 (w poziomie i w pionie)

W innych przypadkach można na obraz o znanej rozdzielczości nałożyć nową siatkę prostokątną o docelowej rozdzielczości. Dla każdego oczka nowej siatki (nowego piksela) można określić barwę na podstawie analizy barw pikseli pierwotnego obrazu znajdujących się w obszarze oczka. Ilustruje to rysunek II.8. W przypadku zmniejszania rozdzielczości (rysunek II.8 a) nowy piksel ma barwę będącą wypadkową barw "przykrytych" pikseli (całych lub ich części). Podobnie określa się barwy nowych pikseli przy zwiększaniu rozdzielczości (rysunek II.8 b).



Rys. II.8. Zmiana rozdzielczości w dowolnym stosunku. a) Zmniejszanie rozdzielczości, b) zwiększanie rozdzielczości

W przypadku, gdy stosowana jest punktowa reprezentacja pikseli, obliczenia związane ze zmianą rozdzielczości można zrealizować inaczej. Załóżmy, że mamy sytuację jak na rysunku II.9a. Na istniejący obraz nałożona jest nowa siatka prostokątna o docelowej rozdzielczości. Na rysunku II.9b jest pokazany w powiększeniu fragment oznaczony kółkiem na rysunku II.9a.



Rys. II.9. Zmiana rozdzielczości przy punktowej reprezentacji pikseli

Problem polega na znalezieniu wartości piksela N z nowej siatki jeżeli znane są wartości (jasności) pikseli A, B, C, D w obrazie o początkowej rozdzielczości. Zakładamy oczywiście, że znane są współrzędne pikseli A, B, C, D i N. Weźmy pod uwagę piksele A i B. Metodą interpolacji możemy znaleźć wartość jasności w pomocniczym punkcie P1, leżącym na odcinku AB i dzielącym ten odcinek w stosunku $\alpha : (1-\alpha)$, jako:

$$P1 = \alpha B + (1 - \alpha) A.$$

Analogicznie możemy znaleźć wartość jasności dla punktu P2 leżącego między pikselami C i D, jako:

$$P2 = \alpha D + (1 - \alpha) C.$$

Wreszcie, znając wartości jasności w punktach P1 i P2, korzystając jeszcze raz z metody interpolacji, możemy znaleźć wartość jasności dla piksela N, jako:

$$N = \beta P2 + (1 - \beta) P1.$$

W wykładzie poznaliśmy problemy związane z reprezentacją obrazu w postaci cyfrowej. Poznaliśmy pojęcia piksela oraz rozdzielczości pikselowej obrazu. Wyjaśniony został problem zmiany rozdzielczości pikselowej obrazu. Pokazano również związek między rozdzielczością pikselową obrazu a jego wyglądem.

Przykładowe pytania i problemy do rozwiązania

1. Wyjaśnić pojęcie piksela.
2. Wyjaśnić pojęcie próbkowania obrazu.
3. Obliczyć pojemność pamięci potrzebna dla zapamiętania obrazu o rozdzielczości pikselowej 1024 x 768 x 24. Wynik proszę podać w megabajtach (MB)
4. Określić czas transmisji obrazu z problemu 3, jeżeli przepustowość sieci wynosi 56 kb/s (to znaczy, że w ciągu sekundy można przestać 56 kb informacji kodowanej dwójkowo).

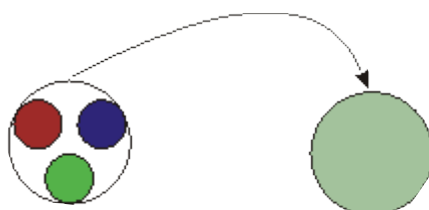
Wykład 3: Technika rastrowa (c.d.)

1. Wyświetlanie obrazów

Obraz reprezentowany w postaci mapy bitowej wcześniej czy później musi być przedstawiony w postaci czytelnej i zrozumiałej dla człowieka. Wykorzystuje się do tego, między innymi, różnego rodzaju urządzenia wyświetlające. Każde z tych urządzeń, niezależnie od konstrukcji czy zasady działania, musi zapewnić możliwość wyświetlania obrazu w taki sposób, żeby obserwator mógł odnieść wrażenie, że ogląda obraz jako jednolitą całość, podobnie jak to ma miejsce przy oglądaniu obrazów w otaczającym nas świecie. Oznacza to, że rozdzielczość wyświetlanego obrazu musi być dostatecznie duża na to, żeby system wzrokowy człowieka nie mógł zauważyć, że obraz składa się z pojedynczych pikseli. Warunki te w mniejszym lub większym stopniu spełniają współczesne monitory współpracujące z komputerami.

Znane są różne technologie wyświetlania obrazów. Niżej ograniczymy się do omówienia dwóch najczęściej spotykanych technologii: emisyjnej wykorzystywanej w monitorach z lampami CRT oraz nieemisyjnej wykorzystywanej w monitorach LCD, w których modulowane jest światło z zewnętrznego źródła światła.

W monitorach z reguły korzysta się z modelu RGB do reprezentowania barwy piksela. Powstaje pytanie jak na podstawie takiej informacji można uzyskać barwny punkt. Stosowane są dwa rozwiązania. W pierwszym z nich, tak zwanym przestrzennym mieszaniu barw, informacja o podstawowych składowych kolorach (R, G, B) piksela steruje równocześnie jasnością trzech małych elementów świecących. Obserwator znajdujący się w dostatecznie dużej odległości postrzega te trzy małe elementy jako jeden punkt o pewnym wypadkowym kolorze. Ilustruje to rysunek III.1.



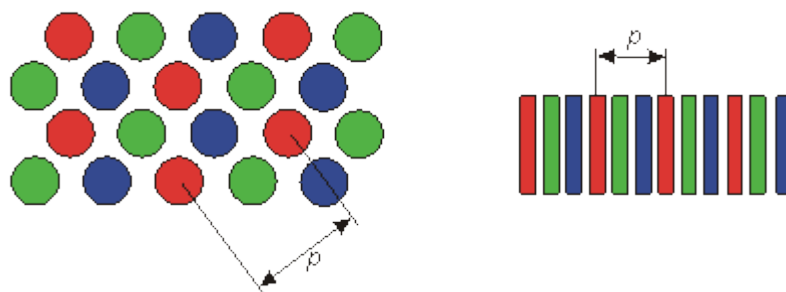
Rys. III.1. Ilustracja sposobu generowania barwnych pikseli na ekranie. Z lewej strony pokazane są trzy małe elementy świecące a z prawej strony barwny piksel widziany z dostatecznie dużej odległości

W drugim rozwiązaniu ten sam efekt wizualny uzyskuje się na zasadzie sekwencyjnego wyświetlania kolejnych składowych kolorów za każdym razem w tym samym miejscu. Jeżeli ta sekwencja jest powtarzana dostatecznie często, to oko nie jest w stanie rozróżnić poszczególnych składowych i obserwator odnosi wrażenie, że ogląda piksel o pewnej barwie.

2. Monitory CRT

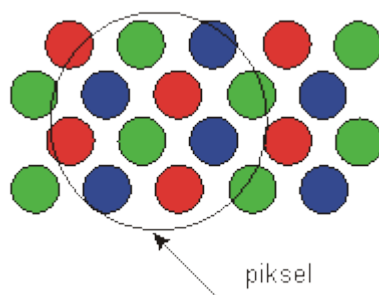
Wyjaśnijmy teraz zasadę wyświetlania obrazu wykorzystywaną w klasycznych monitorach CRT (cathode-ray tube). Są to monitory z lampą kineskopową. Poszczególne piksele są wyświetlane pierwszą z wymienionych wyżej metod. Ekran jest pokryty zbiorem tak zwanych triad. Każda triada jest to zestaw trzech małych plamek luminoforów. Plamki te są (w odpowiednim momencie) pobudzone równocześnie do świecenia przez trzy strumienie elektronów i emitują światło odpowiednio o barwie czerwonej (R), zielonej (G) i niebieskiej (B). Wielkość emisji przez każdą plamkę zależy od intensywności odpowiedniego strumienia elektronów. W efekcie obserwator znajdujący się w pewnej odległości od ekranu widzi jedynie punkt o barwie będącej wypadkową barw emitowanych przez poszczególne plamki luminoforów.

Na rysunku III.2 pokazano dwa wzory rozmieszczania plamek na ekranie spotykane we współczesnych monitorach CRT. Zaznaczony na rysunku parametr p określa odstęp między środkami dwóch sąsiednich plamek luminoforów generujących tę samą barwę. Określa on rozdzielczość lampy kineskopowej.



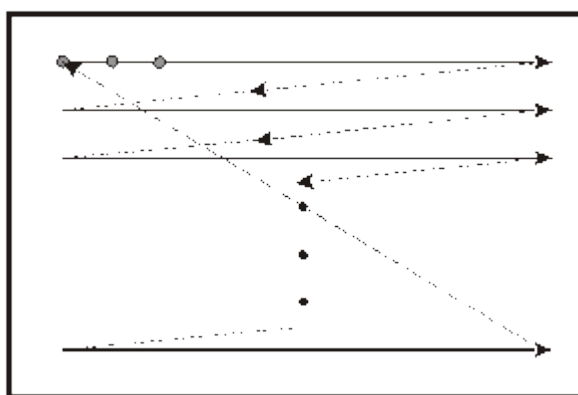
Rys. III.2. Wzory plamek luminoforów w monitorach CRT

Struktura plamek luminoforów stanowi osnowę dla wyświetlania obrazów reprezentowanych poprzez zbiór pikseli. Zintegrowana wiązka trzech strumieni elektronów z reguły pobudza do świecenia plamki luminoforów znajdujące się w obszarze obejmującym pewną liczbę nominalnych triad (por. rysunek III.3), przy czym każdy z trzech strumieni elektronów pobudza do świecenia tylko odpowiadające mu luminofory. Wielkość obszaru odpowiadającego pikselowi zależy od wybranej rozdzielczości monitora (800 \times 600, 1024 \times 768 itp.) i tak jest dobierana, żeby obraz o takiej rozdzielczości mógł być wyświetlony na całej dostępnej powierzchni ekranu.



Rys. III.3. Przykładowy obszar piksela na podstawie plamek luminoforów

Wyświetlanie obrazu na ekranie polega na kolejnym wyświetlaniu pikseli obrazu w ustalonej kolejności pokazanej na rysunku III.4. Najpierw są wyświetlane kolejne piksele pierwszego wiersza, potem drugiego wiersza itd., aż do ostatniego wiersza rastra. Po wyświetleniu całego obrazu proces zaczyna się od początku. Praktycznie oznacza to, że wiązka elektronów (składająca się z trzech strumieni elektronów) jest odchylana w lampie kineskopowej tak, żeby przesuwała się po ekranie i wyświetlała kolejne piksele zgodnie z opisaną kolejnością. W czasie, kiedy wiązka elektronów przesuwa się od końca jednego wiersza do początku następnego (tak zwany powrót poziomy) oraz kiedy przesuwa się od końca ostatniego wiersza do początku pierwszego wiersza (tak zwany powrót pionowy) intensywność strumieni jest zredukowana, luminofoery przestają świecić i nic nie jest wyświetlane na ekranie. Obraz uzyskany w wyniku jednokrotnego wyświetlenia całego rastra jest określany mianem ramki. Po wyświetleniu ramki rozpoczyna się wyświetlanie następczej ramki.



Rys. III.4. Kolejność wyświetlania pikseli na ekranie monitora CRT

Na to żeby obserwator nie odczuwał efektu migotania obrazu musi być wyświetlany z dostatecznie dużą częstotliwością. W praktyce częstotliwość ta nie powinna być mniejsza od 50 Hz. Częstotliwość odświeżania w dobrym monitorze powinna wynosić 85 Hz. (Przypomnijmy, że w kinie kolejne klatki są wyświetlane z częstotliwością 24 Hz i są dwukrotnie ekspozowane; każda klatka jest wyświetlana od razu w całości, a nie piksel po pikselu jak w naszym przypadku). Oznacza to, że na wyświetlenie jednej ramki dostępny jest czas nie większy niż kilkanaście 20 ms. Z kolei, jeżeli rozdzielczość ekranu wynosi na przykład 1024x768 pikseli, to na wyświetlenie pojedynczego piksela mamy czas rzędu kilku lub kilkunastu ns.

Przykład

Załóżmy, że monitor ma rozdzielczość 1024 × 768 pikseli oraz, że częstotliwość odświeżania wynosi 85 Hz. Przyjmijmy dodatkowo, że czas powrotu plamki w poziomie wynosi 4 × s a czas powrotu w pionie 40 × s. Należy obliczyć ile czasu jest do dyspozycji dla wyświetlenia jednego piksela.

Jeżeli częstotliwość odświeżania wynosi 85 Hz, to czas dostępny na wyświetlenie jednej ramki wynosi około 11,765 ms. W tym czasie muszą zostać wyświetlone wszystkie piksele (jest ich 1024 × 768) i musi nastąpić 767 powrotów poziomych

oraz jeden powrót pionowy. Oznaczając czas dostępny na wyświetlenie jednego piksela przez x otrzymujemy, że:

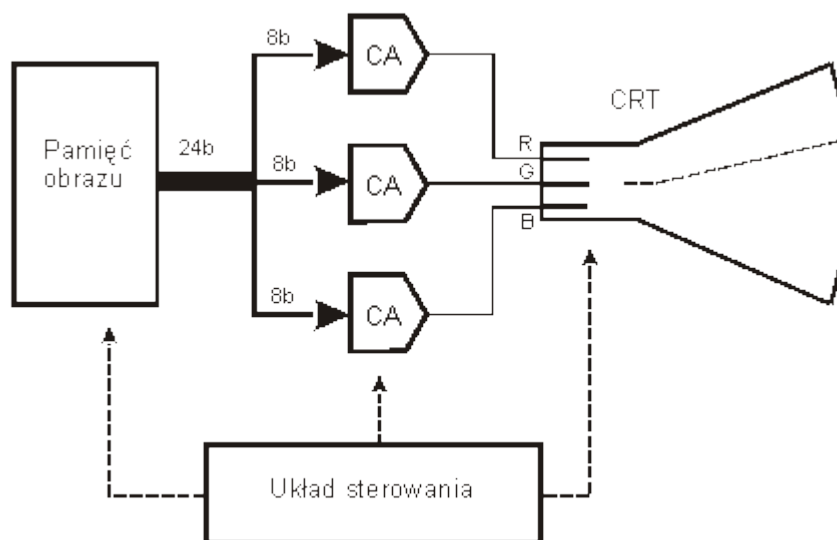
$$11\,765 \mu\text{s} = 1024 \times 768 \times x + 767 \times 4 \mu\text{s} + 40 \mu\text{s}$$

Stąd:

$$x \approx 11 \text{ ns.}$$

Informacja o wyświetlanym obrazie jest pamiętana w pamięci obrazu w postaci cyfrowej. Natomiast z punktu widzenia monitora CRT potrzebna jest informacja w postaci napięcia, a więc w postaci analogowej. Stąd należy pamiętać, że w torze między pamięcią obrazu a monitorem CRT musi znajdować się przetwornik cyfrowo-analogowy (CA) dokonujący odpowiedniej konwersji postaci cyfrowej na wymaganą postać analogową. W praktyce są to trzy przetworniki dokonujące odpowiednio konwersji dla składowych R, G i B.

Na rysunku III.5 pokazano schemat blokowy toru wyświetlania. Z pamięci obrazu pobierane są kolejno informacje o pikselach, które mają być wyświetlone. W typowym systemie informacja o barwie piksela jest kodowana za pomocą 24 bitów, po 8 bitów na każdą składową R, G i B. Każdy z trzech przetworników CA dokonuje konwersji odpowiednich 8 bitów na napięcia (na ogół z przedziału 0 V do 0,7 V), które są podawane na wejścia analogowe monitora CRT i sterują działaniami elektronowymi. Emitowane strumienie elektronów pobudzają do świecenia odpowiednie plamki luminoforów. Synchronizację toru wyświetlania zapewnia układ sterowania.



Rys. III.5. Schemat blokowy toru wyświetlania

Monitory CRT są urządzeniami nieliniowymi w tym sensie, że odpowiedź monitora (luminancja Y) na poziom sygnału wejściowego I jest nieliniowa. Na ogół zależność tę opisuje się wzorem $Y = K I^\gamma$ gdzie K jest współczynnikiem wagowym. Dla monitorów CRT współczynnik γ przyjmuje wartości z przedziału 2 - 2,7.

3. Monitory LCD

W monitorach LCD (Liquid-crystal displays) podstawowy element wyświetlający ma ustalony kształt. Do budowy elementów wyświetlających wykorzystuje się ciekłe kryształy. Są to elementy które nie emitują światła a modulują albo transmitują światło pochodzące z niezależnego źródła. Stosowane są dwa podstawowe rozwiązania różniące się umieszczeniem źródła światła względem elementów wyświetlających. W jednym z nich źródło światła znajduje się za panelem z elementami wyświetlającymi i światło jest przepuszczane. W drugim rozwiązaniu światło zewnętrzne po przejściu przez panel jest odbijane od powierzchni umieszczonej z tyłu panelu. W obu przypadkach zasada działania elementu wyświetlającego sprowadza się do tego, że zmiany zewnętrznego pola elektrycznego powodują zmiany orientacji molekuł w ciekłym kryształ, a w konsekwencji zmiany stanu tłumienia światła przez ciekły kryształ (między stanem przepuszczania i stanem blokowania światła).

Ciekłe kryształy blokują bądź przepuszczają światło i nie wnoszą niczego jeśli chodzi o barwę. Stąd, w celu uzyskania barwnych pikseli, elementy wyświetlające pokrywa się filtrami kolorowymi i wykorzystuje się źródła światła białego. W praktyce najczęściej umieszcza się obok siebie trzy elementy wyświetlające z trzema różnymi filtrami (R, G, B). Całość w efekcie tworzy pojedynczy piksel, który może przyjmować różne barwy.

W przypadku monitorów LCD mamy do czynienia z ustaloną strukturą rastra. Piksele są uporządkowane w postaci prostokątnej matrycy i każdy z nich odpowiada jednemu pikselowi wyświetlanego obrazu. Każdy piksel matrycy jest wybierany (adresowany) na zasadzie określenia numerów kolumny i wiersza na przecięciu których znajduje się w matrycy. Elementy matrycy są wybierane w kolejności takiej samej jak w przypadku monitorów CRT (por. rysunek III.4).

Każdy element matrycy jest wybierany na stosunkowo krótki czas. O ile jednak w monitorach CRT dzięki pewnej "bezwładności" luminofor pobudzony do świecenia emituje światło jeszcze przez pewien czas po zakończeniu pobudzania go, to w przypadku ciekłych kryształów efekt taki nie występuje. Stąd, dla uzyskania odpowiednich parametrów obrazu stosuje się rozwiązania umożliwiające przedłużenie czasu trwania stanu, w którym znalazł się element wyświetlający w chwili zaadresowania. W tym celu do każdego elementu wyświetlającego dołącza się niezależny tranzystor cienkowarstwowy (Thin Film Transistor), który wraz z niewielką pojemnością tworzy element pamiętający, podtrzymujący stanysterowania elementu wyświetlającego. Monitory, w których stosowane jest takie rozwiązanie są określane jako monitory z aktywną matrycą (w odniesieniu do monitorów, w których nie stosuje się takiego rozwiązania używa się określenia "monitory z pasywną matrycą"). Przykładem monitorów z aktywną matrycą są monitory TFT LCD.

Z faktu, że w monitorach LCD jest ustalony format matrycy pikseli wynika, że obraz może być wyświetlany tylko z taką rozdzielczością jaka wynika z rozmiarów matrycy. W przypadku konieczności wyświetlenia obrazów o innej rozdzielczości pikselowej musi nastąpić konwersja do rozdzielczości danego monitora.

Zwróćmy w tym miejscu uwagę na znaczenie pojęcia piksel. Generalnie oznacza ono element obrazu i odnosi się zarówno do obrazu reprezentowanego cyfrowo jak i do urządzenia wyświetlającego. Jednak w obu tych przypadkach znaczenie jest różne.

W odniesieniu do obrazu pierwotne znaczenie wiąże się z próbkowaniem obrazu i w tym kontekście jest to punktowa próbka oryginalnego obrazu, która niesie informację o barwie obrazu w teoretycznie nieskończenie małym punkcie. Jak wspomniano wcześniej, ze względu na wygodę, często przyjmuje się, że piksel reprezentuje pewien obszar a więc, że ma swój kształt i wymiary. W przypadku urządzenia wyświetlającego piksel ma odniesienie do fizycznego sposobu generowania elementarnego punktu obrazu i ma odpowiedni kształt i wymiary geometryczne. W praktyce, mimo tych różnic semantycznych, w obu przypadkach posługujemy się pojęciem piksela - na ogół jego konkretne znaczenie wynika jednoznacznie z kontekstu.

Podsumowanie

W trakcie wykładu poznaliśmy problemy związane z wyświetlaniem obrazów cyfrowych na dwóch podstawowych typach monitorów: CRT oraz LCD. Poznaliśmy zasadę wyświetlania rastrowego oraz metody wyświetlania barwnych pikseli. Wiemy również z jaką częstotliwością powinny być wyświetlane obrazy na ekranie.

Przykładowe pytania i problemy do rozwiązania

1. Wyjaśnić zasadę wyświetlania rastrowego.
2. Uzasadnić konieczność stosowania przetworników cyfrowo-analogowych w torze wyświetlania z monitorem CRT.
3. Porównać metody wyświetlania barwnego piksela w monitorze CRT i w monitorze LCD.
4. Jak można rozwiązać problem wyświetlania obrazu na monitorze LCD jeżeli rozdzielczość pikselowa obrazu jest większa niż rozdzielczość monitora? Dla przykładu założyć, że rozdzielczość obrazu wynosi 1024 \times 768 a rozdzielczość monitora wynosi 800 \times 600.

Wykład 4: Barwa w grafice komputerowej

Otoczający nas świat jest światem barwnym. Stąd również obrazy, które chcemy generować korzystając z komputera powinny być obrazami barwnymi. Okazuje się jednak, że sposób opisu barwy odpowiedni dla zastosowań komputerowych nie jest prosty. Wynika to już stąd, że samo pojęcie barwy nie jest oczywiste. W trakcie wykładu zostanie wyjaśnione pojęcie barwy oraz omówione zostaną wybrane zagadnienia związane z percepcją barwy, istotne dla zastosowań w grafice komputerowej.

1. Pojęcie barwy

Pojęcie barwy (koloru) jest powszechnie znane i stosowane. Warto jednak zwrócić uwagę na fakt, że barwa jako taka nie istnieje - nie jest właściwością żadnego obiektu fizycznego. Jest to wrażenie percepcyjne człowieka będące skutkiem

pobudzenia systemu wzrokowego przez fale elektromagnetyczne z zakresu widzialnego. Wrazenia wzrokowe rozumiane pod pojęciem barwy są wynikiem interakcji wielu czynników takich jak rodzaj źródła światła, właściwości odbijania światła przez obserwowane obiekty, czułość obserwatora.

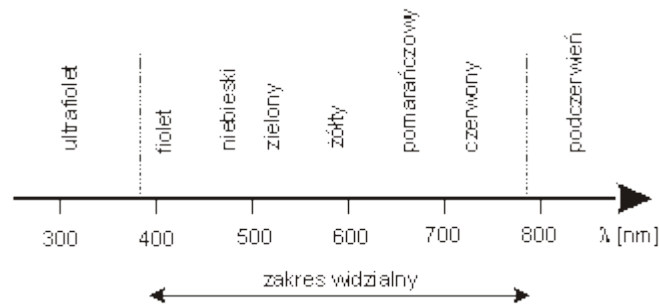
Barwa jest więc atrybutem wrażeń wzrokowych. Barwy obiektów zależą od interakcji trzech elementów: źródeł światła, obiektów i systemu wzrokowego człowieka. Źródła światła są fizycznymi emiterami widzialnej energii o określonych widmowych rozkładach energetycznych. Obiekty są charakteryzowane przez ich geometryczne i widmowe rozkłady energii, którą odbijają albo transmitują. System wzrokowy człowieka jest charakteryzowany przez czułość na poszczególne długości fal.

W tym kontekście nie może budzić zdziwienia fakt, że wszelkie próby definiowania barwy napotykają wiele trudności. Problem okazuje się jeszcze bardziej złożony przy próbach dokładnego opisanie i specyfikowania barwy. Od dawna są prowadzone prace w tym kierunku. Opracowano wiele modeli, które są wykorzystywane w praktyce niemniej wciąż trwają prace nad jeszcze lepszymi sposobami opisu barwy.

Teoria barwy jest bardzo obszerna i obejmuje wiele dziedzin wiedzy, przy czym każda z tych dziedzin rozwija tę teorię na swoje potrzeby i często stosuje swoje słownictwo. Stąd nie ma jednolitości nazewnictwa. W szczególności dotyczy to pojęć kolor, barwa czy odcień barwy. Tutaj przyjmujemy, że podobnie jak w mowie potocznej kolor i barwa są synonimami. Odcień barwy będzie jednoznacznie określał barwy widmowe, z tym że niejednokrotnie, jeżeli nie będzie to budziło wątpliwości, zamiast używać określenia odcień barwy będziemy używali określenia barwa. Niżej ograniczymy się do podania wybranych zagadnień związanych z barwą, kierując się ich przydatnością z punktu widzenia potrzeb grafiki komputerowej.

2. Percepcja barwy

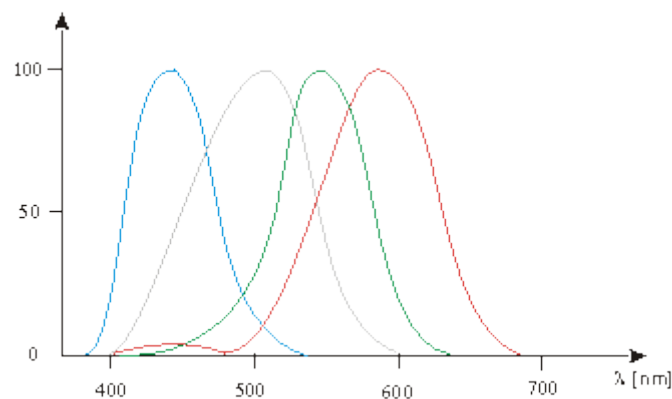
Zakres spektrum widzialnego obejmuje fale o długości od 380 nm do 780 nm (są to wartości przybliżone). Z punktu widzenia fizycznego barwa jest tym pojęciem, które odpowiada długości fali świetlnej - różnym długościom fali świetlnej w widmie widzialnym odpowiadają różne barwy. Barwy związane tylko z jedną długością fali są określane jako monochromatyczne. Są jednak również barwy, które nie mogą być przypisane określonym pojedynczym długościom fali. Na przykład purpury są określane przez kombinację dwóch fal z przeciwległych końców spektrum odpowiadających barwom niebieskiej i czerwonej. Na rysunku IV.1 pokazano rozmieszczenie wybranych barw w widmie widzialnym.



Rys. IV.1. Widmo widzialne

Z punktu widzenia fizycznego nie ma różnicy między falami o różnych długościach, czy częstotliwościach. Fakt, że postrzegamy fale o różnych długościach jako różne barwy wynika jedynie ze sposobu interpretacji przez mózg pobudzeń odbieranych przez elementy światłoczułe znajdujące się w oczach. Te elementy to czopki i pręciki znajdujące się o obrębie siatkówek.

Pręciki generalnie są czułe na światło w całym zakresie widma widzialnego. Są one przede wszystkim receptorami "luminancji" i zapewniają widzenie przy słabym oświetleniu (widzenie zmierzchowe). Natomiast czopki mają mniejszą czułość i wymagają dużego poziomu oświetlenia. Każdy z trzech występujących typów czopków (L, M, S) jest bardziej czuły w określonym zakresie widma (por. rysunek IV.2). Czopki umożliwiają percepcję zjawisk barwnych.

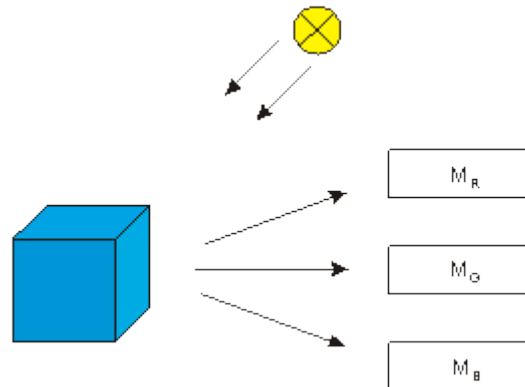


Rys. IV.2. Względna czułość czopków i pręcików funkcji długości fali. Linia szara - pręciki, linie: czerwona, niebieska i zielona - czopki

Na percepcję barwy składa się kilka czynników: charakterystyka źródła światła, charakterystyki odbicia lub transmisji dla obserwowanego obiektu, charakterystyki czułości poszczególnych receptorów (wszystkie charakterystyki w funkcji długości fali świetlnej). W mózgu następuje ostateczna interpretacja docierającej informacji i identyfikacja postrzeganego koloru obiektu.

Najprostszy model percepcji barwy zilustrowano na rysunku IV.3. Światło odbite od obiektu pada na trzy rodzaje receptorów. Każdy z nich generuje pewną odpowiedź M_R , M_G i M_B . Suma tych odpowiedzi określa postrzeganą barwę. Poszczególne receptory występujące w tym modelu określa się jako receptory R (czerwony), G (zielony) i B (niebieski). Zauważmy, że w tym modelu przyjmuje się, że receptory

są czułe na trzy wybrane barwy, podczas gdy w rzeczywistości charakterystyki czułości poszczególnych czopków rozciągają się w szerokim zakresie widma widzialnego, jak ilustruje to rysunek IV.2.



Rys. IV.3. Model percepcji barwy

Zwróćmy uwagę, że zależnie od oświetlenia obiekty o różnych właściwościach odbijania światła mogą być postrzegane jako obiekty o takiej samej barwie. Efekt taki jest określany mianem metameryzmu. Podobnie dwa obiekty o nominalnie takiej samej barwie mogą być postrzegane różnie zależnie od oświetlenia przez różne źródła światła. Dodatkowo percepcja zależy od cech osobniczych obserwatorów, od poziomu oświetlenia, od tła itp.

Ponieważ są trzy typy receptorów odpowiedzialne za rozpoznawanie koloru to można się spodziewać, że dobierając odpowiednie trzy źródła światła i ich procentowy udział, można uzyskać percepcję pożądaną barwy. Spostrzeżenie to leży u podstaw generowania obrazów barwnych w urządzeniach wyświetlających gdzie wykorzystuje się źródła trzech podstawowych barw R, G i B. Barwę uzyskuje się na zasadzie mieszania addytywnego polegającego na dodawaniu składowych. W przypadku druku gdzie ma miejsce absorpcja światła przez pigmenty mamy do czynienia z metodą subtraktywną (różnicową). W tym przypadku typowy zestaw barw podstawowych obejmuje barwy C (cyjan), M (magenta) i Y (żółty). Warto jednak dodać, że bazowanie na trzech barwach podstawowych jest pewnym uproszczeniem z punktu widzenia rzeczywistej percepcji kolorów przez system wzrokowy człowieka. Rzeczywistość jest bardziej złożona i w praktyce nie jest możliwe dobranie takich trzech barw podstawowych, które pozwalałyby reprodukcować dowolne barwy.

Podsumowanie

Po zapoznaniu się z wykładem powinniśmy zdać sobie sprawę z trudności jakie wiążą się z definiowaniem i opisem barwy. Mimo, że na pozór pojęcie barwy jest oczywiste, to jednak przy próbie formalizowania zagadnień związanych z opisem barwy dla potrzeb grafiki komputerowej okazuje się, że zagadnienie jest wciąż trudne. W trakcie wykładu poznaliśmy również zagadnienia związane z percepcją barwy przez człowieka. Znajomość tych zagadnień jest istotna z punktu widzenia zarówno metod opisu barwy jak i metod generowania i wyświetlania obrazów z

wykorzystaniem komputerów. Problemy związane z modelami barw są omawiane w następnym wykładzie.

Przykładowe pytania i problemy do rozwiązania

1. Proszę wyjaśnić od czego zależy obserwowana barwa obiektu.
2. Proszę omówić model percepcji barwy.
3. Co to są barwy monochromatyczne?
4. Czy za pomocą trzech barw podstawowych R, G i B można uzyskać dowolną barwę widzialną?

Wykład 5: Barwa w grafice komputerowej (c.d.)

Streszczenie

W poprzednim wykładzie wyjaśniliśmy podstawowe zagadnienia związane z pojęciem barwy i z percepcją barwy przez człowieka. Wskazaliśmy również na trudności związane z opisem barwy. Trudności te są między innymi przyczyną tego, że opracowano wiele modeli barw i jak dotychczas żaden z nich nie pozwala opisywać wszystkich efektów związanych z barwą. W tym wykładzie poznamy różne modele barw wykorzystywane w grafice komputerowej. Zwrócimy również uwagę na problem związany z przenośnością obrazów barwnych i metody jego rozwiązywania.

1. Modele barw

1.1. Skala szarości

Przed omówieniem różnych modeli barw zwróćmy uwagę na bardzo często wykorzystywaną w praktyce skalę szarości obejmującą odcienie szarości zawarte między bielą a czernią. Najczęściej dla opisanego odcienia szarości wykorzystuje się słowo ośmiobitowe. Wtedy możemy operować 256 odcieniami szarości.

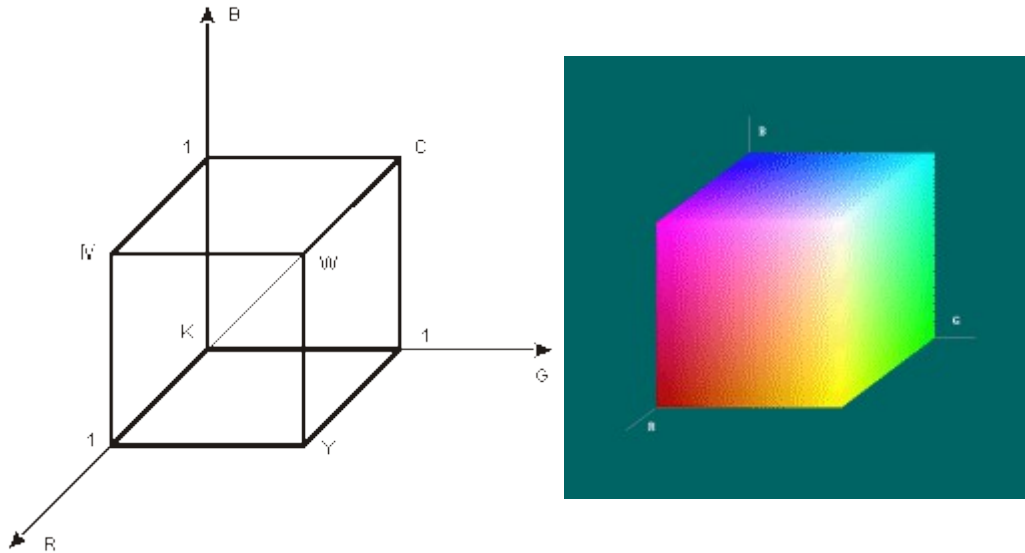
Na rysunku V.1 pokazano przykładową skalę szarości.



Rys. V.1. Skala szarości

1.2. Model RGB

Model RGB bazuje na założeniu, że dostępne są trzy barwy podstawowe R, G i B. Wszystkie barwy, które można uzyskać za pomocą mieszania tych składowych są reprezentowane przez punkty należące do sześcianu jednostkowego zdefiniowanego w układzie współrzędnych R, G, B. Model jest pokazany na rysunku V.2. W wierzchołkach sześcianu są reprezentowane barwy R,G,B,C,M,Y oraz czarna K (black) i biała W (White). Na przekątnej łączącej wierzchołki K i W znajdują się odcienie szarości.

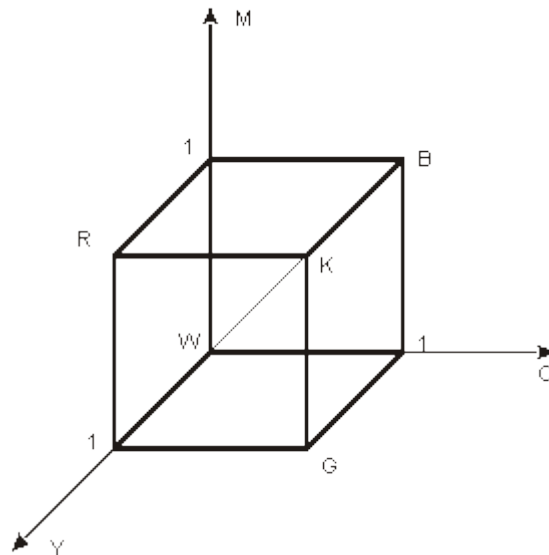


Rys. V.2. Model barw RGB. W wersji kolorowej widoczne są tylko barwy znajdujące się na powierzchni sześcianu.

Z punktu widzenia zastosowań grafiki komputerowej model barw jest modelem dyskretnym. Każda ze składowych może być reprezentowana za pomocą słów o pewnej długości. W szczególności jeżeli każde słowo reprezentujące pojedynczą składową ma 8 bitów, to barwny punkt jest reprezentowany przez 24 bity. System, w którym jest stosowana taka reprezentacja jest określany jako system pełnokolorowy (ang. full color, true color). Liczba różnych barw, które można uzyskać w takim systemie wynosi 2^{24} co w przybliżeniu jest równe 16,7 milionów. Jest to niewątpliwie duża liczba barw i można się zastanawiać, czy system wzrokowy jest w stanie rozróżnić taką liczbę barw. Nie można jednak zapomnieć, że są to jedynie barwy, które można uzyskać metodą mieszania addytywnego trzech składowych podstawowych. Poza tym są takie barwy, których nie można uzyskać w ten sposób.

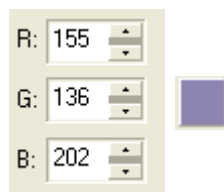
1.3. Model CMY

Model CMY ma podobne cechy jak model RGB. Jest on również reprezentowany za pomocą jednostkowej kostki sześciennej, z tym, że w układzie współrzędnych C,M,Y. Model ten pokazano na rysunku. Model ten jest modelem subtraktywnym i wykorzystywany jest głównie w drukarstwie. W praktyce stosowany jest model CMYK, w którym ze względów praktycznych dodatkowo uwzględnia się barwę czarną - dzięki temu uzyskuje się lepszą czerń niż w przypadku uzyskiwania czerni ze składowych C, M i Y.



Rys. V.3. Model barw CMY

W odniesieniu do obu modeli RGB i CMY mówi się, że są to modele nieintuicyjne. Chodzi o to, że korzystając z tych modeli przeciętnemu użytkownikowi stosunkowo trudno jest dobrać takie wartości składowych przy których uzyska się pożądaną barwę. Przy interaktywnej pracy z programami graficznymi często mamy do dyspozycji wyświetlone na ekranie suwaki cyfrowe (tak jak na rysunku V.4), które umożliwiają ustawienie odpowiedniej wartości składowych, na przykład R, G i B (każda składowa w zakresie od 0 do 255). Ponadto z reguły dostępne jest okno, w którym wyświetlana jest barwa wypadkowa.



Rys. V.4. Przykład suwaków cyfrowych umożliwiających dobór składowych potrzebnych dla uzyskania pożądanego koloru

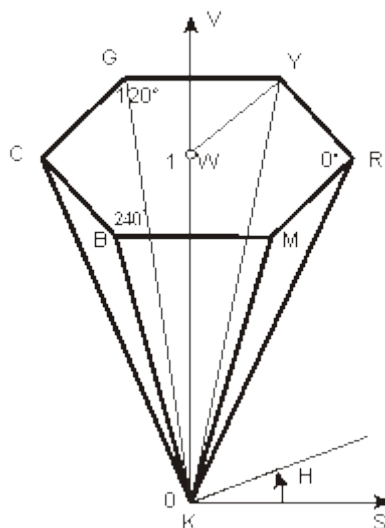
1.4. Model HSV

W mowie potocznej dla intuicyjnego opisu kolorów używa się takich określeń jak odcień barwy (barwa), nasycenie i jasność. Odcień barwy określa się za pomocą nazw. I tak mówi się o barwie czerwonej, niebieskiej itd.

Nasycenie określa gdzie dany odcień barwy znajduje się między czystą barwą monochromatyczną (o nasyceniu 100%) a barwą białą (nasycenie 0%).

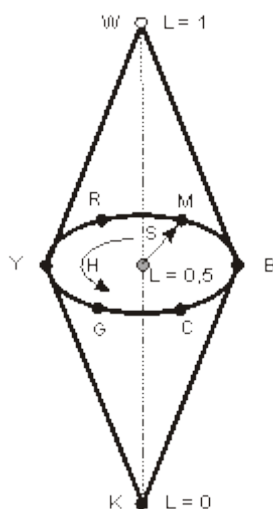
Jasność (wartość) koloru intuicyjnie określa postrzeganą intensywność światła o danej barwie. Określenie to pozwala rozróżniać barwy jasne od ciemnych, na przykład jasnoróżową od ciemnoróżowej.

Na bazie takiego intuicyjnego opisu kolorów został skonstruowany model HSV (rysunek V.5). W modelu tym parametr H (od. ang. Hue) określa odcień barwy wyrażany w stopniach od 0° do 360° , parametr S (od ang. Saturation) określa nasycenie wyrażane w skali od 0 do 1 albo w procentach od 0% do 100%. Parametr V (od ang. Value) określa jasność (wartość) w skali od 0 (dla czerni) do 1 (dla bieli). W modelu tym na obwodzie dowolnego przekroju poziomego leżą barwy nasycone, wewnątrz przekroju znajdują się barwy nienasycone a o jasności barw decyduje odległość przekroju od punktu K reprezentującego barwę czarną. Na osi V znajdują się odcienie szarości.



Rys. V.5. Model HSV

Korzystając z modelu HSV można stosunkowo łatwo dobierać składowe H, S i V określające pożądany kolor. W tym celu można najpierw określić odcień barwy H a następnie dobrać wartości nasycenia S i jasności H. W tym sensie model HSV jest uważany za model intuicyjny. Wiele pakietów graficznych umożliwia określanie kolorów właśnie w tym modelu albo w pochodnym modelu HLS pokazanym na rysunku V.6. W modelu HLS składowa L oznacza jasność (od ang. Lightness).



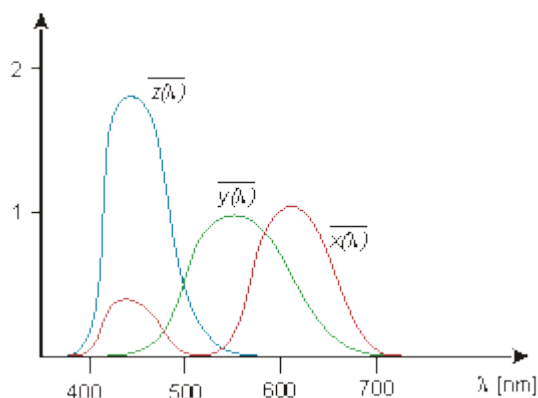
Rys. V.6. Model HLS

Oczywiście ostatecznie na monitorze wyświetlany jest kolor uzyskany w wyniku konwersji do modelu RGB.

1.5. Model CIE XYZ

O ile opisany wyżej prosty model HSV odpowiada potocznemu intuicyjnemu sposobowi opisywania kolorów, to z punktu widzenia precyzyjnego opisu kolorów, potrzebnego na przykład w kolorymetrii, jest on mało użyteczny. Stąd opracowane zostały inne modele kolorów.

W 1931 roku Międzynarodowa Komisja Oświetleniowa (Commission Internationale de l'Éclairage, w skrócie CIE) opracowała model kolorów, w którym za punkt wyjścia przyjęto trzy teoretyczne barwy podstawowe X, Y i Z (nie mające rzeczywistych odpowiedników). Dla tych trzech barw określono składowe trójchromatyczne widmowe: $\bar{x}(\lambda)$, $\bar{y}(\lambda)$, $\bar{z}(\lambda)$ (por. rysunek V.7). Znając te funkcje można określić wartości składowych X, Y i Z potrzebnych dla wyświetlenia określonego koloru.

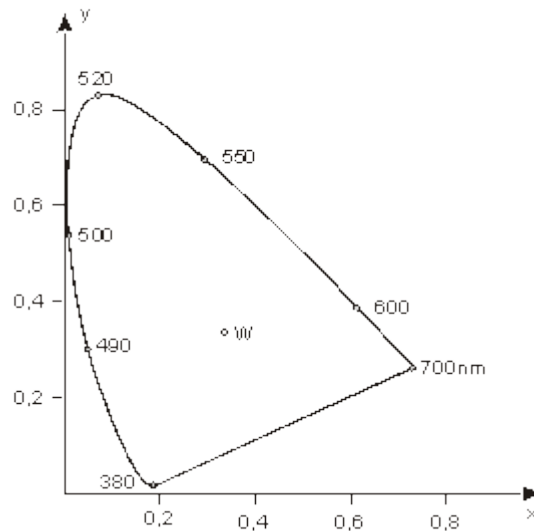


Rys. V.7. Składowe trójchromatyczne widmowe przyjęte przez CIE

Z kolei, ze względów praktycznych, Komisja CIE zdefiniowała bardziej użyteczny sposób opisywania kolorów, a mianowicie model Yxy. W modelu tym została zachowana składowa Y, która reprezentuje jasność źródła światła. Pozostałe dwie wielkości, określane jako współrzędne trójchromatyczne, zostały zdefiniowane następująco

$$x = \frac{X}{X + Y + Z} \quad \text{oraz} \quad y = \frac{Y}{X + Y + Z} .$$

We współrzędnych x, y został skonstruowany wykres chromatyczności pokazany na rysunku V.8.



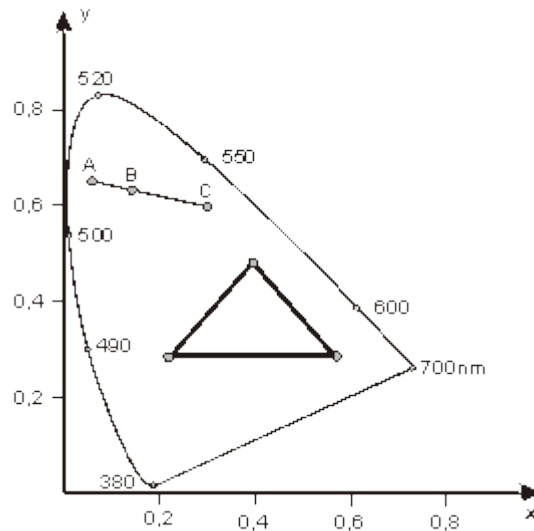
Rys. V.8. Wykres chromatyczności

Wykres ten razem z luminancją Y pozwala dobrze opisywać różne barwy. W wykresie chromatyczności na części krzywoliniowej obwodu znajdują się wszystkie barwy widmowe nasycone. Na odcinku w dolnej części wykresu znajdują się purpury. W środkowej części wykresu znajduje się punkt W reprezentujący barwę białą. We wnętrzu wykresu znajdują się barwy nienasycone. Przedstawiony wykres chromatyczności jest przekrojem modelu trójwymiarowego dla ustalonej wartości Y. Cały model reprezentuje wszystkie barwy widzialne.

Korzystając ze współrzędnych chromatycznych należy pamiętać, że dla pełnego opisu barwy trzeba podać jeszcze wartość luminacji Y.

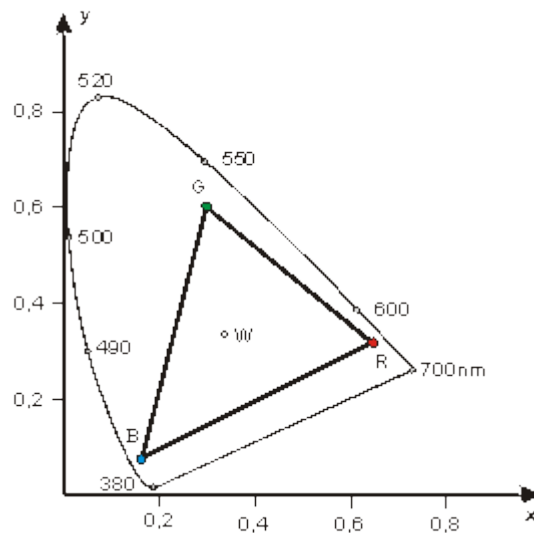
Zwróćmy uwagę na niejednoznaczność potocznego pojęcia barwy białej. W istocie istnieje wiele różnych bieli. Są też różne definicje bieli. Niektóre z nich bazują na koncepcji ciała doskonale czarnego analizowanego w fizyce. Przykładowo, tak zwana biel 6500 jest to kolor ciała doskonale czarnego ogrzanego do temperatury 6500 °K. Biel ta odpowiada światłu dziennemu; $x = 0,3127$, $y = 0,3297$). Biel równoenergetyczna E ma współrzędne $x = y = 0,333$. Biel C reprezentuje światło słoneczne obserwowane w południe ($x = 0,3101$, $y = 0,3162$).

Jeżeli na wykresie chromatyczności zaznaczymy położenie dwóch barw (na przykład A i C na rysunku V.9), to barwa (na przykład B na rysunku V.9), którą uzyskamy po zmieszaniu barw początkowych będzie leżała na odcinku łączącym te barwy. Stąd, jeżeli zmieszamy trzy barwy, to barwa wypadkowa będzie leżała wewnątrz trójkąta, w którego wierzchołkach będą barwy pierwotne. Ilustruje to rysunek.



Rys. V.9. Punkt reprezentujący barwę B powstała w wyniku zmieszania barw A i C leży na odcinku łączącym punkty reprezentujące barwy A i C. Punkty reprezentujące barwę uzyskaną w wyniku zmieszania trzech barw początkowych leżą we wnętrzu trójkąta, którego wierzchołki reprezentują barwy pierwotne.

Korzystając z tej właściwości można na wykresie chromatyczności zobrazować różne barwy z modeli barw bazujących na barwach podstawowych. W szczególności można przedstawić model RGB tak jak to pokazuje rysunek V.10 - barwy z wnętrza sześciianu RGB znajdują się wewnątrz trójkąta.



Rys. V.10. Model RGB na tle wykresu chromatyczności

Można zauważyć, że zestaw barw reprezentowanych przez model RGB (tak zwana gama barw RGB) nie obejmuje wszystkich barw widzialnych. Można również zauważyć, że żaden inny zestaw trzech barw widzialnych nie umożliwia uzyskania wszystkich barw widzialnych.

2. Przestrzenie percepcyjnie równomierne

Przestrzeń Yxy obok swych zalet ma też ograniczenie a mianowicie nie jest percepcyjnie równomierna. Oznacza to, że jednakowym zmianom wartości parametrów nie odpowiadają jednakowe zmiany postrzeganych kolorów. W celu usunięcia tej wady komisja CIE opracowała w 1976 roku przestrzenie barw $L^*a^*b^*$ (w skrócie CIELAB) i $L^*u^*v^*$ (w skrócie CIELUV). Obie przestrzenie są percepcyjnie równomierne.

Przestrzeń $L^*u^*v^*$ jest używana głównie w odniesieniu do urządzeń emitujących światło, natomiast przestrzeń $L^*a^*b^*$ jest używana głównie do specyfikowania kolorów obiektów (tj. światła odbitego). Poszczególne składowe reprezentują odpowiednio L^* - jasność, a^* - położenie między barwami czerwoną a zieloną, b^* - położenie na odcinku między barwami niebieską a żółtą. (Konstrukcja obu wymienionych przestrzeni bazuje na teorii barw przeciwstawnych, której geneza wywodzi się ze spostrzeżenia, że nie spotyka się określeń takich barw jak czerwono-zielona czy żółto-niebieska, natomiast postrzegane są barwy będące kombinacją czerwonej i niebieskiej, zielonej i żółtej.)

3. Systemy porządkujące barwy

Systemy tego typu pozwalają opisywać barwy, czy też systematyzować je, bez korzystania z matematycznego opisu w odniesieniu do wielkości mierzalnych fizycznie. Systemy takie są niezależne od urządzeń. Określają one pewne uporządkowanie kolorów i zawierają logiczny system oznaczeń. Mają percepcyjnie zrozumiałe wymiary. Zawierają stabilne, dokładne i precyzyjne próbki.

Nie wszystkie systemy spełniają te warunki. Jednym z takich systemów jest Pantone Color Formula Guide, który specyfikuje kolory farb ale ich nie porządkuje, ponieważ nie zawiera ciągłej skali. Jest to w zasadzie system nazewnictwa kolorów. System Pantone jest zawarty w Pantone Color Formula Guide. Znajduje się tam 1012 próbek. Każdej z nich jest przypisany numer. Kolor uzyskany w druku powinien możliwie dobrze przybliżać odpowiednią próbkę Pantone. System Pantone zawiera również Pantone Process Color Imaging Guide. Tutaj znajdują się 942 próbki kolorów, które mogą być symulowane w procesie druku czterokolorowego (CMYK).

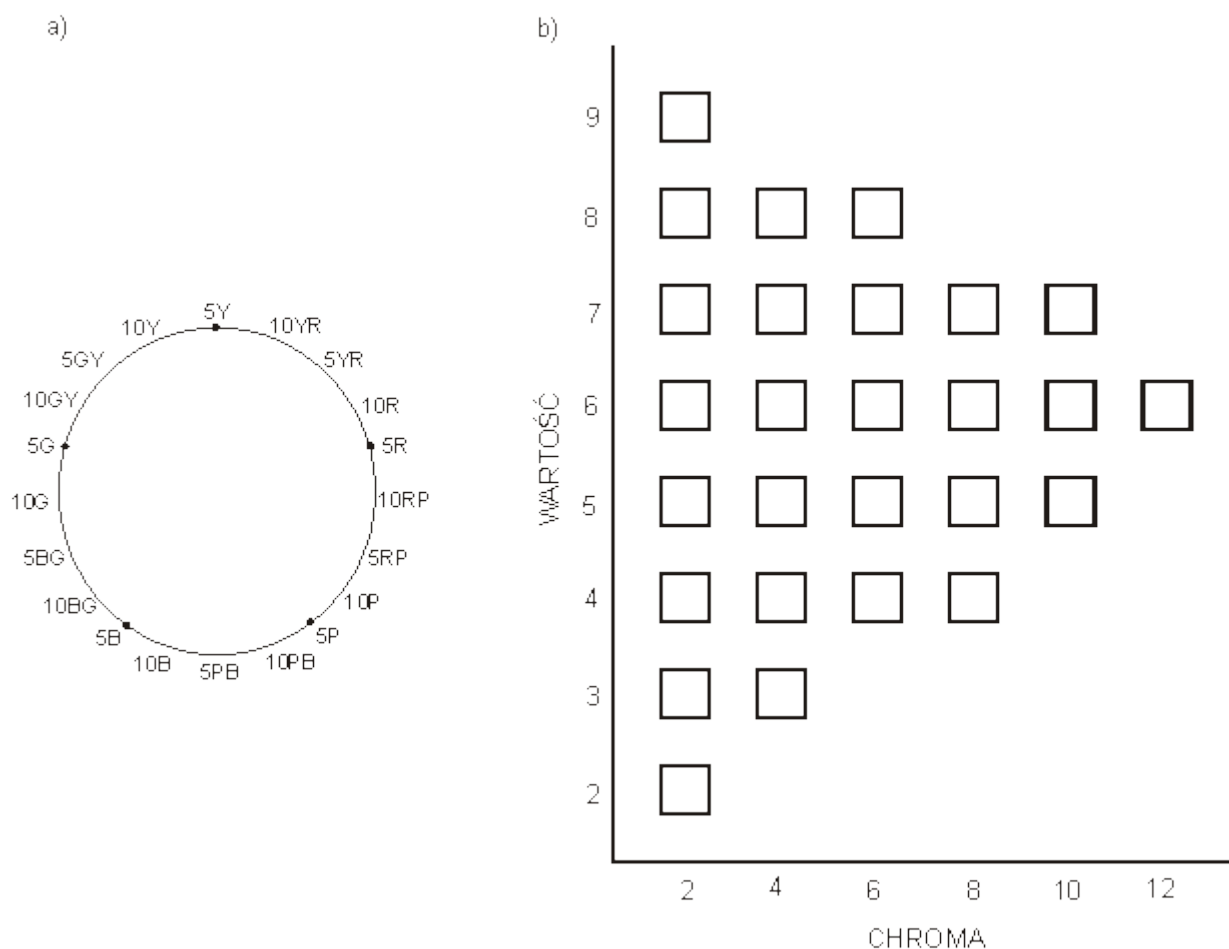
Innym systemem znacznie bogatszym jest system PostScript Process Color Guide opublikowany przez firmę Agfa, zawierający ponad 16 000 próbek kolorów.

Jednym z najbardziej znanych systemów porządkujących jest system Munsella. Barwa jest tu reprezentowana za pomocą trzech atrybutów V, H, C - odpowiednio: wartość, odcień barwy, chroma. Munsell stawiał sobie za cel takie określenie kolorów, żeby dla każdego z trzech percepcyjnych wymiarów występowały równe zmiany wrażeń wzrokowych.

Na skali wartości wyróżniono 10 głównych przedziałów. Czerni przypisano wartość 0, bieli 10 a odcieniom szarości pośrednie wartości. Barwy rozmieszczono na okręgu i podzielono na 5 grup (purpury, niebieskie, zielone, żółte i czerwone oznaczone odpowiednio jako 5P, 5B, 5G, 5Y i 5R). Okrąg podzielono na pięć percepcyjnie równomiernych przedziałów. Dalej wyróżniono 5 pośrednich barw 5PB, 5BG, 5GY, 5YR, 5RP. Z kolei dla każdej z dotychczas określonych 10-ciu barw określono 10 kolejnych barw (por. rysunek V.11 a). W sumie określono 100 barw o numerach

całkowitych znajdujących się w percepcyjnie równomiernych odstępach. Barwy o numerach ułamkowych znajdują się pomiędzy sąsiednimi barwami o numerach całkowitych.

Trzeci atrybut chroma ma określone równe percepcyjnie przedziały o numerach zwiększających się od zera dla neutralnych próbek w kierunku próbek o większej zawartości barwy. Zakres skali chromy jest różny dla różnych barw (por. rysunek V.11 b).



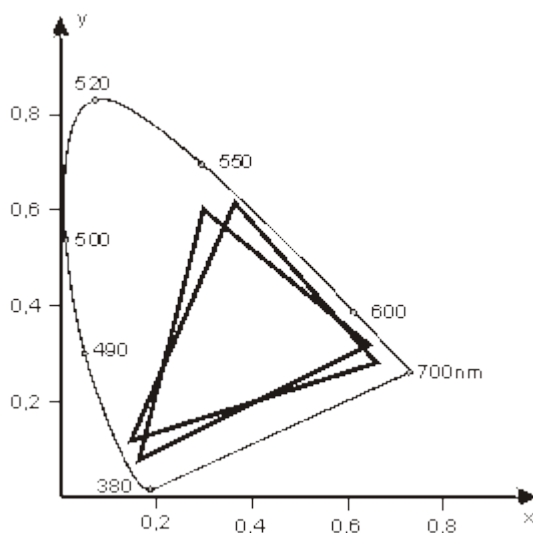
Rys. V.11. System Munsella. a) Koło odcieni barw, b) płaszczyzna wartość/chroma dla określonego odcienia barw

Poszczególne barwy są określane przez wartości trzech atrybutów w kolejności odcień barwy, wartość, chroma. System Munsella jest zawarty w opracowaniu Munsell Book of Color. Znajduje się tam około 1500 uporządkowanych różnych próbek kolorów.

4. Systemy zarządzania kolorem

W grafice komputerowej mamy do czynienia z różnymi urządzeniami, które umożliwiają pozyskiwanie obrazów w postaci cyfrowej (skanery, fotograficzne aparaty cyfrowe) bądź ich reprodukcję (monitory, drukarki). Każde z nich może mieć inną gamę kolorów - obszar barw odtwarzalnych (por. rysunek V.12). W takiej

sytuacji może stać się niemożliwe wierne reprodukowanie barw na dwóch urządzeniach. Możliwym rozwiązaniem jest ograniczenie gam kolorów dwóch urządzeń do wspólnego obszaru gam tych urządzeń. Inne rozwiązania zakładają odwzorowanie gamy kolorów jednego urządzenia na gamę kolorów innego urządzenia za pomocą odpowiednich transformacji w wybranej wspólnej przestrzeni kolorów. W praktyce dokonuje się najczęściej projekcji koloru w kierunku środka gamy kolorów, co powoduje redukcję nasycenia i w pewnym stopniu jasności, przy zachowaniu barwy.



Rys. V.12. Wykres chromatyczności z przykładowymi różnymi gamami kolorów dwóch urządzeń

Inne możliwe rozwiązanie polega na korzystaniu z jednej wspólnej przestrzeni barw niezależnej od poszczególnych urządzeń. Przestrzenią taką może być jedna z przestrzeni zdefiniowanych przez Komisję CIE. Z kolei dla każdego urządzenia można określić tak zwany profil urządzenia, który umożliwi odwzorowanie między przestrzenią kolorów danego urządzenia a przyjętą przestrzenią.

Niektóre firmy (Kodak, Agfa) opracowały odpowiednie systemy zarządzania kolorami. Został również opracowany standardowy format dla opisu profili (ICC). W systemie Windows pliki zawierające profile mają rozszerzenie .icm. Profile nowych urządzeń są często zawarte w sterownikach urządzeń. Są to oczywiście domniemane profile - do użytkownika należy zapewnienie właściwego zestrojenia urządzenia.

System zarządzania kolorem ma za zadanie zarządzanie profilami oraz przesyłanie obrazów między urządzeniami z uwzględnieniem profili. Trzeba jednak pamiętać, że nie ma jednego uniwersalnego sposobu dokonywania transformacji kolorów jeżeli gamy kolorów urządzeń nie są identyczne. Stąd, mimo wszystko, trzeba liczyć się z tym, że mogą występować różnice przy reprodukcji obrazów w różnych systemach. Sytuację komplikuje dodatkowo fakt, iż najczęściej stosowana w praktyce przestrzeń $L^*a^*b^*$ nie uwzględnia w pełni warunków obserwacji obrazu.

Podsumowanie

W wykładzie przedstawione zostały różne modele barw wykorzystywane w grafice komputerowej. Pokazany został również problem pojawiający się przy wyświetlaniu bądź reprodukcji obrazów barwnych na różnych urządzeniach. Wyjaśniono także do czego służą systemy zarządzania kolorem.

Przykładowe pytania i problemy do rozwiązania

1. Proszę uzasadnić stwierdzenie: "model RGB wykorzystywany w grafice komputerowej ma budowę dyskretną".
2. Proszę podać ile różnych barw mamy do dyspozycji jeżeli piksel jest reprezentowany za pomocą słów odpowiednio 8-bitowych, 16-bitowych i 24-bitowych.
3. Proszę omówić wykres chromatyczności.
4. Proszę wyjaśnić przeznaczenie systemów zarządzania kolorami

Wykład 6: Algorytmy techniki rastrowej

Streszczenie

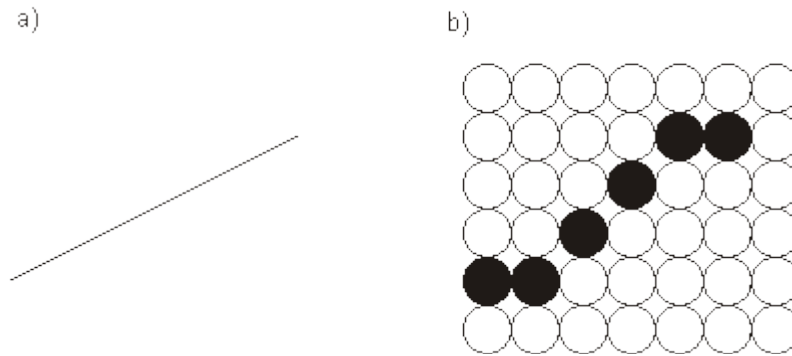
W poprzednich wykładach wyjaśniliśmy, że we współczesnych rozwiązaniach stosowanych w komputerach obraz jest tworzony na bazie prostokątnego rastra. Stąd powstaje problem opracowania algorytmów, które pozwolą tworzyć obrazy przy założeniu, że do dyspozycji jest tylko skończony zbiór punktów - pikseli.

W tym wykładzie poznamy kilka algorytmów umożliwiających tworzenia najprostszyc obiektów, takich jak odcinki czy okręgi. Następnie zwrócimy uwagę na jeden z zasadniczych problemów grafiki rastrowej a mianowicie na problem aliasingu. Z kolei omówimy algorytmy wypełniania figur i zasygnalizujemy problem obcinania. Dalej omówimy podstawowe przekształcenia geometryczne wykorzystywane w grafice 2D. Po części algorytmicznej, w drugiej części rozdziału omówimy wybrane zagadnienia związane z praktyką korzystania z gotowego oprogramowania 2D.

1. Odcinki

Na początku zajmiemy się problemem tworzenia odcinka na zasadzie składania go z punktów dostępnych w rastrze.

Zacznijmy od przedstawienia problemu. Zadanie polega na narysowaniu odcinka o znanych współrzędnych początku i końca. Na rysunku VI.1 a pokazano wygląd odcinka do jakiego jesteśmy przyzwyczajeni i jaki możemy uzyskać na papierze, korzystając z ołówka i linijki. Na rysunku VI.1 b pokazano wygląd tego samego odcinka narysowanego na bazie rastra. Oczywiście w tym przykładzie specjalnie została wybrana bardzo mała rozdzielczość rastra, po to żeby dobrze pokazać problem. Ponadto, wybór pikseli wchodzących w skład odcinka był dosyć arbitralny.

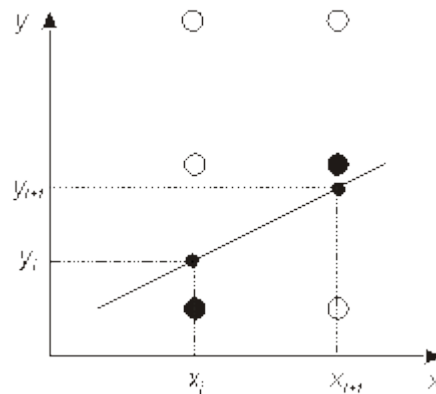


Rys. VI.1. Idealny wygląd odcinka (a); odcinek narysowany na rastrze (b)

Zastanówmy się wobec tego nad algorytmem, który umożliwi wybieranie pikseli dla odcinka o zadanych współrzędnych początku i końca. Przyjmijmy, że z naszym rastrem związany jest układ współrzędnych x , y . Wiadomo, że jeżeli znane są współrzędne końców odcinka, to można znaleźć równanie prostej, na której ten odcinek leży. Pamiętamy, że równanie takie ma postać

$$y = mx + b$$

gdzie m jest współczynnikiem określającym nachylenie prostej i $m = \operatorname{tg} \alpha$ a a jest kątem nachylenia prostej. Znając równanie prostej możemy dalej postępować następująco. Dla kolejnej kolumny rastra o współrzędnej x_i można wyznaczyć z równania prostej wartość współrzędnej y_i punktu leżącego na prostej. Następnie można znaleźć najbliższy punkt rastra w kolumnie x_i . Wyjaśnia to rysunek VI.2. Procedurę należy powtarzać dla każdej kolumny w zakresie zmian współrzędnej x dla odcinka.



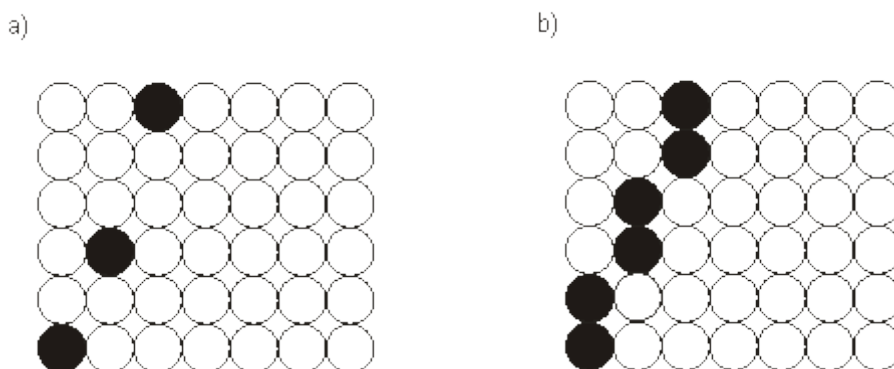
Rys. VI.2. Wyznaczanie kolejnych pikseli przy korzystaniu z równania linii

Metoda jest prosta. Jednak dla wyznaczenia kolejnego piksela trzeba wykonać operację mnożenia, dodawania i zaokrąglenia. Liczbę wykonywanych operacji można zmniejszyć jeżeli zauważy się, że przy stałym nachyleniu linii i stałym odstępem między kolumnami pikseli równym 1 wartość y_{i+1} w kolejnej kolumnie x_{i+1} można obliczyć dodając do wartości y_i stały przyrost $\Delta y = m$. Korzystając z zależności:

$$y_{i+1} = y_i + m$$

dla obliczenia kolejnego piksela trzeba wykonać jedno dodawanie i zaokrąglenie. Oszczędzamy wobec tego jedno mnożenie. Jeżeli weźmiemy pod uwagę, że przy tworzeniu jednego obrazu trzeba często narysować kilkadziesiąt odcinków i dla każdego z nich trzeba obliczyć kilkadziesiąt pikseli, to okaże się, że oszczędność w sensie liczby wykonywanych obliczeń staje się znacząca. A pamiętamy, że czas jaki mamy na obliczenie kolejnego obrazu jest zawsze bardzo krótki. Algorytm w przedstawionej wersji jest określany jako algorytm DDA.

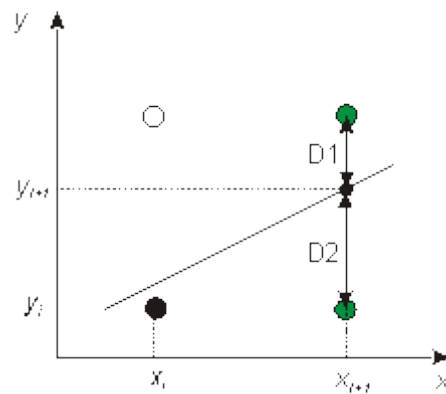
Zauważmy, że opisany sposób postępowania jest słuszny przy założeniu, że nachylenie odcinka jest mniejsze od 45° . Przy kątach większych od 45° i mniejszych od 90° trzeba wprowadzić modyfikację polegającą na tym, że teraz wyznaczone będą piksele w kolejnych wierszach, a nie w kolejnych kolumnach tak jak poprzednio. Gdyby tej zmiany nie wprowadzić liczba pikseli tworzących odcinek mogłaby być zbyt mała. Ilustruje to rysunek VI.3.



Rys. VI.3. Odcinek o nachyleniu większym od 45° . a) Zestaw pikseli przy wybieraniu według kolumn - wariant niepoprawny. b) Zestaw pikseli przy wybieraniu według wierszy - wariant poprawny

Okazuje się, że można jeszcze zmniejszyć czas obliczeń przy rysowaniu odcinka dodając wymóg, by wykonywane były tylko obliczenia stałopozycyjne, a nie zmiennopozycyjne jak to ma miejsce w algorytmie DDA. Odpowiedni algorytm został opracowany przez Bresenhama.

Zanim podamy pełny algorytm Bresenhama spróbujmy zobaczyć jak wyglądało rozumowanie, które doprowadziło do końcowej postaci algorytmu. Zakładamy, że nachylenie odcinka jest mniejsze od 45° (czyli $0 < m < 1$). Załóżmy, że został już znaleziony piksel w kolumnie x_i i mamy znaleźć piksel w następnej kolumnie x_{i+1} . Można zauważyć, że wybór ogranicza się tylko do dwóch pikseli: tego leżącego w tym samym wierszu i tego w wierszu powyżej (zobacz rysunek VI.4.). Pozostaje więc znalezienie prostego kryterium, które pozwoli wybrać piksel przy jak najmniejszym nakładzie obliczeniowym. Bresenham zaproponował, żeby takie kryterium skonstruować wykorzystując różnicę odległości ($D_1 - D_2$) rozważanych pikseli od punktu leżącego na rzeczywistym odcinku (por. rysunek VI.4). Można zauważyć, że badając znak tej różnicy można wybrać piksel. Przy znaku minus będzie to górny piksel, przy znaku plus dolny piksel.



Rys. VI.4. Ilustracja toku rozumowania Bresenhama. Na zielono zaznaczono piksele między którymi należy dokonać wyboru w kolejnej kolumnie

Ostatecznie algorytm podany przez Bresenhama ma następującą postać.

1. Znając współrzędne końców odcinka należy wybrać koniec odcinka o mniejszej współrzędnej x . (Ponieważ algorytm nie zawsze daje ten sam wynik przy rozpoczynaniu od różnych końców odcinka, przyjęto za zasadę, że zawsze zaczyna się od tego końca, dla którego współrzędna x jest mniejsza). Wybrany punkt (x_0, y_0) jest pierwszym punktem rysowanego odcinka.

2. Obliczyć pomocnicze wielkości:

$$\Delta x = x_2 - x_1, \Delta y = y_2 - y_1, a = 2\Delta y, b = 2\Delta y - 2\Delta x$$

oraz wartość początkową pomocniczego parametru decyzyjnego jako

$$p_0 = 2\Delta y - \Delta x.$$

3. Dla kolejnych kolumn o współrzędnych x_k , zaczynając od $k = 0$ należy sprawdzić znak wartości pomocniczego parametru p_k .

W przypadku gdy $p_k < 0$ następny piksel ma współrzędne (x_{k+1}, y_k) i nowa wartość parametru decyzyjnego jest określana z zależności $p_{k+1} = p_k + a$.

W przeciwnym przypadku następny punkt ma współrzędne (x_{k+1}, y_{k+1}) i nowa wartość parametru decyzyjnego jest określana z zależności $p_{k+1} = p_k + b$.

4. Krok 3 jest powtarzany dopóki nie dojdziemy do końca odcinka.

Zauważmy, że w każdym kroku, w celu wyznaczenia kolejnego piksela musimy wykonać tylko jedno dodawanie całkowitoliczbowe.

Przykład

Korzystając z metody Bresenhama znaleźć kolejne piksele dla odcinka o współrzędnych końców (2,2), (8,5).

1. Wybieramy pierwszy punkt odcinka. Jest to punkt (2,2).
2. Obliczamy pomocnicze wielkości

$$\Delta x = 8 - 2 = 6 \quad \Delta y = 5 - 2 = 3 \quad a = 2 \cdot 3 = 6 \quad b = 6 - 12 = -6$$

$$p_0 = 2\Delta y - \Delta x = 6 - 6 = 0$$

3. Wyznaczamy kolejne piksele

Ponieważ $p_0 = 0$, to następny piksel ma współrzędne (3,3) i

$$p_1 = p_0 + b = 0 - 6 = -6$$

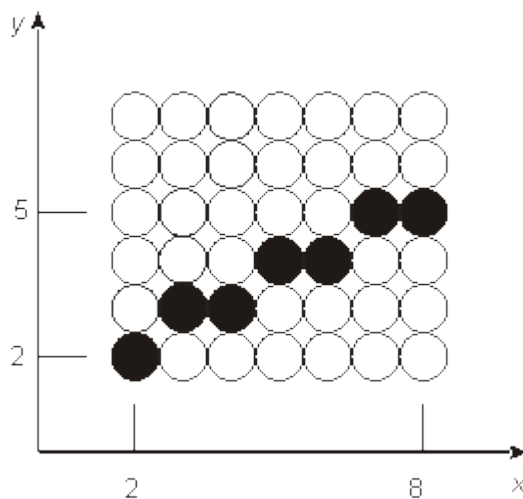
Ponieważ $p_1 < 0$, to następny piksel ma współrzędne (4,3) i

$$p_2 = p_1 + a = -6 + 6 = 0$$

Ponieważ $p_2 = 0$, to następny piksel ma współrzędne (5,4) i

$$p_3 = p_2 + b = 0 - 6 = -6$$

itd. Cały odcinek pokazano na rysunku VI.5.

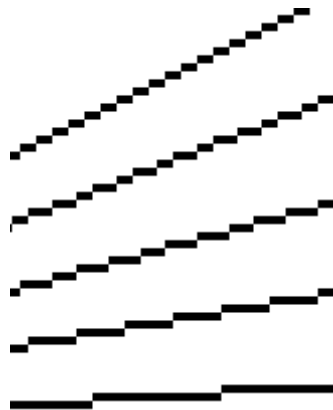


Rys. VI.5. Przykład działania algorytmu Bresenhama

2. Aliasing

Wygląd odcinka narysowanego na rastrze z pewnością nie jest najlepszy. Przy niezbyt dużej rozdzielczości rastra wyraźnie widać, że kształt odcinka odbiega od idealnego. Sytuacja staje się jeszcze mniej korzystna jeżeli odcinek będzie obracał się względem jakiegoś punktu. W trakcie zmieniania nachylenia odcinka zmienia się

wzór ułożenia pikseli. Na rysunku VI.6 pokazano kilka przykładów odcinków o różnych nachyleniach.

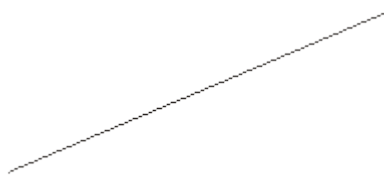


Rys. VI.6. Przykłady wyglądu odcinka przy różnych kątach nachylenia

Opisywany efekt określany jest mianem aliasingu. Jest on skutkiem skończonej rozdzielczości rastra z jakim mamy do czynienia. Efekt ten nie może być usunięty żadną metodą. Można jednak próbować go zmniejszać. Metody służące temu są określane jako metody antyaliasingowe.

Narzucającym się rozwiązaniem jest zwiększanie rozdzielczości. Są jednak z tym związane ograniczenia zarówno technologiczne jak i ekonomiczne. Niemniej tendencja do zwiększania rozdzielczości utrzymuje się od lat i obecnie efekty aliasingu są coraz mniej zauważalne, zwłaszcza jeżeli dodatkowo są stosowane metody algorytmiczne. Jednak mimo, że często nie zauważamy efektu aliasingu to on istnieje i można się o tym przekonać powiększając obraz widziany na ekranie monitora, na przykład w czasie projekcji na dużym ekranie ściennym.

a)



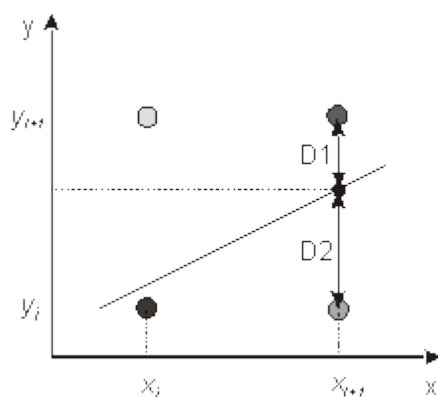
b)



Rys. VI.7. a) Odcinek i b) fragment odcinka w powiększeniu

Na rysunku VI.7 pokazano z lewej strony odcinek tak jak go widać na ekranie monitora a z prawej strony powiększony fragment odcinka, na którym widać strukturę pikselową odcinka. Przyglądając się powiększonej wersji odcinka można zauważyć, że poszczególne piksele mają różne odcienie szarości. Jest to skutek zastosowania metody antyaliasingowej. W tym przypadku zastosowana została metoda, w której w każdej kolumnie wybiera się dwa piksele. Są to dwa piksele leżące najbliżej punktu na idealnym odcinku. Zależnie od odległości pikseli od punktu na odcinku określa się odcienie szarości tych pikseli - im piksel jest bliżej

punktu na odcinku tym jest ciemniejszy. Suma odcieni szarości obu pikseli jest stała. W szczególnym przypadku, gdy jeden piksel leży dostatecznie blisko punktu na odcinku w kolumnie jest wyświetlany jeden piksel. Omawiany sposób zilustrowano na rysunku VI.8.



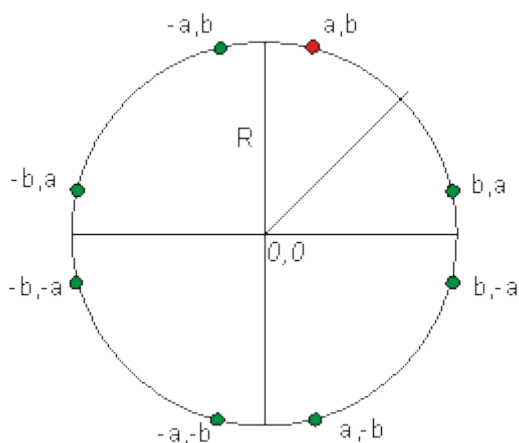
Rys. VI.8. Ilustracja metody zmniejszania efektu aliasingu, w której w każdej kolumnie są wyświetlane dwa piksele

3. Okrąg

Zastanówmy się teraz jak można narysować okrąg o środku w początku układu współrzędnych i o promieniu r . Oczywiście można skorzystać z równania okręgu

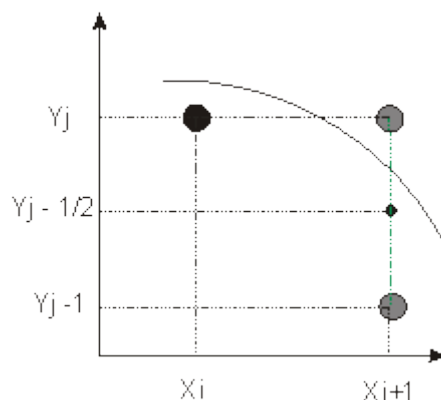
$x^2 + y^2 = r^2$ i dla kolejnych wartości x_i wyznaczać w każdej kolumnie wartości y i po zaokrągleniu znajdować odpowiednie piksele. Metoda taka jest jednak kosztowna obliczeniowo (występuje między innymi podnoszenie do kwadratu oraz pierwiastkowanie). Ponadto rozmieszczenie pikseli przybliżających okrąg nie jest równomierne wzdłuż okręgu (proszę się zastanowić dlaczego). Stąd w praktyce są stosowane inne metody. Niżej przedstawiono metodę z punktem środkowym (metoda mid point).

Przede wszystkim można zauważyć, że przy wyznaczaniu pikseli przybliżających okrąg można wykorzystać właściwość symetrii okręgu. Pozwala to na ograniczenie obliczania pikseli tylko do jednego oktantu (jednej ósmej) okręgu a pozostałe piksele znajdować zgodnie z zasadą pokazaną na rysunku VI.9.



Rys. VI.9. Po wyznaczeniu punktu zaznaczonego na czerwono, punkty zaznaczone na zielono są wyznaczane przy wykorzystaniu właściwości symetrii okręgu

Wyjaśnijmy teraz na czym polega pomysł metody z punktem środkowym. Przede wszystkim przypomnijmy, że dla każdego punktu leżącego na okręgu spełnione jest równanie $x^2 + y^2 - r^2 = 0$. Dla punktu leżącego na zewnątrz okręgu spełniona jest nierówność $x^2 + y^2 - r^2 > 0$, a dla punktu leżącego wewnątrz okręgu spełniona jest nierówność $x^2 + y^2 - r^2 < 0$. Z kolei, jeżeli ograniczymy się do pierwszego oktantu okręgu, to po wyznaczeniu piksela dla kolumny x_i , w następnej kolumnie x_{i+1} możemy ograniczyć się tylko do wyboru między dwoma pikselami: pikselem leżącym w tym samym wierszu i pikselem leżącym w wierszu poniżej (por. rysunek VI.10). Wyboru między tymi dwoma pikselami można dokonać analizując położenie względem okręgu punktu znajdującego się w połowie odcinka łączącego dwa rozważane piksele, tak zwanego punktu środkowego. Zależnie od tego czy punkt środkowy leży wewnątrz okręgu czy na zewnątrz okręgu, należy wybrać odpowiednio górny albo dolny piksel. Okazało się, że w przypadku problemu rysowania okręgu, podobnie jak w przypadku rysowania odcinka metodą Bresenhama, można znaleźć kryterium wyboru piksela w kolejnej kolumnie, wymagające wykonania jedynie kilku prostych operacji.



Rys. VI.10. Ilustracja metody z punktem środkowym

Ostatecznie algorytm rysowania okręgu o środku w początku układu współrzędnych i o promieniu r metodą z punktem środkowym wygląda następująco.

1. Punkt początkowy ma współrzędne $(0, r)$. Natomiast początkowa wartość

pomocniczego parametru decyzyjnego wynosi
$$p_0 = \frac{5}{4} - r$$

2. Dla kolejnych kolumn o współrzędnych x_k , zaczynając od $k = 0$, należy sprawdzić znak wartości pomocniczego parametru p_k .

W przypadku gdy $p_k < 0$, następny piksel ma współrzędne (x_{k+1}, y_k) i nowa wartość parametru decyzyjnego jest określana z zależności

$$p_{k+1} = p_k + 2x_{k+1} + 1.$$

W przeciwnym przypadku, następny punkt ma współrzędne (x_{k+1}, y_{k-1}) i nowa wartość parametru decyzyjnego jest określana z zależności

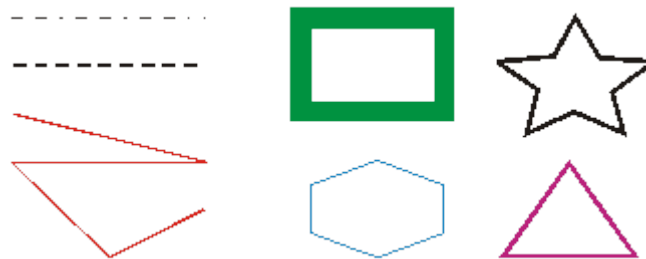
$$p_{k+1} = p_k + 2x_{k+1} + 1 - 2y_{k+1}.$$

3. Wyznaczyć siedem pikseli o współrzędnych wynikających z właściwości symetrii okręgu.
4. Powtarzać kroki 2 i 3 do czasu gdy spełniony będzie warunek $x = y$.

Proponuję samodzielne wyznaczenie pikseli dla okręgu o środku w początku układu współrzędnych i o promieniu $r = 10$.

4. Inne krzywe i figury

Mając do dyspozycji algorytm rysowania odcinka można rozwiązać problem rysowania linii łamanych i wielokątów. Na rysunku VI.11 pokazano przykłady takich obiektów. Zauważmy, że liniom i konturom figur można przypisywać różne atrybuty, takie jak styl (linia ciągła, przerywana, kropkowana itp.), grubość czy kolor.



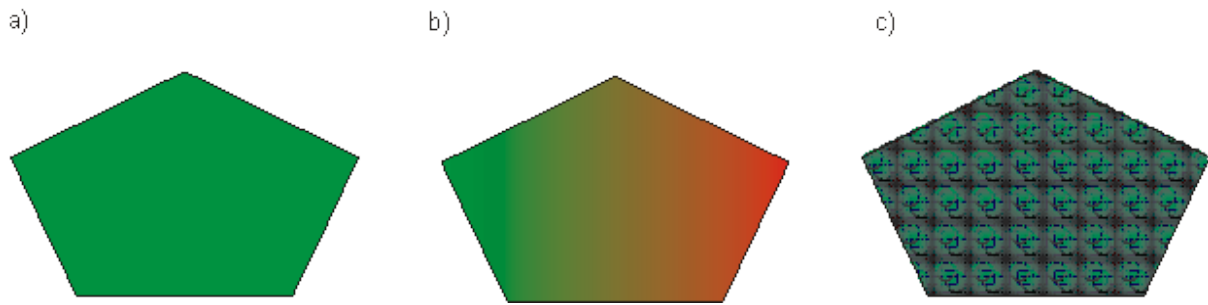
Rys. VI.11. Przykłady obiektów budowanych z odcinków, którym przypisano różne atrybuty

Tak proste obliczeniowo algorytmy jak dla odcinka i okręgu istnieją dla niewielu innych obiektów (na przykład dla elipsy). W przypadku innych krzywych opisanych równaniami na ogół konieczne jest korzystanie z równania krzywej i wyznaczania wartości współrzędnej y dla kolejnych wartości współrzędnej x i zaokrąglania do najbliższego piksela. W niektórych przypadkach można korzystać z właściwości symetrii (na przykład dla krzywej sinusoidalnej).

Często jednak nie znamy równania krzywej, która jest nam potrzebna. Wtedy pozostaje interakcyjne narysowanie takiej krzywej. Nie zawsze jest to wygodny sposób. W takich przypadkach można skorzystać z krzywych wielomianowych specjalnego typu, na przykład z krzywych Béziera (por. pkt VII.5).

5. Wypełnianie

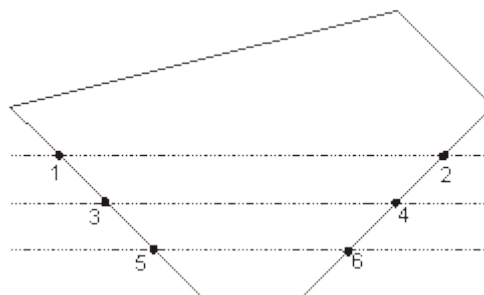
Ważną operacją w grafice 2D jest wypełnianie wnętrza figur. Możliwe są różne warianty wypełniania. Wnętrze może być wypełnione jednolitą barwą, zmienną barwą (wypełnienie tonalne) albo teksturą. Przykłady wypełniania pokazano na rysunku VI.12.



Rys. VI.12. Przykłady wypełniania wnętrza figury: a) ciągłe, b) tonalne, c) teksturą

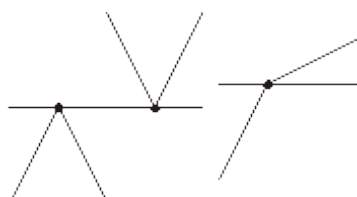
Spróbujmy teraz zastanowić się jak może wyglądać algorytm wypełniania jednolitą barwą. Zakładamy, że znany jest kontur figury i że jest on zamknięty. Wśród stosowanych metod można wyróżnić dwie grupy: metody przeglądania wierszami oraz metody z punktem początkowym.

W metodach przeglądania wierszami występują dwie fazy. W pierwszej fazie, dla każdego wiersza rastra znajduje się przecięcia z konturami obiektów i ustala się parę punktów przecięcia, między którymi należy dokonać wypełnienia. W drugiej fazie ma miejsce wypełnianie odcinków wyznaczonych w pierwszej fazie. Na rysunku VI.13 pokazano przykładowe linie rastra przecinające wielokąt i zaznaczono punkty przecięć z krawędziami wielokąta. W drugiej fazie zostaną wypełnione odcinki ograniczone przez parę punktów $P_1(1,2)$, $P_2(3,4)$, $P_3(5,6)$.



Rys. VI.13. Wyznaczanie odcinków należących do wnętrza wielokąta, które w drugiej fazie algorytmu przeglądania wierszami zostaną wypełnione

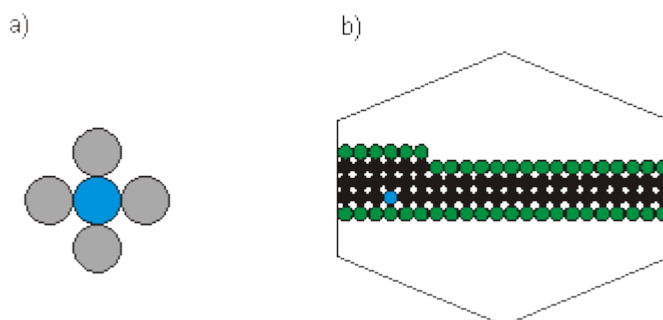
Algorytm koncepcyjnie jest prosty i nie stwarza istotnych problemów implementacyjnych w przypadku wielokątów wypukłych, zwłaszcza trójkątów. Przy konkretnej implementacji, dla ustalonej klasy obiektów, trzeba zwracać uwagę na wierzchołki. Na ogół konieczne jest wyróżnienie dwóch klas wierzchołków: takich, dla których dwie stykające się w wierzchołku krawędzie leżą po jednej stronie linii rastra i takie, dla których krawędzie leżą po obu stronach linii rastra (por. rysunek VI.14).



Rys. VI.14. Dwa rodzaje wierzchołków

W grupie metod z punktem początkowym wypełnianie zaczyna się od znanego punktu należącego do wnętrza figury. W jednym z wariantów metody, określanym jako metoda z czterema sąsiadami, w każdym kroku próbuje się wypełnić cztery piksele sąsiednie w stosunku do rozpatrywanego piksela (por. rysunek VI.15 a). Kolejność analizowanych pikseli może być różna, z tym, że zawsze rozpatrywany piksel należy do listy aktywnych pikseli - tych, które leżą na obrzeżu już wypełnionego obszaru. Rozważa się oczywiście wyłącznie punkty należące do wnętrza figury.

Przykładowa kolejność postępowania może być następująca. Od punktu początkowego poruszamy się w lewo wzdłuż wiersza rastra aż do napotkania brzegu obszaru. Następnie w tym samym wierszu poruszamy się w prawo od punktu początkowego aż do napotkania brzegu obszaru. Z kolei przechodzimy do sąsiedniego wiersza i powtarzamy procedurę, zaczynając od punktu leżącego nad punktem początkowym. Powtarzamy to dopóki jest to możliwe. Następnie przechodzimy do wiersza poniżej wiersza, w którym leży punkt początkowy i postępujemy podobnie jak poprzednio. Następnie sprawdzamy stan listy aktywnych pikseli (lista ta jest przez cały czas aktualizowana) i próbujemy kontynuować procedurę od pierwszego piksela znalezionej na liście aktywnych pikseli itd. Na rysunku VI.15 zilustrowano opisany sposób postępowania, pokazując pewien pośredni stan realizacji wypełniania. Na niebiesko zaznaczono punkt początkowy. Zielonym kolorem wyróżniono piksele znajdujące się na liście aktywnych pikseli.



Rys. VI.15. Ilustracja metody wypełniania z punktem początkowym. a) Cztery punkty sąsiednie w stosunku do piksela. b) Wypełnianie od punktu zaznaczonego na niebiesko - etap pośredni

Zauważmy, że algorytm może wypełniać figury zarówno wypukłe jak i niewypukłe.

Podsumowanie

Po zapoznaniu się z wykładem, można się zorientować na czym polega rysowanie na bazie rastra. Z pewnością jest to zupełnie coś innego niż rysowanie za pomocą ołówka i linijki czy cyrkla. Warto również zwrócić uwagę na fakt optymalizacji algorytmów z punktu widzenia szybkości obliczeń. Problem ten jest niestety istotny w grafice komputerowej. Konieczność wyznaczenia barwy każdego z

wieluset tysięcy pikseli składających się na obraz, i to w ograniczonym czasie, zmusza do opracowywania takich algorytmów i sprzętu, które umożliwiają tworzenie obrazów o wymaganej jakości i w rozsądnym czasie. W wykładzie zwrócono również uwagę na bardzo istotny problem aliasingu i na metody zmniejszania jego efektów. Omówiono również algorytmy wypełniania konturów zamkniętych.

Przykładowe pytania i problemy do rozwiązania

1. Korzystając z algorytmu Bresenhama naszkicować na kratkowanym papierze odcinek o współrzędnych wierzchołków (2,3), (6,7) oraz odcinek o współrzędnych wierzchołków (2,3), (4,8).
2. Znaleźć czwarty piksel dla okręgu o środku w początku układu współrzędnych i promieniu $r = 8$.
3. Wyjaśnić skąd bierze się problem aliasingu i czy można jego efekty całkowicie wyeliminować.
4. Naszkicować na papierze dowolny kontur zamknięty niewypukły i wskazać jakiś punkt należący do wnętrza konturu. Proszę zastanowić się w jakiej kolejności może odbywać się wypełnianie wnętrza konturu przy wykorzystaniu metody z punktem początkowym.

Wykład 7: Algorytmy techniki wektorowej 2D

Streszczenie

W ramach wykładu poznamy kilka algorytmów stosowanych w odniesieniu do obiektów. Będą to przede wszystkim przekształcenia geometryczne, które pozwalają przesuwać obiekty, skalować je, obracać bądź pochylać. Dalej poznamy algorytmy obcinania, istotne między innymi z punktu widzenia optymalizacji czasu obliczeń. W końcowej części wykładu poznamy krzywe Béziera definiowane za pomocą punktów sterujących. Omawiane algorytmy dotyczą tak zwanej grafiki wektorowej, w której operuje się obiektami, w stosunku do których można wykonywać różne operacje.

1. Grafika wektorowa

W poprzednim wykładzie poznaliśmy kilka algorytmów, które umożliwiały rysowanie odcinków i figur w technice rastrowej z dokładnością pikselową. W praktyce przy tworzeniu obrazu na płaszczyźnie na ogół nie operuje się poszczególnymi pikselami, natomiast operuje się obiektami takimi jak odcinki, figury czy krzywe. Obiekty te opisywane są za pomocą wierzchołków bądź punktów sterujących i odpowiednich atrybutów. Problem końcowego rysowania figur jest odkładany do ostatniej fazy tworzenia obrazu, tak zwanej rasteryzacji, kiedy to wyznaczane są wszystkie piksele obrazu.

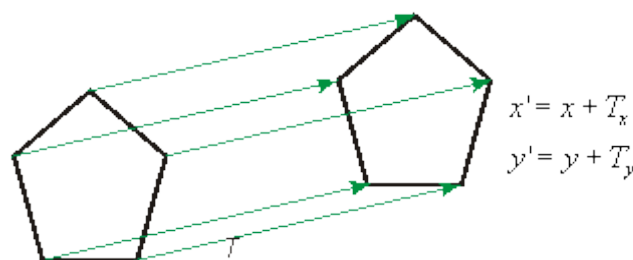
W przeciwieństwie do grafiki rastrowej, w której operuje się mapą pikselową obrazu, w grafice wektorowej operuje się obiektami definiowanymi za pomocą wierzchołków bądź punktów sterujących. Korzystanie z grafiki wektorowej ma kilka zalet. Po pierwsze operuje się na znacznie mniejszym zbiorze punktów niż w

przypadku gdybyśmy mieli do czynienia z mapą pikselową. Skutkiem tego jest znaczne skrócenie czasu niektórych obliczeń. Po drugie, opis wektorowy obiektów zajmuje znacznie mniej miejsca w pamięci oraz skraca czas ewentualnej transmisji obrazu. Po trzecie, przy opisie wektorowym każdy obiekt może być traktowany niezależnie, podczas gdy w przypadku mapy pikselowej mamy do czynienia z całym obrazem - poszczególne obiekty tracą swój indywidualny charakter i przestają być niezależnymi bytami. Po czwarte opis wektorowy jest łatwo skalowalny. Oznacza to, że chcąc na przykład zwiększyć wymiary obiektu, wystarczy wykonać operację skalowania jedynie w odniesieniu do punktów definiujących obiekty. Końcowa rasteryzacja będzie wykonana w odniesieniu do już przeskalowanego obiektu, a więc zawsze będzie wykonana z rozdzielczością, z jaką obraz będzie reprodukowany. Zauważmy, że w przypadku powiększania obrazu rastrowego (mapy pikselowej) zwiększa się efekt aliasingu.

2. Przekształcenia geometryczne

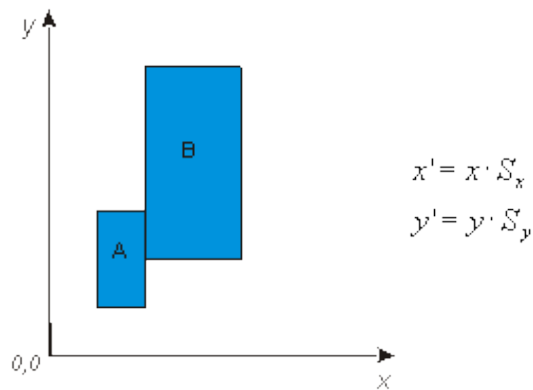
Obiekty zdefiniowane za pomocą wierzchołków bądź punktów sterujących, można poddawać różnego rodzaju przekształceniom (transformacjom). Tutaj ograniczymy się do kilku podstawowych przekształceń geometrycznych: przesuwania, skalowania, obrotów i pochylenia. Jak już powiedzieliśmy wcześniej, przekształcenia są realizowane w odniesieniu do poszczególnych wierzchołków figur bądź punktów sterujących. Dopiero po znalezieniu nowych położenia wierzchołków bądź punktów sterujących następuje rysowanie całej figury.

Przekształcenie polegające na przesunięciu (translacji) obiektu polega na zmianie położenia wszystkich wierzchołków o zadany wektor przesunięcia. Przykład takiego przekształcenia pokazano na rysunku VII.1. Na rysunku podano również równania wykorzystywane przy takim przekształceniu. Równanie te pozwalają znaleźć współrzędne (x',y') punktu po przesunięciu o wektor $T(T_x, T_y)$.



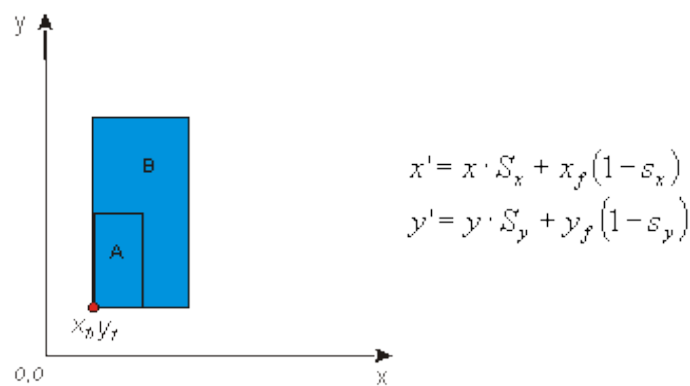
Rys. VII.1. Przesunięcie obiektu o wektor T

Skalowanie pozwala zmienić wielkość obiektu ze współczynnikiem skalowania S . Przykład skalowania pokazano na rysunku VII.2. Obok pokazano równania umożliwiające znalezienie nowego położenia punktu poddanego skalowaniu względem osi x ze współczynnikiem S_x i względem osi y ze współczynnikiem S_y .



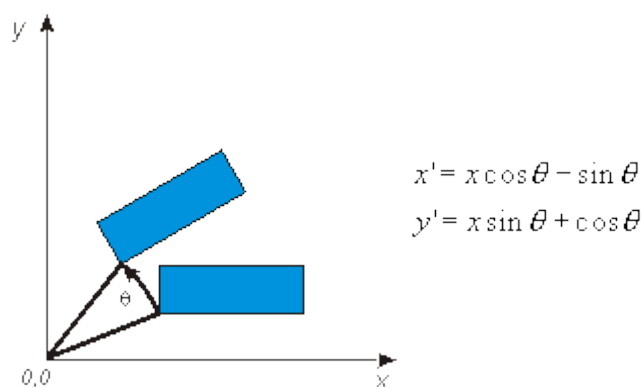
Rys. VII.2. Przykład skalowania obiektu ze współczynnikami skalowania $S_x = S_y = 2$. Punktem odniesienia jest początek układu współrzędnych

Zauważmy, że przy operacji skalowania istotny jest punkt odniesienia. Na rysunku VII.2 punktem odniesienia był początek układu współrzędnych. Natomiast na rysunku VII.3 pokazano przykład skalowania, w którym punkt odniesienia znajduje się w lewym dolnym rogu skalowanego obiektu.



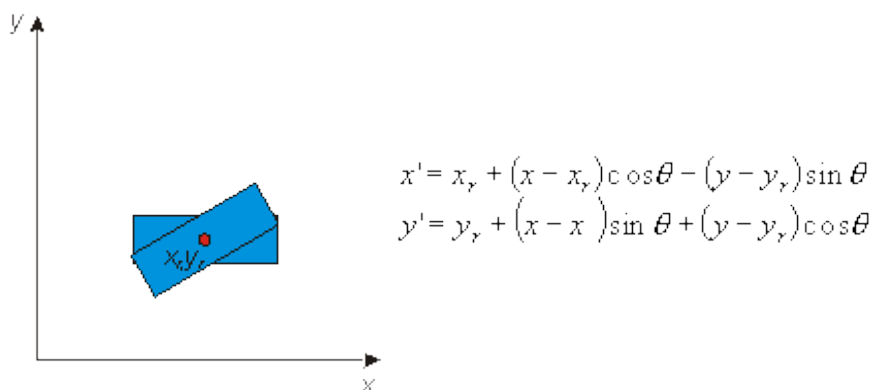
Rys. VII.3. Przykład skalowania obiektu ze współczynnikami skalowania $S_x = S_y = 2$. Punktem odniesienia jest punkt o współrzędnych $x_f = 1, y_f = 1$

Podobnie jak w przypadku skalowania, dla obrotu (rotacji) istotny jest punkt, wokół którego następuje obrót. Na rysunku VII.4 pokazano przykład obrotu wokół początku układu współrzędnych o kąt θ . Podano również równania umożliwiające obliczanie współrzędnych punktów po obrocie o zadany kąt θ względem początku układu współrzędnych.



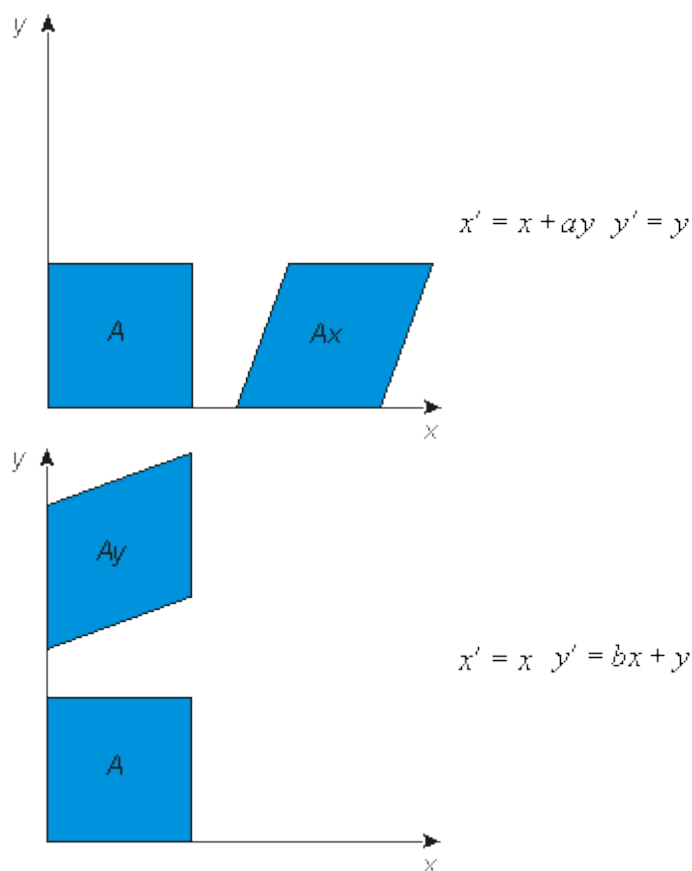
Rys. VII.4. Obrót obiektu o zadany kąt θ wokół początku układu współrzędnych

Z kolei na rysunku VII.5 pokazano przykład obrotu tej samej figury co poprzednio, o taki sam kąt, ale wokół punktu o współrzędnych x_r, y_r .



Rys. VII.5. Obrót obiektu o zadany kąt θ wokół punktu o współrzędnych x_r, y_r

Kolejne przydatne w praktyce przekształcenie to pochylenie wzdłuż osi x ze współczynnikiem a albo wzdłuż osi y ze współczynnikiem b . Oba rodzaje pochyłeń są pokazane na rysunku VII.6, wraz z odpowiednimi równaniami.



Rys. VII.6. Pochylenie obiektu wzdłuż osi x (Ax) i wzdłuż osi y (Ay)

3. Współrzędne jednorodne

W grafice komputerowej omówione wyżej przekształcenia geometryczne są opisywane z wykorzystaniem tak zwanych współrzędnych jednorodnych. Obiekt opisany w określonym układzie współrzędnych prostokątnych, na przykład x, y , może być przedstawiony również w układzie współrzędnych o jeden wymiar większym, na przykład w układzie x, y, z . Wtedy, na przykład w przypadku punktu o współrzędnych (x, y) dochodzi jedna współrzędna z , która w ogólnym przypadku może mieć dowolną wartość. Jeżeli przyjmiemy, że ta nowa współrzędna będzie miała wartość 1 to mówimy, że punkt jest reprezentowany we współrzędnych jednorodnych. I tak, punkt o współrzędnych (x, y) ma teraz współrzędne $(x, y, 1)$. Dodajmy od razu, że gdyby zdarzyło się, że w wyniku obliczeń we współrzędnych jednorodnych otrzymalibyśmy współrzędne (x', y', w) , to od razu trzeba wykonać krok normalizacyjny tak, żeby uzyskać postać standardową dla której $w = 1$. Ten krok normalizacyjny jest wykonywany zgodnie z następującymi zależnościami:

$$x = \frac{x'}{w}, \quad y = \frac{y'}{w}$$

Przejdźcie do współrzędnych jednorodnych umożliwia jednolity zapis podstawowych przekształceń geometrycznych: przesunięcia, obrotu, skalowania i innych za pomocą macierzy przekształceń 3×3 . Pozwala to realizować takie operacje za pomocą specjalizowanego sprzętu i w efekcie przyspieszanie obliczeń.

Korzystając ze współrzędnych jednorodnych można, w przypadku złożonych przekształceń, mnożyć macierze składowych przekształceń przez inne macierze przekształceń i uzyskiwać pojedynczą macierz dla złożonego przekształcenia (również o wymiarze 3×3). Pozwala to niejednokrotnie w istotny sposób zredukować ilość potrzebnych obliczeń.

W literaturze są stosowane dwie konwencje zapisu punktu w postaci wektorowej: kolumnowa i wierszowa. Obie konwencje są równoważne. W przypadku reprezentacji punktu w postaci wektora kolumnowego mnoży się macierz przekształcenia $M_{3 \times 3}$ przez wektor kolumnowy tak jak w poniższym przykładowym zapisie:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = M_{3 \times 3} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Przy reprezentacji punktu w postaci wektora wierszowego mnoży się ten wektor przez transponowaną macierz przekształcenia $M_{3 \times 3}^T$ tak jak w poniższym przykładzie:

$$[x' \quad y' \quad 1] = [x \quad y \quad 1] \cdot M_{3 \times 3}^T$$

Postać macierzy 3×3 w jednej konwencji jest postacią transponowaną macierzy w drugiej konwencji. W dalszym ciągu będzie używana konwencja z wektorem kolumnowym.

Podstawowe przekształcenia: przesunięcie, mnożenie, skalowanie, pochylenie i dowolne ich kombinacje są określane jako przekształcenia afiniczne i ich ogólny zapis jest następujący:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Niżej podano macierze dla poszczególnych przekształceń geometrycznych na płaszczyźnie.

Przesunięcie o wektor (t_x, t_y) :

$$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Skalowanie ze współczynnikami skalowania S_x, S_y :

$$\begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Obrót o kąt θ względem początku układu współrzędnych:

$$\begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Pochylenie wzdłuż osi x ze współczynnikiem a i wzdłuż osi y ze współczynnikiem b :

$$\begin{bmatrix} 1 & a & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ b & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Jak już wspomniano wyżej, jedną z zalet reprezentowania przekształceń w postaci macierzowej jest to, że można je mnożyć przez siebie w celu uzyskania macierzy dla przekształcenia złożonego. Jeżeli więc konieczne będzie wykonanie szeregu przekształceń w odniesieniu do określonego punktu P , to można wstępnie wykonać mnożenie macierzy M_i kolejnych przekształceń (w ustalonej kolejności) i uzyskać jedną macierz wypadkową M' dla tego ciągu przekształceń. Jest to szczególnie korzystne obliczeniowo wtedy, gdy ten sam ciąg przekształceń ma być wykonany w odniesieniu do wielu punktów. Wyjaśniają to poniższe równania:

$$P' = M_3 M_2 M_1 P$$

$$M = M_3 M_2 M_1$$

$$P' = M P.$$

Jeżeli na przykład chcielibyśmy obracać obiekt nie względem początku układu współrzędnych a względem pewnego punktu $P(x_1, x_2)$, to powinniśmy wykonać kolejno następujące operacje: takie przesunięcie obiektu, żeby punkt $P(x_1, x_2)$ znalazł się w początku układu współrzędnych, wykonanie obrotu o kąt θ , takie przesunięcie obróconego obiektu żeby punkt znajdujący się w początku układu współrzędnych wrócił do początkowego położenia $P(x_1, x_2)$. Oznacza to, że kolejno powinniśmy wykonać operacje przesunięcia o wektor $(-x_1, -x_2)$, obrotu o kąt θ i przesunięcia o wektor (x_1, x_2) . Można to zrealizować wykonując kolejno trzy operacje mnożenia wektora przez odpowiednie macierze. Równoważny sposób polega na tym, że na początku znajdziemy macierz wypadkową dla całej operacji, a dopiero potem korzystamy z niej w odniesieniu do poszczególnych wierzchołków obracanego obiektu. Niżej pokazany jest proces wyznaczania wypadkowej macierzy. Zwróćmy uwagę na kolejność macierzy.

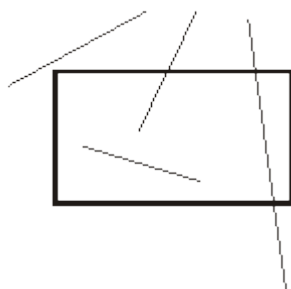
$$\begin{bmatrix} 1 & 0 & x_1 \\ 0 & 1 & y_1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_1 \\ 0 & 1 & -y_1 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & x_1(1 - \cos \theta) + y_1 \sin \theta \\ \sin \theta & \cos \theta & y_1(1 - \cos \theta) - x_1 \sin \theta \\ 0 & 0 & 0 \end{bmatrix}$$

W podobny sposób można wyznaczać macierze wypadkowe dla innych złożonych przekształceń. Proponuję samodzielne znalezienie macierzy wypadkowej dla operacji skalowania względem punktu nie leżącego w początku układu współrzędnych.

4. Obcinanie

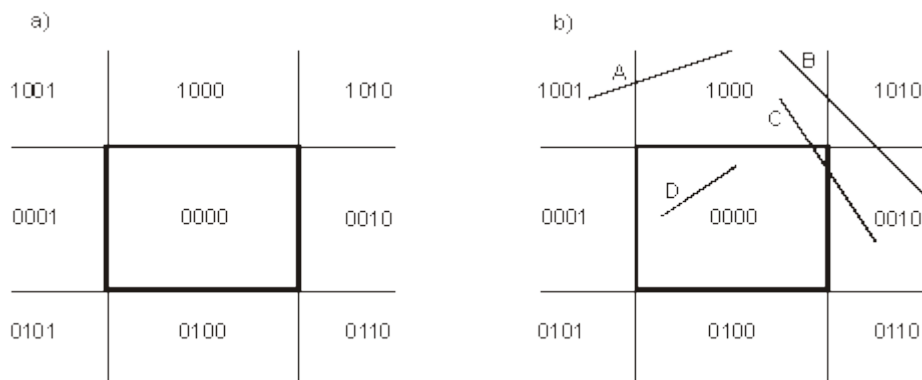
Operacja obcinania może być potrzebna wtedy, gdy zaprojektowany rysunek jest na tyle duży, że nie może być w całości wyświetlony na ekranie. W takiej sytuacji sensowne jest wybranie z rysunku tylko tego fragmentu, który może pojawić się na ekranie i poddać przetwarzaniu tylko te obiekty, które znajdują się w wybranym fragmencie (oknie). Znanych jest kilka różnych metod obcinania. Tutaj ograniczymy się do omówienia klasycznej metody obcinania odcinków Cohena-Sutherlanda oraz metody Sutherlanda-Hodgmana obcinania wielokątów.

Zacznijmy od problemu obcinania odcinków. Algorytm jest realizowany w dwóch fazach. W pierwszej fazie chodzi o to, żeby z procesu obcinania wyeliminować te odcinki, które z pewnością nie wymagają obcinania: albo leżą poza oknem albo całkowicie mieszczą się w oknie. Przykłady różnego usytuowania odcinków względem okna pokazuje rysunek VII.7. W drugiej fazie każdy odcinek z tych, które pozostaną po pierwszej fazie, jest niezależnie obcinany.



Rys. VII.7. Przykładowe położenia odcinków względem okna obcinania

Dla celów szybkiego sprawdzania, czy odcinek może być wyeliminowany z dalszej procedury obcinania Cohen i Sutherland zaproponowali, żeby podzielić płaszczyznę okna na dziewięć części uzyskanych w wyniku przedłużenia krawędzi okna. Każdej części został przypisany kod czterobitowy. Ilustruje to rysunek VII.8. Zasada przypisania kodów jest następująca. Wszystkie części leżące z lewej strony okna mają jedynkę na pierwszej pozycji kodu. Wszystkie części leżące z prawej strony okna mają jedynkę na drugiej pozycji kodu itd.



Rys. VII.8. a) Podział obszaru na części i sposób przypisania kodów; b) przykładowe analizowane odcinki

Decyzję co do możliwości wyeliminowania odcinka podejmuje się w następujący sposób. Każdemu końcowi odcinka przypisywany jest kod obszaru, w którym ten koniec znajduje się. Następnie znajduje się iloczyn logiczny tych kodów (wyznacza się iloczyn logiczny dla każdej pary odpowiadających sobie bitów niezależnie). Jeżeli iloczyn jest różny od zera to odcinek można odrzucić (na przykład odcinek A na rysunku) jako leżący poza oknem. Można również wyeliminować z dalszej procedury obcinania odcinki, których oba końce leżą w oknie (odcinek D na rysunku). Innych odcinków nie można wyeliminować, nawet mimo tego, że z pewnością leżą poza oknem, tak jak na przykład odcinek B na rysunku. Wynika to stąd, że tak prosta procedura nie może rozróżnić odcinków takich jak B i C na rysunku. Niemniej mimo, że skuteczność procedury nie jest stuprocentowa, jest ona stosowana ze względu na prostotę. Wszystkie odcinki, które nie zostaną wyeliminowane są poddawane dalszej procedurze obcinania.

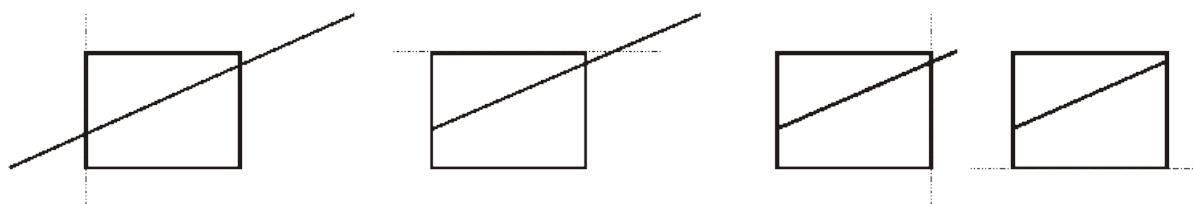
Przykład

Sprawdzić czy odcinki A i C z rysunku VII.8 można wyeliminować z dalszej procedury obcinania.

Końce odcinka A znajdują się w obszarach o kodach 1001 oraz 1000. Iloczyn tych kodów jest równy 1000, a więc jest różny od zera. Wobec tego odcinek może być wyeliminowany.

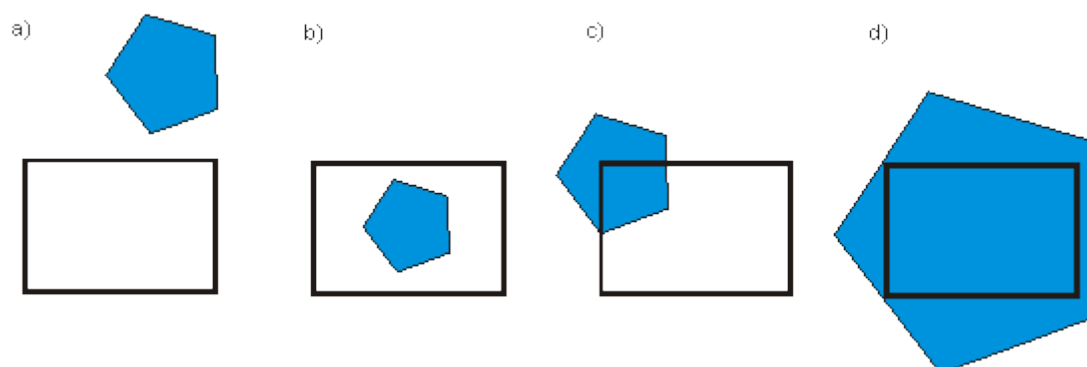
Końce odcinka C znajdują się w obszarach o kodach 1000 oraz 0010. Iloczyn tych kodów jest równy 0000, a więc jest równy zeru. Wobec tego odcinek nie może być wyeliminowany.

Procedura obcinania składa się z czterech kroków. W każdym z nich dokonuje się obcięcia przez jedną krawędź okna, zawsze w ustalonej kolejności. W kroku obcinania przez wybraną krawędź znajduje się przecięcie odcinka z krawędzią i usuwa się tę część odcinka, która leży na zewnątrz okna. Procedurę ilustruje rysunek VII.9.



Rys. VII.9. Procedura obcinania odcinka przez kolejne krawędzie okna

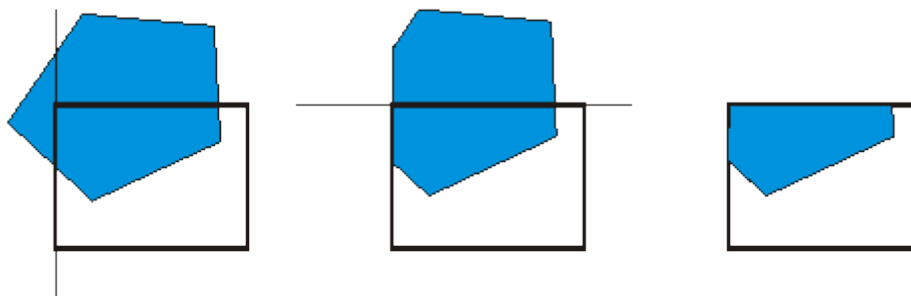
Procedurę obcinania wielokątów wypukłych można zrealizować następująco. Najpierw warto sprawdzić relację położenia wielokąta i okna. Możliwe przypadki są pokazane na rysunku VII.10.



Rys. VII.10. Możliwe relacje położenia wielokąta i okna. a) Obiekty są rozłączne, b) wielokąt leży wewnątrz okna, c) wielokąt przecina się z oknem, d) okno zawiera się w wielokącie

W trzech przypadkach sposób postępowania jest oczywisty. Jedynie w przypadku c) konieczne jest kontynuowanie procedury. Można ją zrealizować w czterech krokach. W każdym kroku można dokonywać obcinania przez kolejną krawędź okna (a ściślej przez linię, na której leży krawędź). Za każdym razem odrzucana jest

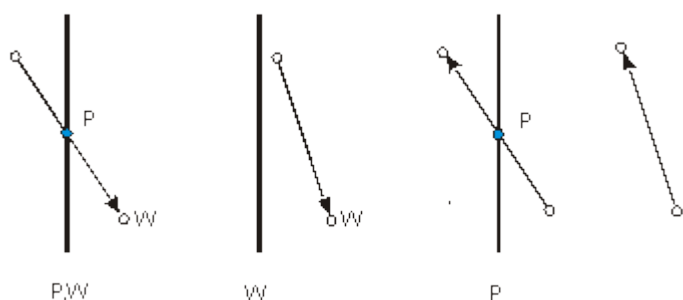
część wielokąta leżąca po zewnętrznej stronie okna. Ilustruje to rysunek VII.11. W tym przypadku wystarczyłoby obcięcie przez dwie krawędzie okna.



Rys. VII.11. Przykład obcinania wielokąta

W procedurze, w najprostszym przypadku, można bezpośrednio wykorzystać algorytm obcinania odcinków w odniesieniu do poszczególnych krawędzi obcinanego wielokąta. Wygodniejsze może się jednak okazać skorzystanie z pomysłu zaproponowanego przez Sutherlanda i Hodgmana.

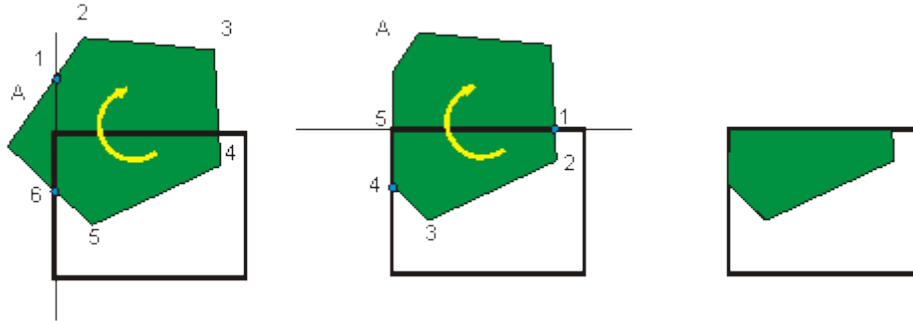
Założmy, że będziemy przeglądali krawędzie wielokąta w kierunku przeciwnym do ruchu wskazówek zegara. Założmy też, że dla każdej linii, na której leży krawędź okna wyróżniamy stronę zewnętrzną i wewnętrzną - po stronie wewnętrznej leży okno. Istotę pomysłu ilustruje rysunek VII.12. Wyróżniono na nim cztery sytuacje jakie mogą wystąpić przy analizie relacji krawędzi wielokąta z rozpatrywaną krawędzią okna (linią). Krawędź wielokąta może przechodzić ze strony zewnętrznej na stronę wewnętrzną rozpatrywanej krawędzi okna. Wtedy znajdujemy punkt przecięcia P i na pomocniczą listę wpisujemy ten punkt przecięcia P oraz docelowy koniec krawędzi wielokąta W . Jeżeli krawędź wielokąta leży całkowicie po wewnętrznej stronie krawędzi okna, to na pomocniczą listę wpisujemy docelowy wierzchołek krawędzi wielokąta W . Jeżeli krawędź wielokąta przechodzi na stronę zewnętrzną krawędzi okna, to na pomocniczą listę wpisujemy punkt przecięcia z krawędzią okna. Jeżeli krawędź wielokąta leży w całości po stronie zewnętrznej, to nic nie wprowadzamy na pomocniczą listę.



Rys. VII.12. Cztery przypadki usytuowania krawędzi wielokąta względem krawędzi okna. Dla każdego przypadku zaznaczono informacje jakie należy wprowadzić na pomocniczą listę

Na rysunku VII.13 pokazano poprzedni przykład obcinania wielokąta przez okno z uwzględnieniem przedstawionych reguł. Dla uproszczenia zamiast tworzyć pomocniczą listę bezpośrednio na rysunku zaznaczano odpowiednie punkty,

przypisując im numery w kolejności pojawiania się. W każdym kroku analizę zaczynano od krawędzi oznaczonej literą A. Każdorazowo, po ustaleniu pomocniczej listy, łączy się punkty w kolejności pojawiania się na liście: od pierwszego do ostatniego i do pierwszego. Zewnętrzną część wielokąta usuwa się.



Rys. VII.13. Kolejne kroki algorytmu obcinania

Proponuję samodzielnie wykonać algorytm obcinania dla wielokąta pokazanego na rysunku VII.14.

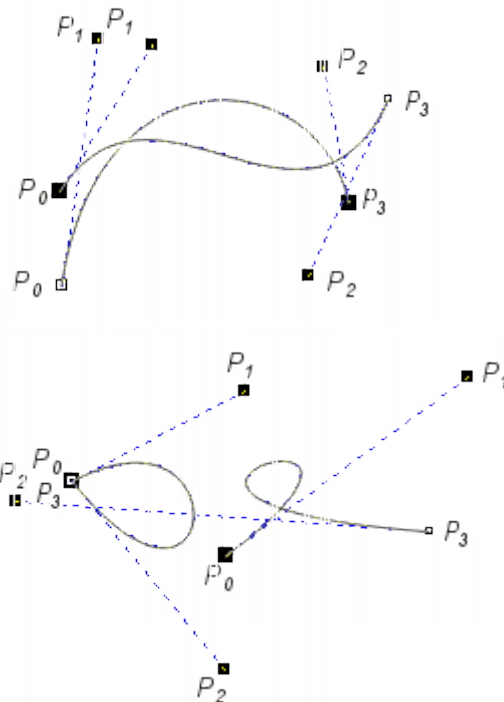


Rys. VII.14. Przykład do samodzielnego wykonania obcinania

5. Krzywe Béziera

Krzywe Béziera są krzywymi wielomianowymi, których kształt można opisywać za pomocą tak zwanych punktów sterujących (kontrolnych). W ogólnym przypadku liczba punktów sterujących nie jest ograniczona - jednak im większa liczba tych punktów tym wyższy stopień wielomianu opisującego krzywą. W praktyce najczęściej wykorzystuje się krzywe trzeciego stopnia definiowane za pomocą czterech punktów sterujących i dalej ograniczymy się do omówienia takich właśnie krzywych.

Na rysunku VII.15 pokazano kilka przykładów krzywych Béziera zdefiniowanych za pomocą czterech punktów sterujących.



Rys. VII.15. Przykładowe krzywe Béziera

Przyglądając się pokazanym przykładom można zauważyć kilka wspólnych cech charakterystycznych dla krzywych Béziera. Istotna jest kolejność punktów sterujących. Krzywa zawsze przechodzi przez punkt pierwszy i ostatni. Odcinek łączący dwa pierwsze punkty krzywej jest zawsze styczny do krzywej w punkcie początkowym. Podobnie, odcinek łączący dwa ostatnie punkty krzywej jest styczny do krzywej w punkcie ostatnim. Ponadto warto również zwrócić uwagę na fakt, iż krzywa zawsze leży we wnętrzu wielokąta opisanego na całym zbiorze punktów sterujących. Możliwe jest również tworzenie krzywych zamkniętych. Zmiana położenia dowolnego punktu sterującego powoduje zmianę kształtu całej krzywej - właściwość ta umożliwia łatwą edycję krzywych.

Równania opisujące krzywe Béziera są zapisywane w postaci parametrycznej. Równanie dla krzywej trzeciego stopnia ma postać

$$B(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t) P_2 + t^3 P_3$$

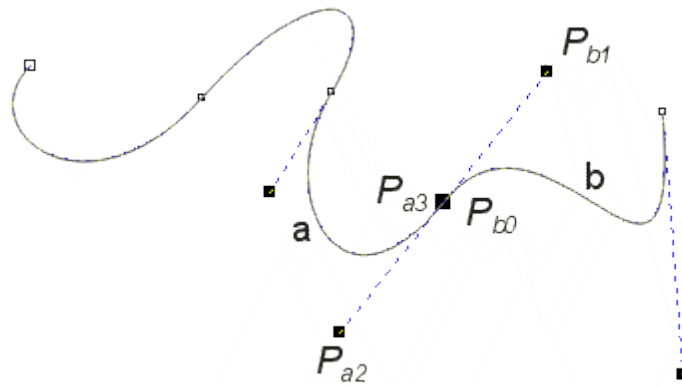
Występujący w równaniu parametr bieżący t zmienia się wzdłuż krzywej od wartości 0 na początku krzywej do wartości 1 na końcu krzywej. Punkty P_i , $i = 0, 1, 2, 3$ są punktami sterującymi krzywej.

Biorąc pod uwagę współrzędne punktów sterujących x oraz y powyższe równanie może być zastąpione dwoma równaniami, które pozwalają wyznaczać wartości współrzędnych x i y punktów leżących na krzywej i określanych przez bieżącą wartość parametru t :

$$x(t) = (1-t)^3 x_0 + 3t(1-t)^2 x_1 + 3t^2(1-t) x_2 + t^3 x_3$$

$$y(t) = (1-t)^3 y_0 + 3t(1-t)^2 y_1 + 3t^2(1-t)y_2 + t^3 y_3.$$

Możliwe jest tworzenie bardziej złożonych krzywych za pomocą składania z segmentów. Przykład pokazano na rysunku VII.16. Zwróćmy uwagę na ciągłość krzywej w punktach łączenia segmentów. Warunkiem uzyskania ciągłości jest to, żeby dwa końcowe punkty sterujące jednego segmentu (P_{a2} , P_{a3} na rysunku) leżały na tej samej prostej co dwa pierwsze punkty sterujące drugiego segmentu (P_{b0} , P_{b1} na rysunku), przy czym punkt ostatni P_{a3} pierwszego segmentu pokrywa się z pierwszym punktem P_{b0} drugiego segmentu.



Rys. VII.16. Przykład krzywej Béziera złożonej z czterech segmentów

Podsumowanie

Poznane w tym wykładzie przekształcenia geometryczne należą do najczęściej wykonywanych operacji przy tworzeniu obrazów na płaszczyźnie. Realizacja tych przekształceń jest często wspomagana sprzętowo, co jest ułatwione między innymi dzięki stosowaniu obliczeń z wykorzystaniem współrzędnych jednorodnych. Omówione algorytmy obcinania pozwalają przyspieszać obliczenia; są one wykorzystywane również przy tworzeniu różnych efektów specjalnych. Krzywe Béziera są wygodne przy tworzeniu i edycji różnego rodzaju linii i konturów krzywoliniowych.

Przykładowe pytania i problemy do rozwiązania

1. Wyjaśnić koncepcję współrzędnych jednorodnych.
2. Znaleźć macierz przekształcenia dla operacji obejmującej przesunięcie trójkąta o wierzchołkach A(2,2), B(4,3), C(3,5) oraz obrót o kąt 45° względem przesuniętego wierzchołka A.
3. Naszkicować krzywą Béziera określona przez następujące punkty sterujące: $P_0(2,2)$, $P_1(3,5)$, $P_2(6,6)$, $P_3(4,3)$.
4. Czy punkt o współrzędnych (1,4) może należeć do krzywej Béziera z problemu 3?

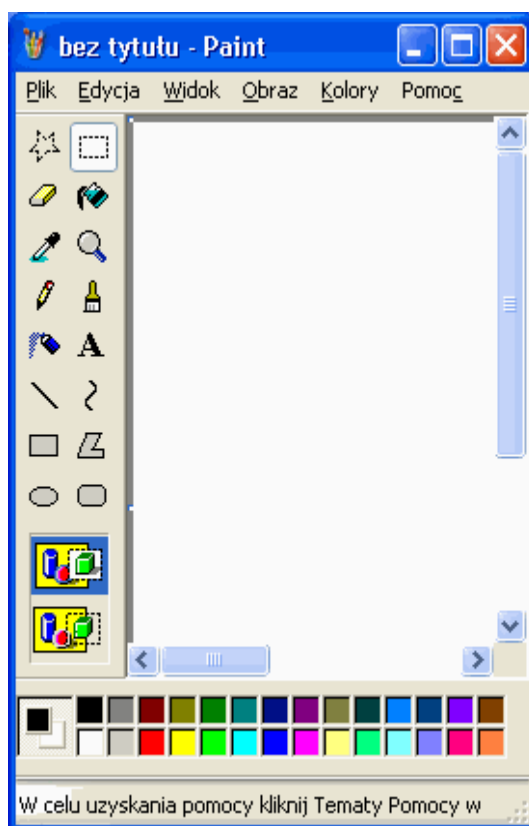
Wykład 8: Narzędzia w programach grafiki 2D

Streszczenie

W poprzednich wykładach poznaliśmy różne algorytmy wykorzystywane w grafice 2D. Algorytmy te są wykorzystywane w różnych programach graficznych. W praktyce wykorzystuje się programy graficzne zarówno rastrowe jak i wektorowe. Te pierwsze pozwalają tworzyć mapy pikselowe, natomiast te drugie umożliwiają tworzenie obiektów i obrazów budowanych z tych obiektów. W grafice 2D w wielu przypadkach, korzystając z komputera postępujemy podobnie jak w przypadku rysowania na kartce papieru. Z reguły mamy do dyspozycji określone pole robocze odpowiadające kartce papieru, na którym możemy rysować oraz mamy do dyspozycji pewien zestaw narzędzi. Niektóre z nich symulują rzeczywiste narzędzia takie jak pióro, pędzel czy gumkę a inne, jak na przykład siatki czy prowadnice, odpowiadające kratce na papierze czy linijce, ułatwiają tworzenie rysunków czy obrazów. Zestaw dostępnych narzędzi jest różny w różnych programach. W wykładzie zostaną przedstawione wybrane, najczęściej spotykane narzędzia.

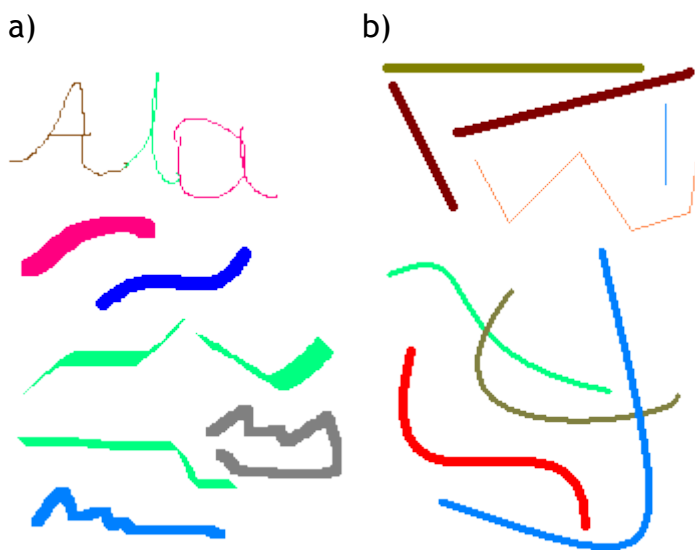
1. Narzędzia w programach rastrowych

Zacznijmy od omówienia narzędzi występujących w programie Paint dostępnym w systemie Windows. Na rysunku VIII.1 pokazano wygląd okna programu Paint. Proponuję odnaleźć ten program i uruchomić go (lub inny prosty program bitmapowy), tak by móc bezpośrednio poznawać działanie omawianych narzędzi. W razie potrzeby można dodatkowo skorzystać z pomocy dostępnej w programie Paint.



Rys. VIII.1. Okno programu Paint

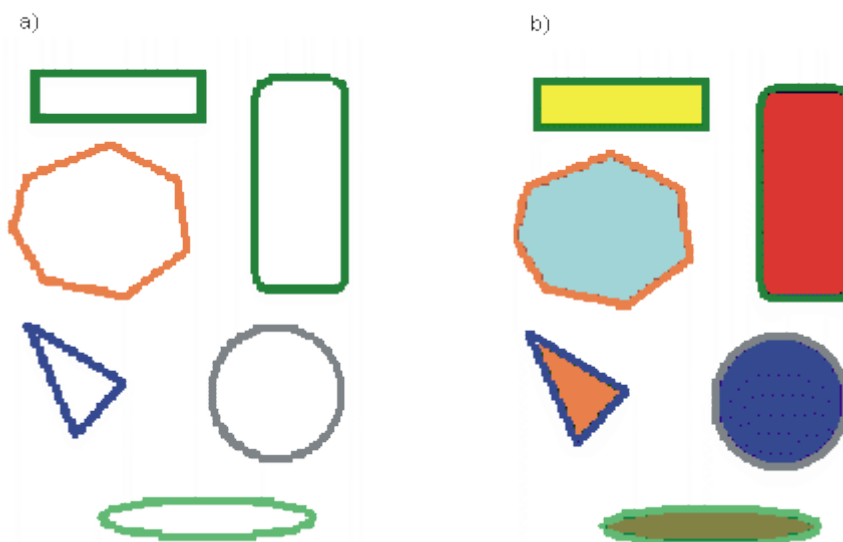
Z lewej strony ekranu programu Paint, w tak zwanym przyborniku, znajdują się ikony ułatwiające dostęp do poszczególnych narzędzi. Zaczniemy od narzędzi do rysowania odręcznego: ołówek i pędzla. Ołówek umożliwia rysowanie przypominające ręczne szkicowanie na papierze. Podobnie pędzel symuluje malowanie na papierze. Możliwy jest wybór kształtu pędzla. Zwróćmy uwagę na różne końcówki pędzla: prostokątną, okrągłą i ukośną. Możliwe jest wybranie koloru rysowania ołówkiem bądź pędzlem. Na rysunku VIII.2a pokazano kilka efektów uzyskanych za pomocą tych dwóch narzędzi.



Rys. VIII.2. Przykłady stosowania narzędzi w programie Paint. a) ołówek i pędzel, b) odcinki i krzywe

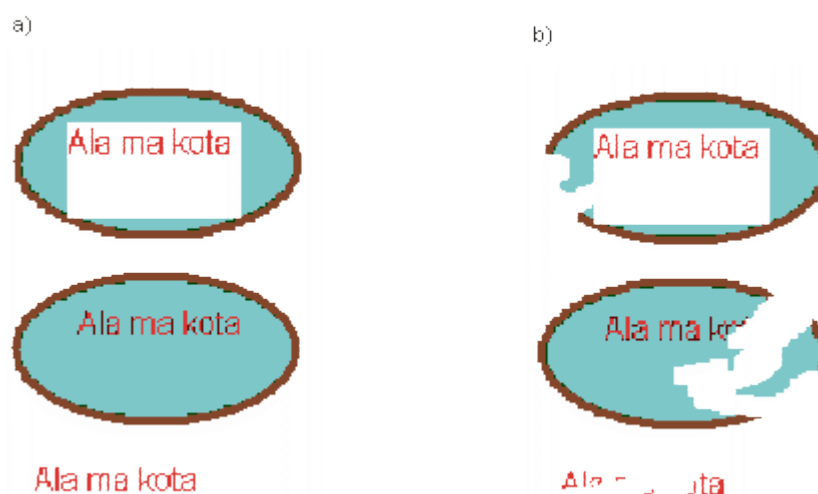
Próbując korzystać z ołówka z pewnością zauważyliśmy, że precyzyjne narysowanie odcinka czy okręgu może sprawiać pewne kłopoty. Stąd dostępne są inne narzędzia przeznaczone do rysowania określonych obiektów i ustalania odpowiednich atrybutów. Przede wszystkim są narzędzia do rysowania odcinków i krzywych. Oba typy obiektów mogą być rysowane różnymi kolorami i można dobierać grubość linii. Kilka przykładów stosowania tych narzędzi pokazano na rysunku VIII.2b. Zwróćmy uwagę na możliwość rysowania łamanych. W celu narysowania krzywej należy narysować odcinek a następnie wskazać myszką punkt odcinka i trzymając lewy przycisk myszki przeciągnąć myszkę w pożądanym kierunku.

Kolejne narzędzia umożliwiają rysowanie wybranych figur: prostokątów i prostokątów z zaokrąglonymi wierzchołkami, wielokątów oraz elips i okręgów (rysowanie elipsy przy wciśniętym klawiszu Shift). Przykładowe figury pokazano na rysunku VIII.3a. Każda figura może zostać wypełniona wybranym kolorem. Służy do tego narzędzie do wypełniania. Umożliwia ono, po wybraniu koloru wypełniania, wskazanie wypełnianego obszaru i wypełnienie tego obszaru. Na rysunku VIII.3b pokazano figury z rysunku VIII.3a wypełnione różnymi kolorami.



Rys. VIII.3. Przykłady: a) rysowania figur oraz b) wypełniania figur

Możliwe jest również umieszczanie tekstu na rysunku z wykorzystaniem różnych czcionek i kolorów. Tekst można wpisywać w wybranym obszarze okna roboczego. Tło tekstu może być przezroczyste lub nieprzezroczyste. Przykłady pokazano na rysunku VIII.4.



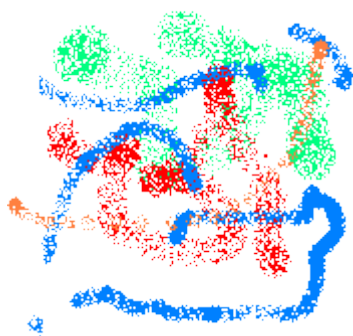
Rys. VIII.4. Przykłady: a) wstawiania tekstu oraz b) zastosowania gumki

Przy rysowaniu można się pomylić, bądź z innych powodów konieczne może być usunięcie któregoś obiektu bądź jego fragmentu. Do tego celu można wykorzystać gumkę. Efekt działania jest pokazany na rysunku VIII.4 b). Można również wybrać odpowiedni fragment rysunku i usunąć go. Wyboru można dokonać albo za pomocą narzędzia zaznaczania, które wybiera prostokątny obszar, albo za pomocą narzędzia zaznaczania dowolnego kształtu, które pozwala ręcznie narysować odpowiedni kształt zaznaczenia. Po wybraniu obszaru usuwa się jego wnętrze klawiszem Del.

Każdorazowo przy wybieraniu koloru można posłużyć się dostępną paletą kolorów wybierając myszką potrzebny kolor. Jeżeli w palecie nie występuje odpowiedni kolor można za pomocą pipety wskazać kolor występujący na obrazku. Inna możliwość polega na skorzystaniu z opcji edycji kolorów. Dostępna paleta kolorów pozwala wybrać żądany kolor na zasadzie wskazania go lub poprzez podanie wartości jego składowych. Można w ten sposób utworzyć własną paletę kolorów niestandardowych. Proponuję bliższe zapoznanie się z tą opcją i utworzenie własnej palety kolorów.

Program umożliwia również powiększanie wskazanego fragmentu obrazu. Warto sprawdzić działanie tego narzędzia.

Program ma również narzędzie o nazwie rozpylacz, umożliwiające tworzenie tak zwanych efektów specjalnych. Możliwe do uzyskania efekty ilustruje rysunek VIII.5. Korzystając z tego narzędzia zwróćmy uwagę na jego działanie zależnie od szybkości przesuwania myszki.



Rys. VIII.5. Przykład działania narzędzia typu rozpylacz

Program Paint zawiera jedynie podstawowe narzędzia spotykane w programach grafiki komputerowej. W innych programach takich narzędzi jest znacznie więcej i mają one znacznie większe możliwości.

Program Paint jest przykładem programów umożliwiających tworzenie grafiki rastrowej. Bezpośrednim efektem w takich programach jest uzyskiwanie mapy pikselowej tworzonego obrazu. Oznacza to, że każdy narysowany obiekt jest bezpośrednio umieszczany w mapie pikselowej i po narysowaniu przestaje być niezależnym obiektem, który może być dalej edytowany. Edycji może podlegać cały obraz lub wybrany fragment mapy pikselowej. Umożliwiają to bardziej rozbudowane programy. Jeden z nich, program Corel PHOTO-PAINT poznamy w trakcie zajęć laboratoryjnych.

2. Narzędzia w programach wektorowych

Inaczej jest w programach umożliwiających tworzenie grafiki wektorowej. W takich programach każdy obiekt stanowi niezależną jednostkę i może być niezależnie edytowany (do momentu kiedy nie zostanie połączony lub zgrupowany z innymi obiektami - wtedy edycji podlega cała grupa, albo do momentu, kiedy utworzony obraz zostanie zamieniony w mapę bitową). Zwróćmy uwagę, że w przypadku

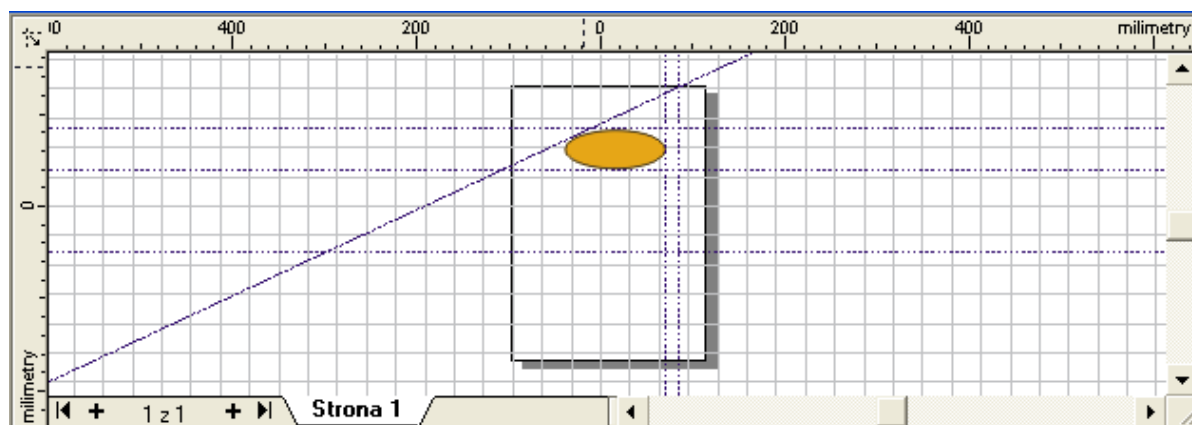
programów wektorowych fakt, iż obraz składający się z kilku obiektów jest wyświetlany jako bitmapa nie oznacza, że poszczególne obiekty utraciły swą odrębność.

W trakcie zajęć laboratoryjnych poznamy profesjonalny wektorowy program graficzny CorelDraw, zawierający bardzo bogaty zestaw narzędzi. Tutaj ograniczymy się jedynie do omówienia kilku typowych narzędzi. Niektóre z nich są dostępne w programie Word w opcji Autokształty. Na rysunku VIII.6 pokazano pasek narzędzi Rysowanie z programu Word. Dostępne tu narzędzia umożliwiają rysowanie odcinków, krzywych oraz wybranych obiektów, a także określanie atrybutów dla tych obiektów. Zachęcam do zapoznania się z tymi narzędziami.



Rys. VIII.6. Pasek narzędzi Rysowanie z programu Word

Przy tworzeniu obiektów bardzo przydatne są wszelkie narzędzia ułatwiające pracę, takie jak linijki, siatki czy prowadnice oraz narzędzia przyciągania. Narzędzia te pozwalają precyzyjnie określać położenie i rozmiary obiektów. Na rysunku VIII.7 pokazano przykład pola roboczego z zaznaczonymi linijkami, siatką i prowadnicami.



Rys. VIII.7. Pole robocze z zaznaczonymi linijkami, siatką i prowadnicami

Jak widać, linijki pozioma i pionowa są wyskalowane w milimetrach. Prowadnice zaznaczone są liniami przerywanymi. Pokazane są trzy prowadnice poziome, dwie pionowe i jedna ukośna. Z siatką oraz z prowadnicami może być związana opcja przyciągania. Opcja ta pozwala na łatwe rysowanie różnych obiektów, na przykład odcinków wzdłuż linii siatki, oraz rysowanie obiektów, których wierzchołki mogą znajdować się jedynie w węzłach siatki

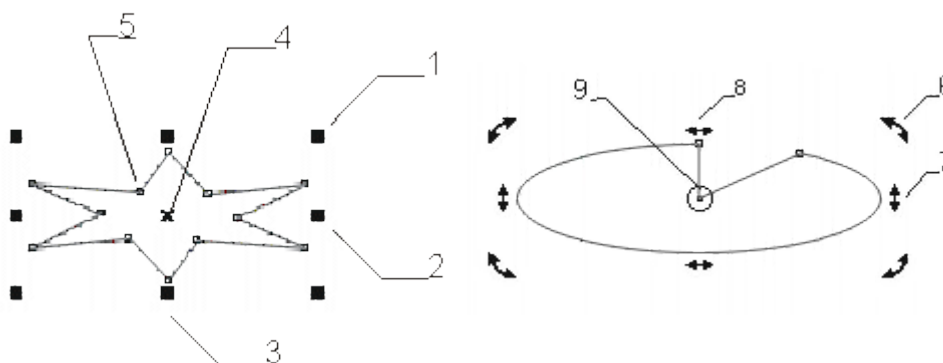
Ogólnie, koncepcja realizacji opcji przyciągania polega na tym, że wokół przyciągającego obiektu tworzy się umowne pole. Każdy obiekt, który znajdzie się w tym polu jest przyciągany do obiektu przyciągającego, podobnie jak elementy metalowe są przyciągane przez magnes.

W programach graficznych spotkamy się również z rozwiązaniami ułatwiającymi tworzenie i ewentualne późniejsze modyfikowanie obrazów. Niżej omówimy kilka przykładowych rozwiązań: interakcję z obiektami, koncepcję warstw, koncepcję grupowania obiektów, koncepcję hierarchii.

W czasie tworzenia rysunku lub jego modyfikacji wielokrotnie powstaje konieczność powrotu do obiektu wcześniej narysowanego i wykonania na nim pewnych operacji. Podstawowym elementem dla tego typu operacji jest umożliwienie wskazywania obiektu. Najczęściej realizuje się to poprzez zwykłe wskazywanie obiektu za pomocą myszki (po wybraniu narzędzia do wybierania). Można również korzystać z opcji polegającej na ujęciu obiektu w ramkę. Wybrany obiekt może być poddany różnym przekształceniom, podobnie jak przy tworzeniu obiektu.

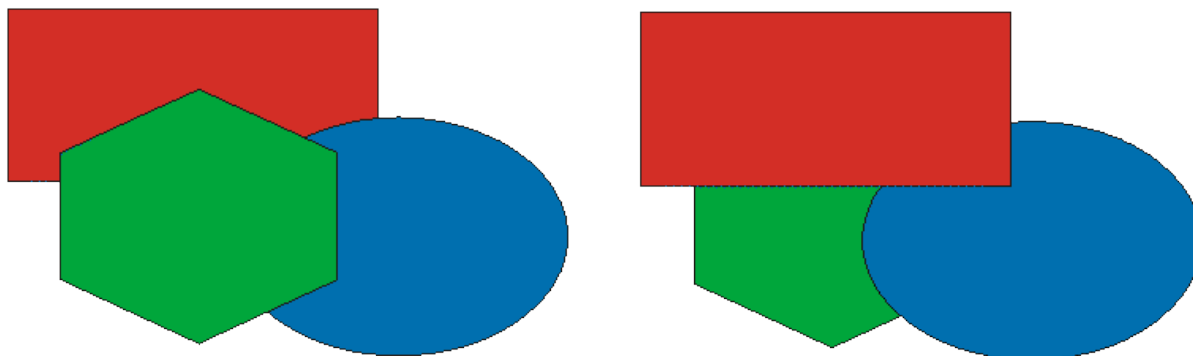
Wskazane obiekty można przeciągać - obiekt może być wybrany i następnie przesunięty w inne miejsce. Wskazany obiekt może być również edytowany i można zmieniać jego atrybuty (kolor, grubość linii konturu itd.).

Po wybraniu obiektu z reguły pojawiają się pomocnicze znaczniki (uchwyty), które ułatwiają wykonywanie odpowiednich przekształceń. Na rysunku VIII.8 pokazano dwa przykłady uchwytów i zaznaczono jakie funkcje można realizować przy ich pomocy.



Rys. VIII.8. Przykłady uchwytów: 1 - skalowanie w obu kierunkach, 2 - skalowanie w poziomie, 3 - skalowanie w pionie, 4 - przesuwanie, 5 - zmiana kształtu obiektu, 6 - obracanie obiektu wokół środka obrotu, 7 - pochylanie w pionie, 8 - pochylanie w poziomie, 9 - ustawianie środka obrotu

W czasie tworzenia rysunku niejednokrotnie powstaje sytuacja, kiedy jedne obiekty nachodzą na inne. Wtedy istotne jest ustalenie kolejności rysowania obiektów. Obiekt przykrywający inny obiekt powinien być narysowany później. Na rysunku VIII.9 pokazano te same obiekty ustawione w różnej kolejności.



Rys. VIII.9. Zmiana kolejności obiektów

Czasami mamy do czynienia z obiektami, które są widoczne w czasie tworzenia rysunku, jak na przykład siatka, natomiast nie powinny być widoczne w końcowej wersji rysunku - ten problem można rozwiązać za pomocą przypisywania obiektom atrybutu widoczności. Inny sposób rozwiązania tego typu problemów polega na wykorzystaniu koncepcji warstw - poszczególne obiekty są przypisywane do różnych warstw. Istotna jest kolejność ułożenia warstw. Wszystkie warstwy mają przypisany ten sam układ współrzędnych, dzięki czemu zostają zachowane wzajemne relacje położenia elementów znajdujących się na różnych warstwach. Każda warstwa ma swoje atrybuty i może być tworzona niezależnie. Najczęściej jedna z warstw może być uznana za warstwę wyróżnioną w tym sensie, że wszystkie elementy umieszczone na tej warstwie pojawiają się również na wszystkich innych warstwach. Końcowy wygląd rysunku uzyskuje się poprzez wyświetlenie, we właściwej kolejności, warstw, którym przypisano atrybut widoczności.

Często pewne operacje chcemy wykonywać w odniesieniu do kilku obiektów, bądź zależy nam na tym żeby kilka obiektów zachowywało się identycznie. Wtedy wygodną opcją jest połączenie tych obiektów w jedną całość - operację taką określa się jako grupowanie obiektów. Od chwili wykonania grupowania obiekty, które weszły w skład grupy są traktowane jako jedna całość - jako nowy obiekt. Przy wszystkich operacjach wykonywanych na tym obiekcie relacje między obiektami składowymi nie zmieniają się. W razie potrzeby istnieje możliwość cofnięcia funkcji grupowania i ponownego uniezależnienia poszczególnych obiektów.

Końcowym efektem tworzenia rysunku za pomocą programów grafiki 2D jest albo obraz w postaci mapy bitowej albo opis obrazu w postaci wektorowej. Z reguły obraz jest zapisany w jednym z formatów dostępnych w danym programie. Przykładowe formaty zostaną omówione w wykładzie XV. Tutaj ograniczymy się jedynie do zwrócenia uwagi jeszcze raz na to, że zapisując obraz w postaci mapy bitowej tracimy informację o poszczególnych obiektach rysunku, a tym samym możliwość ich późniejszej edycji. Obrazy pamiętane w postaci wektorowej przed wyświetleniem są zawsze konwertowane do postaci mapy bitowej.

Podsumowanie

Wykład poświęcony był programom grafiki 2D, a w szczególności wybranym narzędziom dostępnym w tych programach. Zwróciliśmy uwagę na programy do

grafiki rastrowej i wektorowej, podkreślając różnice między nimi. Naturalnie powinniśmy mieć świadomość, że profesjonalne programy grafiki 2D mają wiele innych narzędzi niż te, które omówiliśmy w trakcie wykładu. Wiele nowych narzędzi poznamy w czasie zajęć laboratoryjnych.

Przykładowe pytania i problemy do rozwiązania

1. Wymienić podstawowe różnice między programami grafiki rastrowej i programami grafiki wektorowej.
2. Korzystając z programu Paint (lub innego dostępnego programu) narysować obiekty pokazane na rysunku VIII.3.
3. W jaki sposób można rozwiązać problem określania kolejności wyświetlania obiektów, tak jak to pokazano na rysunku VIII.9.
4. Wyjaśnić na czym polega koncepcja przyciągania do linii siatki lub prowadnic.

Wykład 9: Modelowanie obiektów i scen 3D

Streszczenie

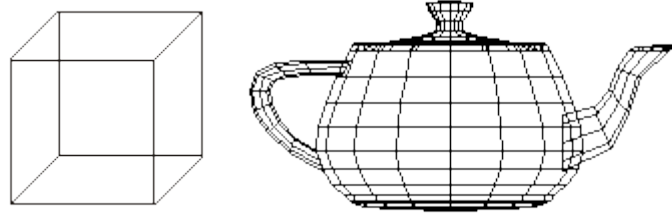
Dotychczas zajmowaliśmy się tworzeniem obrazów na płaszczyźnie, a więc grafiką 2D. Kolejny krok w poznawaniu metod grafiki komputerowej to grafika 3D, w której pojawiają się nowe problemy wymagające rozwiązania. Mówiąc o grafice 3D z reguły wyróżnia się dwa etapy. Pierwszy etap polega na modelowaniu obiektów przestrzennych oraz tworzeniu sceny w której znajdują się przygotowane wcześniej obiekty oraz źródła światła. W drugim etapie realizowany jest ciąg operacji prowadzący do wyświetlenia sceny na ekranie. Etap ten ogólnie jest określany jako rendering sceny. W tym wykładzie zajmiemy się metodami modelowania obiektów i scen.

1. Modelowanie obiektów

W fazie modelowania tworzone są modele poszczególnych obiektów oraz scena zawierająca te obiekty. Scena ta jest podstawą generowania obrazu na ekranie. Każdy obiekt najczęściej jest generowany w swoim układzie współrzędnych i dopiero w fazie tworzenia sceny jest przenoszony do wspólnego układu współrzędnych sceny określanego często jako układ współrzędnych świata.

Znanych jest wiele różnych metod modelowania obiektów trójwymiarowych - każda z nich ma swój obszar stosowania. Dalej ograniczymy się do omówienia najczęściej stosowanych metod.

Bodaj najprostszą metodą modelowania obiektów jest metoda szkieletowa (drutowa). W metodzie tej określa się jedynie szkielet bryły, a więc definiuje się położenie wierzchołków i krawędzi występujących w obiekcie. Przykłady modeli szkieletowych pokazano na rysunku IX.1. Model szkieletowy nie zawiera informacji o powierzchni bocznej bryły ani tym bardziej o jej wnętrzu. Dzięki swej prostocie modele szkieletowe umożliwiają szybkie tworzenie obiektów 3D o wymaganych kształtach i są często stosowane w fazie wstępnej definiowania bryły.



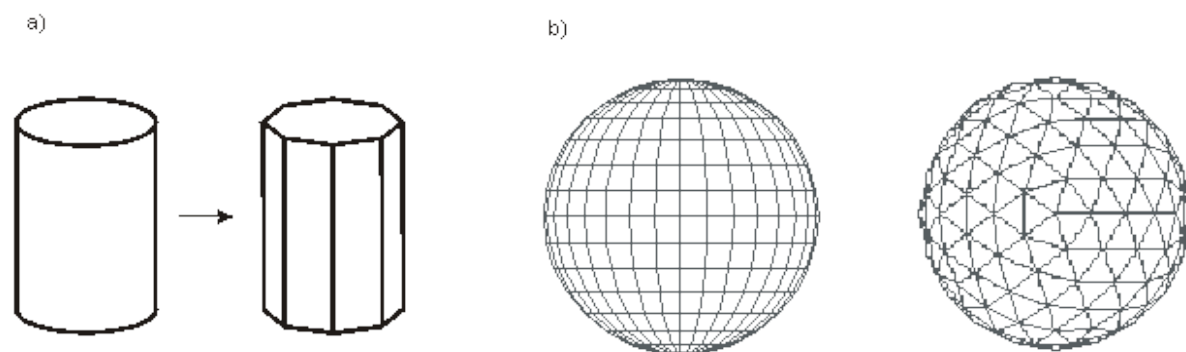
Rys. IX.1. Przykładowe modele szkieletowe

Model szkieletowy stanowi często podstawę dla utworzenia bardziej szczegółowego modelu, w którym opisuje się powierzchnię zewnętrzną obiektu. Metody, które to umożliwiają, są określane jako metody Brep (ang. boundary representation).

W większości zastosowań do opisu powierzchni zewnętrznej wykorzystuje się reprezentację wielokątową - powierzchnia boczna bryły jest opisywana za pomocą zbioru wielokątów. W przypadku takich brył jak wielościany reprezentacja ta jest w pełni naturalna. W przypadku gdy powierzchnie boczne nie są płaskie dokonuje się aproksymacji wielokątowej tych powierzchni.

W niektórych zastosowaniach, gdzie zależy nam na jak największym realizmie generowanych obrazów, powierzchnie boczne złożonych obiektów opisuje się za pomocą odpowiednich powierzchni krzywoliniowych (na przykład powierzchni Béziera albo powierzchni NURBs (ang. nonuniform rational B-splines)).

Na rysunku IX.2 pokazano przykład aproksymacji powierzchni walca za pomocą wielokątów. Pokazano również dwie reprezentacje powierzchni kuli (proszę się zastanowić nad sposobami uzyskania takich reprezentacji kuli oraz nad różnicami między tymi reprezentacjami).

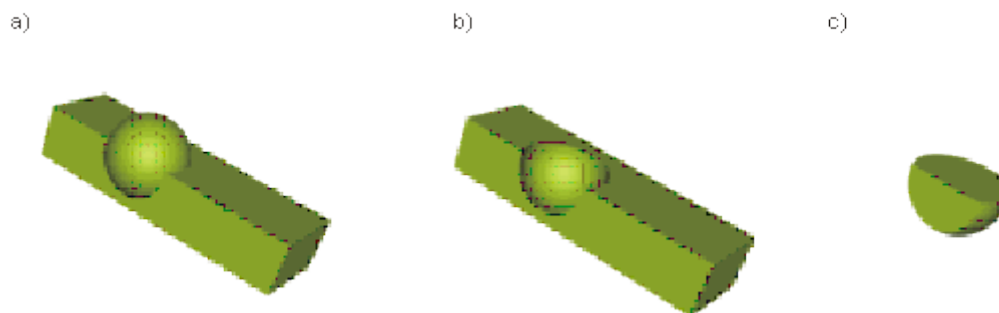


Rys. IX.2. Przykłady wielokątowej aproksymacji powierzchni: a) walec, b) dwa sposoby aproksymacji kuli

W praktyce najczęściej do aproksymacji wykorzystuje się trójkąty. Wynika to z faktu, że każdy trójkąt jednoznacznie wyznacza powierzchnię na której leży. Liczba wielokątów (trójkątów) aproksymujących złożony obiekt może być bardzo duża. Niczym wyjątkowym nie są przypadki aproksymowania powierzchni za pomocą kilkuset tysięcy trójkątów. Im więcej szczegółów zawiera powierzchnia boczna obiektu, tym więcej trójkątów potrzeba do jej aproksymacji. Należy jednak pamiętać o konieczności zachowania kompromisu między dokładnością

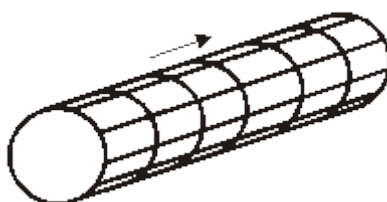
modelowania powierzchni a czasem potrzebnym do jej renderowania - im więcej wielokątów tym dłuższy czas renderingu.

Inny sposób modelowania obiektów 3D zapewnia tak zwana metoda CSG (ang. constructive solid geometry). W metodzie tej zakłada się, że dostępny jest zestaw podstawowych brył (tak zwanych prymitywów), z których można tworzyć bryły bardziej złożone. Wykorzystuje się przy tym operację łączenia brył ze sobą, operację znajdowania części wspólnej brył oraz operację odejmowania brył. Na rysunku IX.3 pokazano przykłady ilustrujące działanie tych operacji.



Rys. IX.3. Przykłady operacji wykonywanych w metodzie CSG. (a) Dodawanie brył, (b) odejmowanie brył, (c) część wspólna brył

Kolejną metodą modelowania brył określaną jest jako metoda przesuwania (ang. sweeping). W metodzie tej definiowany jest przekrój bryły, który następnie jest przesuwany wzdłuż zadanej ścieżki. W czasie przesuwania jest wyznaczana powierzchnia boczna bryły. Ścieżka może być linią prostą albo krzywą (na przykład krzywą Béziera). W czasie przesuwania przekrój bryły może się zmieniać. W szczególnym przypadku, gdy przekrój jest obracany wokół pewnej osi możliwe jest tworzenie brył obrotowych. Na rysunku IX.4 pokazano przykład ilustrujący działanie metody.



Rys. IX.4. Modelowanie metodą przesuwania

W niektórych zastosowaniach wykorzystuje się takie metody modelowania, które umożliwiają reprezentowanie całej bryły a nie tylko jej powierzchni zewnętrznej. Klasyczną metodą w tym zakresie jest metoda wkselowa (objętościowa). W metodzie tej wykorzystuje się pojęcie wksela. Woksel jest to elementarna objętość w przestrzeni 3D, reprezentowana najczęściej jako najmniejszy sześciąt, którym operujemy w fazie modelowania. Woksel można traktować jako komórkę rastra przestrzennego - przestrzenny odpowiednik piksela na płaszczyźnie. Modelowanie obiektu 3D sprowadza się do określenia zbioru wkseli należących do obiektu.

Metoda wkselowa jest bardzo wymagająca jeśli chodzi o pojemność pamięci potrzebnej do zapamiętania informacji o obiekcie. Załóżmy dla przykładu, że ograniczamy precyzję modelu do sześciastu jednostek. Oznacza to, że musimy liczyć się z koniecznością zarezerwowania miejsca w pamięci dla przechowania informacji o $512^3 = 2^{27}$ wkselach (a nie jest to wcale duża rozdzielczość w stosunku do potrzeb). Jednak niewątpliwą zaletą metody jest to, że z każdym wkselom można związać dodatkową informację w postaci odpowiednich atrybutów. Uzyskujemy w ten sposób model, który dostarcza informację nie tylko o tym, które fragmenty przestrzeni należą do obiektu 3D, ale również o innych cechach poszczególnych fragmentów przestrzeni. Tworząc na przykład model wkselowy jakiejś części ciała człowieka można z każdym wkselom związać informację o tym czy w danym miejscu jest tkanka miękka, tkanka kostna, naczynie krwionośne itd. Mając taki model można z kolei wyznaczać różne przekroje albo odtwarzać powierzchnie zewnętrzne różnych narządów wewnętrznych itd.

Uzyskane modele brył mogą być poddawane różnego przekształceniom geometrycznym: przesuwaniu, obrotom, skalowaniu itd. Wykorzystuje się przy tym najczęściej koncepcję współrzędnych jednorodnych, podobnie jak na płaszczyźnie. Z tym, że teraz każdy punkt jest reprezentowany za pomocą czterech współrzędnych $(x, y, z, 1)$. Macierze dla podstawowych przekształceń mają następujące postacie.

Przesunięcie o wektor (t_x, t_y, t_z) :

$$\begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Skalowanie ze współczynnikami skalowania S_x, S_y, S_z :

$$\begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

W przypadku obrotów są trzy macierze - odpowiednio dla obrotu wokół osi x, wokół osi y i wokół osi z. W prawoskrętnym układzie współrzędnych dodatni kąt obrotu oznacza obrót w kierunku przeciwnym do ruchu wskazówek zegara jeżeli patrzy się wzdłuż osi w kierunku początku układu współrzędnych.

Obrót wokół osi x:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \Theta & -\sin \Theta & 0 \\ 0 & \sin \Theta & \cos \Theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Obrót wokół osi y:

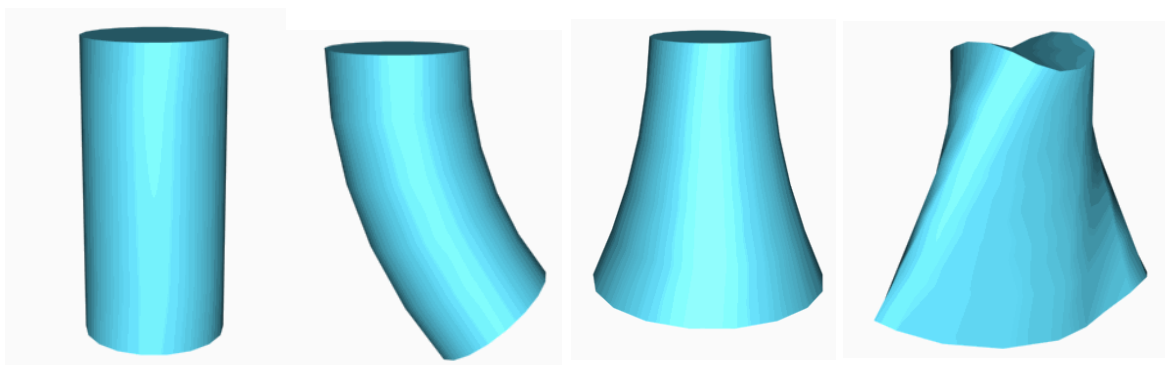
$$\begin{bmatrix} \cos \Theta & 0 & \sin \Theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \Theta & 0 & \cos \Theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Obrót wokół osi z:

$$\begin{bmatrix} \cos \Theta & -\sin \Theta & 0 & 0 \\ \sin \Theta & \cos \Theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Możliwe jest również składanie przekształceń i wyznaczanie macierzy wypadkowej dla określonego ciągu przekształceń. W szczególności można w ten sposób znaleźć macierz obrotu wokół innej osi niż jedna z osi układu współrzędnych.

Poza podstawowymi przekształceniami geometrycznymi możliwe jest stosowanie różnych modyfikacji obiektów, takich jak na przykład skręcanie, wyginanie, przewężanie, rozciąganie itd. Przykłady takich modyfikacji pokazano na rysunku IX.5.



Rys. IX.5. Przykłady modyfikacji obiektów 3D

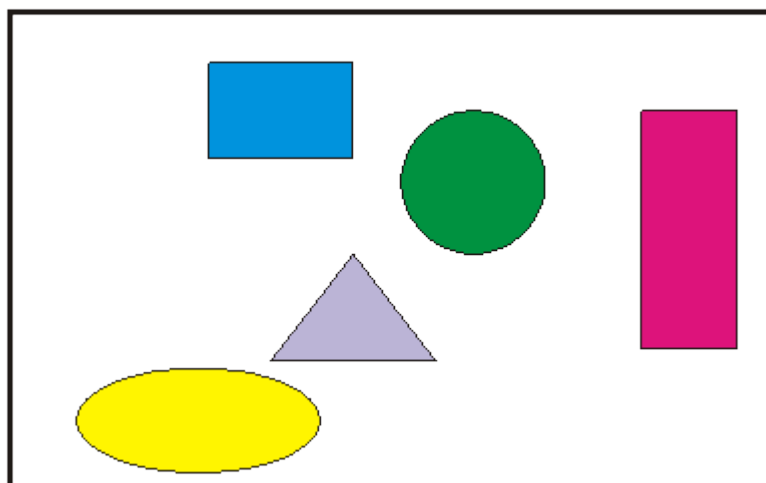
Bardziej złożone obiekty mogą być modelowane z wykorzystaniem koncepcji hierarchii. Elementy składowe obiektu mogą być łączone w jedną wspólną strukturę. Wtedy zastosowanie transformacji do jednego obiektu przenosi się na obiekty z nim związane.

Zwróćmy w tym miejscu uwagę na to, że w fazie modelowania poszczególne obiekty mogą być tworzone w swoich układach współrzędnych. Przy tworzeniu bardziej złożonego obiektu, konieczne jest przeniesienie obiektów składowych do jednego wspólnego układu współrzędnych. Na przykład, przy modelowaniu samochodu nadwozie, podwozie i koła mogą być modelowane w zupełnie niezależnych układach. Jednak przy tworzeniu modelu całego samochodu poszczególne elementy muszą być połączone w jednym wspólnym układzie współrzędnych. Zmiana układu współrzędnych polega na wykonaniu ciągu przekształceń geometrycznych: obrotów, przesunięć i skalowania.

Etap modelowania obiektów dostarcza informacji o kształcie obiektu. Przed przejściem do fazy renderingu konieczne jest jeszcze określenie atrybutów bryły, które umożliwią poprawny rendering. W szczególności trzeba określić kolor obiektu, rodzaj materiału z jakiego jest wykonany obiekt, rodzaj tekstury jaką mają być pokryte powierzchnie boczne obiektu itp. Rodzaj materiału decyduje o właściwościach refleksyjnych powierzchni (czasami również o właściwościach emisyjnych powierzchni). Pozwala to modelować obiekty o powierzchniach pochłaniających, matowych, lśniących, błyszczących czy odbijających zwierciadlanie. Dodajmy od razu, że końcowy widok obiektu będzie dodatkowo zależał od jego oświetlenia, kąta obserwacji itd.

2. Modelowanie sceny

Po opracowaniu modeli poszczególnych obiektów i określeniu ich atrybutów można przystąpić do projektowania sceny, która będzie potem renderowana i prezentowana na ekranie. Projektowanie sceny polega na przeniesieniu przygotowanych modeli obiektów składowych do jednego układu współrzędnych (tak zwanego układu świata) i umieszczeniu ich w odpowiednich pozycjach. Na rysunku IX.6 pokazano przykładową scenę (widok z góry):



Rys. IX.6. Przykładowa scena

Dla pełnego zdefiniowania sceny konieczne jest jeszcze określenie tła i określenie sposobu oświetlenia sceny, a więc dobranie i rozmieszczenie w scenie źródeł

światła. Każde światło może być ustawione w wybranym miejscu sceny i skierowane w odpowiednią stronę.

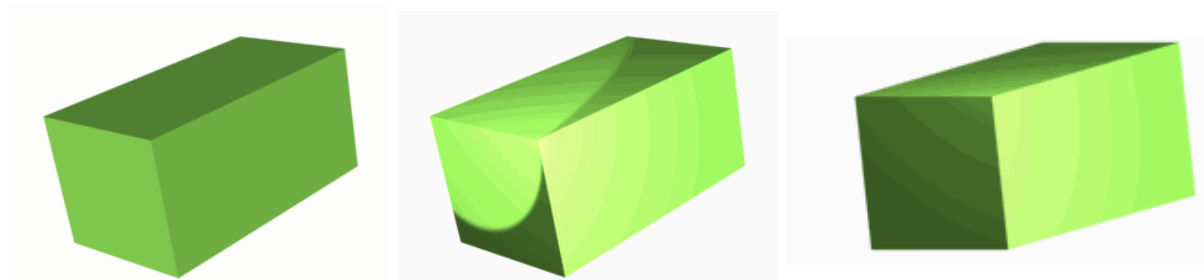
W praktyce stosowane są różne rodzaje światła. Jasność obiektów oświetlanych przez poszczególne źródła zależy od intensywności źródła, kierunku pod jakim padają promienie na obiekt oraz często również od odległości źródła od obiektu.

Najczęściej używa się punktowe źródło światła. Promienie świetlne generowane przez takie źródło rozchodzą się z jednego punktu równomiernie we wszystkich kierunkach. Jeżeli źródło punktowe jest dostatecznie daleko od obiektu, to często przyjmuje się, że promienie świetlne docierają do obiektu jako promienie biegnące równoległe do siebie i w związku z tym oświetlają równomiernie powierzchnię obiektu. W takim przypadku mówi się o kierunkowym źródle światła.

Spośród innych modelowanych źródeł światła, należy wymienić jeszcze źródła o modelowanej charakterystyce rozchodzenia się promieni, takie jak źródła stożkowe (promienie ze źródła punktowego rozchodzą się tylko w obrębie stożka o zadanym kącie rozwarcia) czy źródła reflektorowe (promienie rozchodzą się tylko w obrębie walca o określonym promieniu podstawy).

Obok wymienionych różnych źródeł światła należy wymienić również tak zwane światło otoczenia. Światło to nie jest związane z żadnym konkretnym źródłem światła. Natomiast pozwala ono uwzględnić fakt, iż w rzeczywistości obiekty są widoczne nawet przy braku konkretnych źródeł światła. Na przykład w ciągu dnia w pomieszczeniu, w którym nie ma żadnego źródła światła natomiast jest okno, jesteśmy w stanie obserwować wszystkie znajdujące się w pomieszczeniu obiekty.

Na rysunku IX.7 pokazano przykładową bryłę oświetloną przez różne źródła światła.



Rys. IX.7. Bryła oświetlona przez różne źródła światła

Przestrzeń, w której tworzy się całą scenę określa się jako przestrzeń sceny. W przestrzeni sceny poza obiektami umieszcza się źródła światła. Zdefiniowana scena może być oglądana z różnych punktów obserwacji i pod różnymi kątami. W celu określenia sposobu obserwacji sceny wprowadza się pojęcie obserwatora; korzysta się również z równoważnych pojęć: kamera lub oko obserwatora. Parametry obserwatora określają jego położenie i kierunek patrzenia.

Podsumowanie

Wykład zapoznał nas z różnymi metodami modelowania brył. Każda z tych metod znajduje zastosowanie w odpowiednich aplikacjach. Pokazane zostały również podstawowe metody przekształcania geometrycznego brył oraz modyfikacji brył. Metody te są wykorzystywane w czasie modelowania brył oraz modelowania sceny. Modelując scenę nie można zapomnieć o oświetleniu sceny. W następnych wykładach zajmiemy się metodami wykorzystywanymi w fazie renderowania obiektów oraz całych scen.

Przykładowe pytania i problemy do rozwiązania

1. Wymienić podstawowe różnice między metodami modelowania: Brep, CSG i wokselową.
2. Naszkicować stożek oraz przykładową reprezentację wielokątową jego powierzchni bocznej.
3. Wyznaczyć wypadkową macierz dla operacji obrotu względem osi równoległej do osi z.
4. Korzystając z rysunku IX.7 zwrócić uwagę na wpływ oświetlenia na wygląd obiektu.

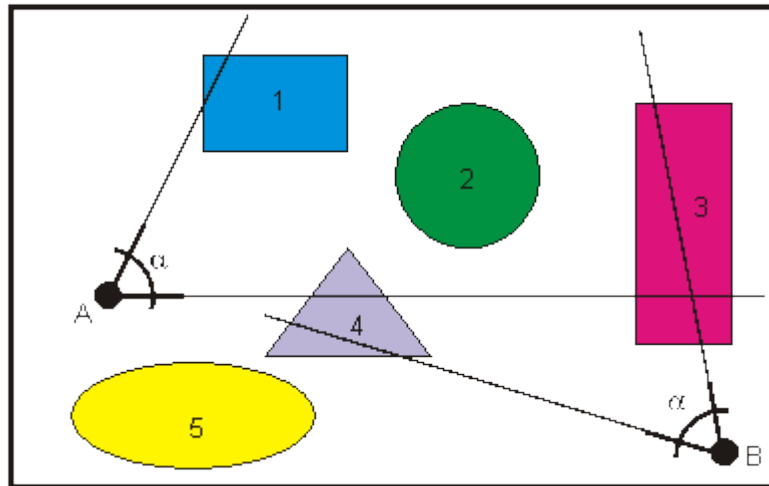
Wykład 10: Rzutowanie i eliminowanie powierzchni niewidocznych

Streszczenie

Zadaniem fazy renderingu jest wykonanie odpowiednich obliczeń w odniesieniu do przygotowanej sceny tak by ostatecznie można było ją wyświetlić na ekranie. Wyświetlony obraz powinien mieć wymagany stopień realizmu oraz nie powinien zawierać elementów, które byłyby sprzeczne z intuicją obserwatora uzyskaną w wyniku oglądania rzeczywistego świata. Na ekranie powinny być pokazane tylko te elementy sceny, które są widoczne dla obserwatora. Konieczne jest więc określenie położenia obserwatora względem sceny i kierunku patrzenia. Obraz końcowy jest obrazem 2D i konieczne jest wobec tego wykonanie odpowiedniego rzutowania sceny 3D na płaszczyznę ekranu. Ważne jest przy tym eliminowanie obiektów niewidocznych dla obserwatora. Następnie powierzchnie widocznych elementów powinny być odpowiednio pocienione. Istotne jest również uwzględnienie cieni rzucanych przez obiekty. Dla realizacji poszczególnych wymienionych operacji opracowano wiele różnych algorytmów. W tym i w następnym wykładzie poznamy metody metody wykorzystywane w fazie renderingu.

1. Bryła widzenia

Zdefiniowana scena może być obserwowana z różnych punktów. Obserwator może znajdować się w dowolnym miejscu sceny, z tym, że na ogół zakłada się, że obserwator jest na zewnątrz poszczególnych obiektów. Ponadto, obserwator może patrzeć w dowolnym kierunku. Zależnie od położenia obserwatora i kierunku patrzenia na ekranie powinien pokazać się odpowiedni widok sceny. Poglądowo wyjaśnia to rysunek X.1 - dla uproszczenia pokazano widok z góry. Założono również, że kąt widzenia α jest dla obserwatora stały.

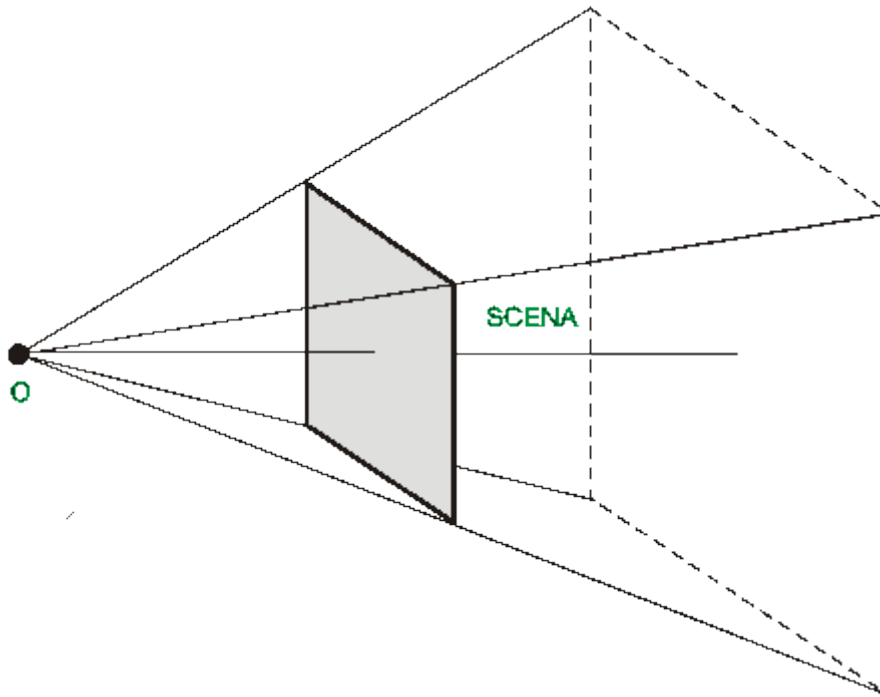


Rys. X.1. Wpływ położenia i kierunku obserwacji na widoczne dla obserwatora elementy sceny. Zaznaczono dwa przykładowe położenia obserwatora: A i B

Przy położeniu A obserwatora można się spodziewać, że na ekranie potencjalnie mogą się pojawić wszystkie obiekty poza 5, z tym, że obiekty 1, 2 i 4 będą widoczne jedynie częściowo a obiekt 3 prawdopodobnie nie będzie widoczny, bowiem będzie zastąpiony przez obiekty leżące bliżej obserwatora. Przy położeniu B obserwatora ponownie obiekt 5 nie będzie widoczny na ekranie, natomiast wszystkie pozostałe obiekty będą widoczne jedynie częściowo.

Z analizy rysunku można wysnuć kilka wniosków. Nie zawsze wszystkie obiekty znajdujące się w scenie są widoczne. Nie zawsze widoczne są całe obiekty, bowiem albo nie mieszczą się w całości w polu widzenia obserwatora określonym przez kąt α albo są zastąpione częściowo przez inne obiekty. Z punktu widzenia szybkości obliczeń nie jest wskazane wykonywanie obliczeń związanych z obiektami, które w całości leżą poza polem widzenia obserwatora.

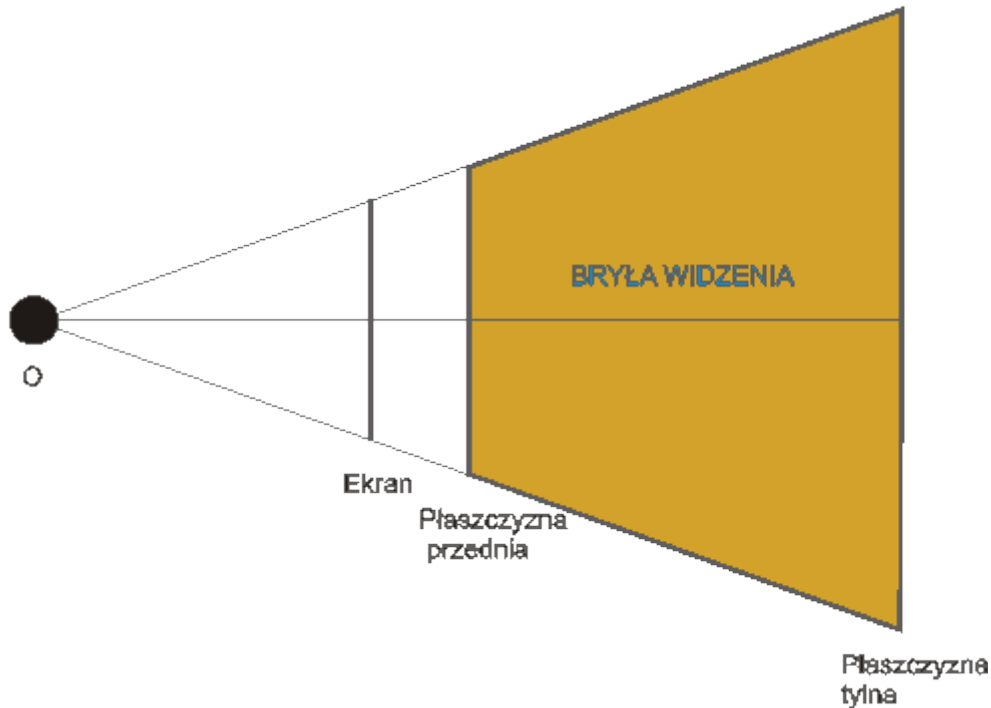
W rzeczywistości, obserwując scenę 3D obserwator powinien zobaczyć na ekranie te elementy, które znajdują się w prawidłowym ostrostupie czworokątnym, którego wierzchołek znajduje się w "oku" obserwatora O, oś główna ostrostupa jest prostopadła do płaszczyzny ekranu a kąt rozwarcia ostrostupa jest określony przez wielkość ekranu i jego odległość od obserwatora. Poglądowo wyjaśnia to rysunek X.2.



Rys. X.2. Ostrostup, w obrębie którego znajdują się obiekty sceny potencjalnie widoczne na ekranie

Teoretycznie wysokość ostrostupa powinna być nieskończenie długa. W praktyce jednak wprowadza się dodatkowe ograniczenia w postaci płaszczyzny przedniej i płaszczyzny tylnej, które ograniczają ostrostup do ostrostupa ściętego. Ograniczenia te wynikają ze względów czysto praktycznych. Nie ma bowiem sensu rozpatrywać obiektów, które znajdują się zbyt daleko od obserwatora i są na przykład zasłonięte albo są zbyt małe. Podobnie nie jest sensowne zajmowanie się obiektami, które na przykład znajdują się za obserwatorem i nie powinny pojawić się na ekranie.

Na rysunku X.3 pokazano widok z boku otrzymanego ostrostupa ściętego. Ostrostup ten jest często określany jako bryła widzenia. Bryła widzenia obejmuje więc tę część przestrzeni sceny, która może być widoczna na ekranie. W efekcie, ze względu na szybkość obliczeń sensowne jest rozważanie tylko tych obiektów sceny, które w całości lub przynajmniej częściowo należą do bryły widzenia.

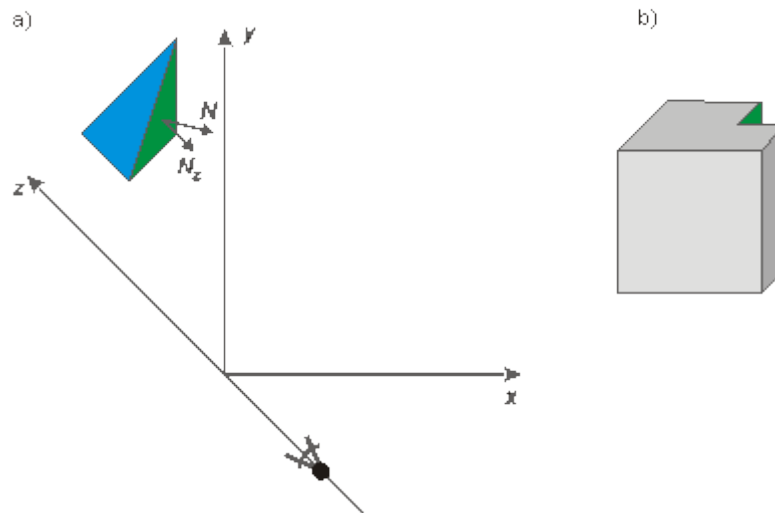


Rys. X.3. Widok z boku ostrostupa ściętego tworzącego bryłę widzenia

2. Eliminowanie powierzchni niewidocznych

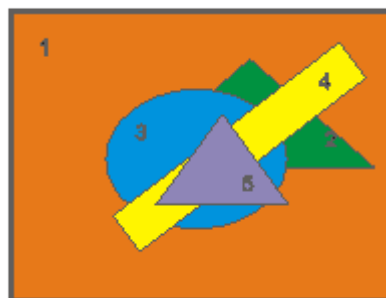
Obserwując dowolny obiekt nieprzezroczysty zawsze widzimy tylko część jego powierzchni bocznej. Musimy oczywiście uwzględnić ten fakt przy wyświetlaniu obiektu na ekranie. Podobnie, wyświetlając kilka obiektów musimy uwzględniać fakt, że jedne obiekty mogą zasłaniać inne. W celu rozwiązania problemu widoczności opracowano różne algorytmy. Niżej przedstawimy trzy najczęściej stosowane.

Zacznijmy od najprostszego przypadku kiedy mamy do czynienia z pojedynczą wypukłą bryłą, której powierzchnia boczna jest opisana za pomocą wielokątów. Dla każdego wielokąta można wyróżnić stronę wewnętrzną (widzianą przy obserwacji od strony wnętrza bryły) i stronę zewnętrzną (widzianą przy obserwacji bryły z zewnątrz). Można również określić wektor normalny do powierzchni skierowany na zewnątrz bryły. Sprawdzając dla konkretnej ściany bocznej bryły czy jej wektor normalny (a dokładniej składowa z tego wektora) jest skierowany do obserwatora czy w przeciwną stronę, możemy stwierdzić czy ta ściana (wielokąt) jest widoczna. Ilustruje to rysunek X.4. Zwróćmy jeszcze raz uwagę na założenie o wypukłości bryły. W przypadku brył, które nie są wypukłe algorytm może dawać niepoprawne wyniki. Na przykład zielona ścianka z rysunku X.4 przy obserwacji z określonego punktu może zostać zakwalifikowana jako widoczna, podczas gdy faktycznie widoczna będzie tylko część tej ścianki.



Rys. X.4. Ilustracja do algorytmu określania widoczności ścianek na podstawie analizy wektorów normalnych. a) Przypadek bryły wypukłej, b) przypadek bryły niewypukłej

Inny algorytm rozwiązywania problemu widoczności powstał w wyniku analizy sposobu tworzenia obrazów przez malarzy, którzy często tworząc obraz zaczynają od namalowania odległego tła, na którym z kolei umieszczają coraz bliższe nieprzezroczyste obiekty. Każdy później umieszczony obiekt przestania obiekty wcześniej namalowane. Przykład postępowania tego typu pokazano na rysunku X.5.

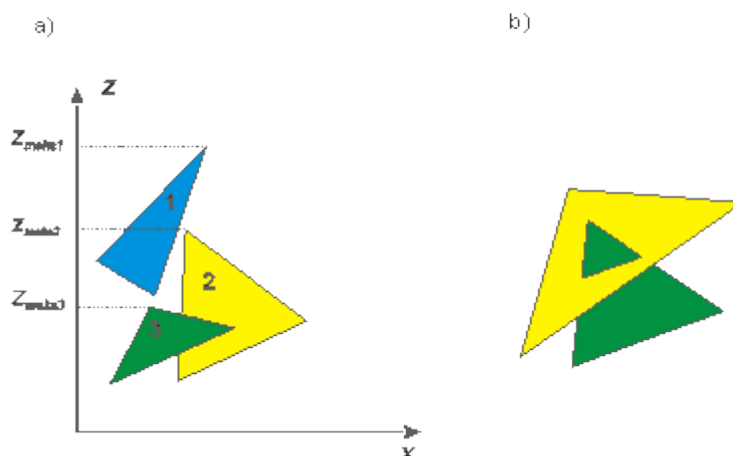


Rys. X.5. Przykład tworzenia obrazu na zasadzie rysowania obiektów w kolejności od najdalszego do najbliższego (z punktu widzenia obserwatora)

W algorytmie malarskim wykorzystywanym w grafice komputerowej wyróżnia się dwie fazy. W pierwszej fazie ustalana jest kolejność w jakiej wielokąty powinny być rysowane a w drugiej fazie ma miejsce faktyczne rysowanie. O ile sposób postępowania w drugiej fazie jest oczywisty, to pierwsza faza może stwarzać pewne kłopoty. Jedno z możliwych podejść do rozwiązania problemu może wyglądać następująco.

Celem jest posortowanie wielokątów występujących w scenie od najdalszego do najbliższego. Wstępne sortowanie można wykonać przyjmując jako podstawę największe wartości współrzędnych z w poszczególnych wielokątach (zakładamy, że współrzędna z jest skierowana od obserwatora w kierunku sceny). Ilustruje to rysunek X.6a. Uzyskana lista niestety nie zawsze daje poprawne rozwiązanie, na

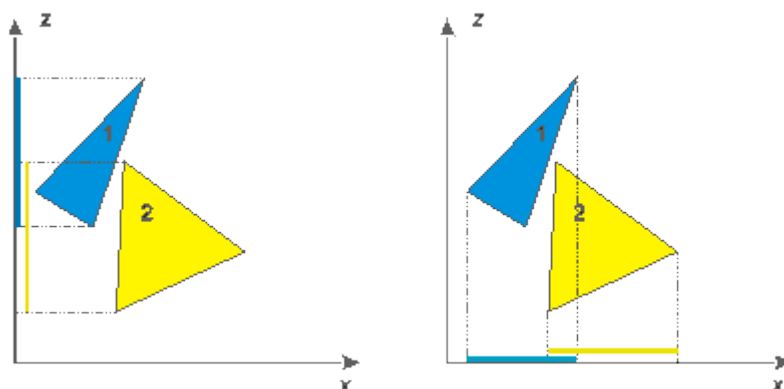
przykład ze względu na to, że niektóre wielokąty mogą się przecinać, tak jak na przykład na rysunku X.6b.



Rys. X.6. a) Ilustracja metody wstępnego sortowania wielokątów. b) Przykład przecinających się wielokątów

W związku z tym konieczne jest przeprowadzenie weryfikacji poprawności listy. Można to zrobić w następujący sposób. Bierzemy pod uwagę dwa wielokąty, które są na początku listy i sprawdzamy, czy są one we właściwej kolejności z punktu widzenia rysowania na ekranie. Jeżeli tak, to bierzemy do sprawdzenia następną parę wielokątów. Jeżeli nie, to trzeba zamienić ich kolejność. Jeżeli po zamianie okaże się, że kolejność jest poprawna to akceptujemy ją i przechodzimy do analizy następnej pary. Jeżeli jednak i tym razem kolejność nie będzie poprawna, to znaczy, że dwa rozpatrywane wielokąty przecinają się i należy jeden z nich podzielić na dwa (wzdłuż krawędzi przecięcia pierwotnych wielokątów) i umieścić je na liście. Proces należy kontynuować dopóki nie zweryfikuje się całej listy.

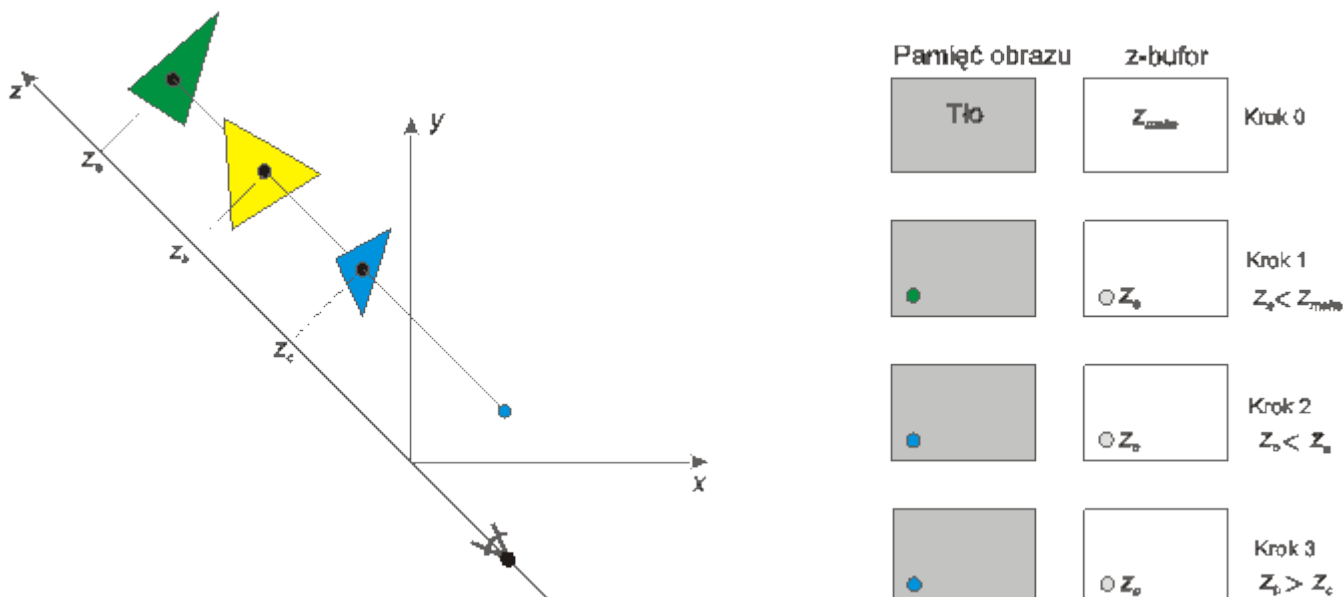
Na rysunku X.7 zilustrowano dwa przykładowe testy pozwalające podjąć decyzję o tym czy kolejność umieszczenia pary wielokątów na liście jest poprawna. W pierwszym teście sprawdza się czy zakresy zmian współrzędnej z wyznaczone dla obu wielokątów są rozłączne. Jeżeli tak, to kolejność wielokątów jest poprawna. Jeżeli nie, to należy wykonać inny test, na przykład taki jak na rysunku X.7b. Tym razem sprawdza się, czy zakresy zmian współrzędnych x dla obu wielokątów są rozłączne. Jeżeli tak, to kolejność jest poprawna. Jeżeli nie, to można wykonać kolejne testy (proponuję sformułować samodzielnie jeszcze jeden lub dwa testy).



Rys. X.7. Przykładowe testy dla sprawdzania poprawności ustawienia pary wielokątów na liście określającej kolejność wyświetlania wielokątów

Jeszcze inna koncepcja rozwiązywania problemu widoczności jest realizowana w algorytmie z-bufora. Tym razem nie przyjmuje się żadnych założeń co do rodzaju wyświetlanych obiektów ani co do kolejności wyświetlania obiektów. Widoczność obiektów rozstrzyga się dla każdego piksela obrazu niezależnie. Do realizacji algorytmu potrzebna jest dodatkowa pamięć, tak zwany z-bufor, w której można dla każdego piksela zapisać odpowiednią wartość współrzędnej z.

Na początku do pamięci obrazu zapisuje się barwę tła dla wszystkich pikseli obrazu natomiast w z-buforze zapisuje się największą wartość współrzędnej z jaka może wystąpić w scenie (zakładamy, że oś z jest skierowana od obserwatora w kierunku sceny); w szczególności może to być współrzędna z tylnej ściany bryły widzenia. Z kolei, jeżeli analizujemy jakiś wielokąt i trafimy na punkt, który potencjalnie znajdzie się w końcowym obrazie w miejscu piksela o współrzędnych x,y, to sprawdzamy, czy współrzędna z tego punktu jest mniejsza od współrzędnej z zapisanej dotychczas w z-buforze w miejscu o współrzędnych x,y. Jeżeli tak, to rozpatrywany punkt leży bliżej niż punkt sceny, którego barwa została zapisana w pamięci obrazu i zastępujemy dotychczasową barwę piksela x,y barwą rozpatrywanego punktu oraz zmieniamy dotychczasową wartość współrzędnej z wartością współrzędnej z rozpatrywanego punktu. W przeciwnej sytuacji pozostawiamy poprzednią wartość piksela x,y oraz poprzednią wartość współrzędnej z. Sposób postępowania ilustruje rysunek X.8. Proces powtarza się dla wszystkich wielokątów występujących w scenie (w bryle widzenia) i wszystkich punktów tych wielokątów, które potencjalnie mogą być odwzorowane na piksele w pamięci obrazu.

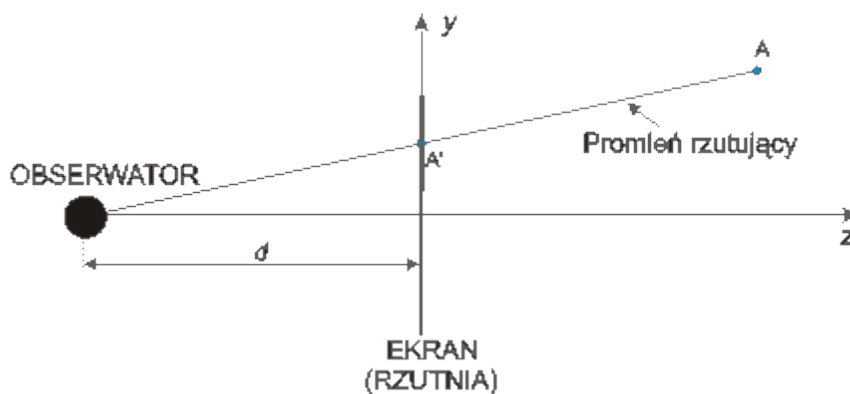


Rys. X.8. Ilustracja metody z-bufora. a) Analiza wielokątów sceny, b) zawartość pamięci obrazu i z-bufora w kolejnych krokach. Wielokąty są analizowane w kolejności zielony, niebieski, żółty

3. Rzutowanie

Zastanówmy się teraz w jaki sposób można przygotowaną scenę 3D odwzorować na płaszczyźnie. Problem ten oczywiście jest znany od dawna i wymyślono wiele różnych sposobów rzutowania. Niektóre z nich zaadaptowano dla potrzeb grafiki komputerowej. Dalej omówimy dwie najczęściej wykorzystywane metody rzutowania: rzutowanie perspektywiczne oraz rzutowanie równoległe.

W rzutowaniu perspektywicznym zakłada się, że obserwator znajduje się w pewnej odległości d przed ekranem oraz, że wszystkie promienie rzutujące przecinają się w "oku" obserwatora. Oznacza to, że na początku trzeba całą scenę przenieść do układu współrzędnych, w którym osie x, y leżą w płaszczyźnie ekranu a oś z jest do niej prostopadła. Należy również obserwatora umieścić na osi z w odległości d przed ekranem. Z kolei, dla znalezienia rzutu punktu leżącego w przestrzeni 3D na ekran, należy przez ten punkt poprowadzić promień rzutujący, który będzie przechodził przez punkt, w którym znajduje się obserwator. Punkt, w którym promień rzutujący przetnie ekran (rzutnię) jest poszukiwanym rzutem punktu z przestrzeni 3D. Poglądowo wyjaśnia to rysunek X.9.



Rys. X.9. Zasada rzutowania perspektywicznego. Rzutem punktu A jest punkt A' \blacklozenge

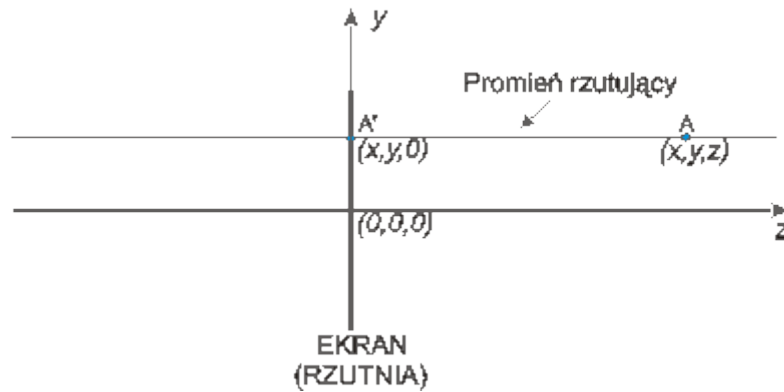
Przy obliczaniu współrzędnych rzutu punktu można korzystać z odpowiedniej macierzy we współrzędnych jednorodnych, podobnie jak w przypadku przekształceń geometrycznych. Odpowiednia macierz ma następującą postać:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix}$$

Korzystając z tej postaci macierzy dla rzutu perspektywicznego należy pamiętać, że obowiązuje ona wtedy gdy dla rzutni $z = 0$ a obserwator (środek rzutowania) jest w punkcie $z = -d$.

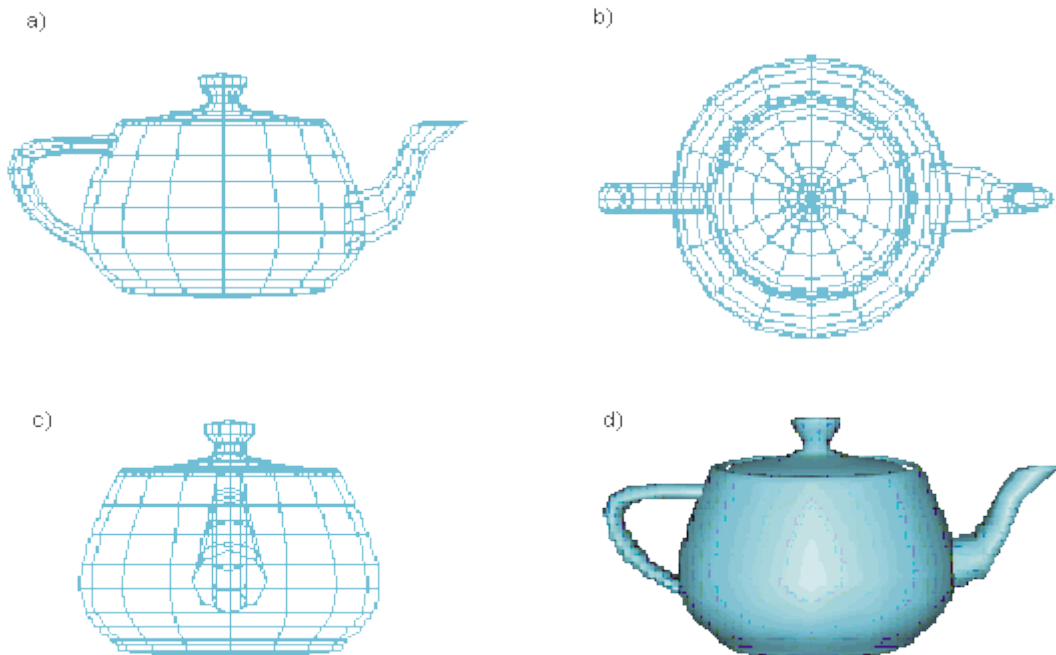
Przy rzutowaniu równoległym promienie rzutujące są do siebie równoległe. Jeżeli dodatkowo są one prostopadłe do rzutni, to mamy do czynienia z tak zwanym

rzutem równoległym prostokątnym albo inaczej z rzutem ortogonalnym. W rzucie ortogonalnym jeżeli oś z jest prostopadła do rzutni i dla rzutni $z = 0$, to po zrzutowaniu punktu $A(x,y,z)$ otrzymujemy punkt $A' (x,y,0)$. Rysunek X.10 ilustruje koncepcję rzutowania ortogonalnego.



Rys. X.10. Rzut ortogonalny. Rzutem punktu A jest punkt A'

Rzuty ortogonalne są wykorzystywane często do pokazywania trzech widoków jakiegoś obiektu: widoku z przodu, widoku z boku i widoku z góry. Na rysunku pokazano przykładowy obiekt w rzucie perspektywicznym (na dole z prawej strony) oraz trzy jego widoki ortogonalne.



Rys.IX.11. Widok perspektywiczny obiektu (d) i jego rzuty ortogonalne: a) z przodu, b) z góry c) z prawej strony

Podsumowanie

W wykładzie poznaliśmy algorytmy wykorzystywane w początkowej fazie renderingu. Algorytmy te umożliwiają pokazanie przestrzennej sceny na płaskim ekranie. W szczególności poznaliśmy metody rzutowania oraz metody eliminowania powierzchni bądź obiektów niewidocznych. W następnym wykładzie poznamy kolejne algorytmy istotne dla fazy renderingu - algorytmy umożliwiające cieniowanie powierzchni obiektów.

Przykładowe pytania i problemy do rozwiązania

1. Wyjaśnić pojęcie bryły widzenia i celowość wprowadzania takiego pojęcia.
2. Naszkicować scenę (a dokładniej jej rzut perspektywiczny) składającą się z dwóch sześciątów, przy czym z punktu widzenia obserwatora jeden sześciąt częściowo zasłania drugi.
3. Naszkicować rzuty ortogonalne sceny z poprzedniego problemu: widok z boku, widok z góry, widok z przodu.
4. Czy z punktu widzenia metody z-bufora istotne jest czy obiekty się przecinają czy nie?

Wykład 11: Cieniowanie i teksturowanie

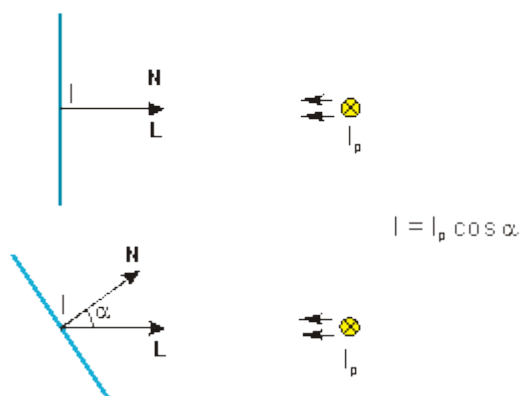
Streszczenie

Wiemy już w jaki sposób można znaleźć rzut sceny i jak rozwiązuje się problem powierzchni niewidocznych. Teraz spróbujemy zastanowić się w jaki sposób można realizować cieniowanie obiektu, a więc w jaki sposób możemy wyznaczać barwy poszczególnych pikseli reprezentujących powierzchnie boczne obiektów. Poznamy również problem teksturowania.

1. Cieniowanie

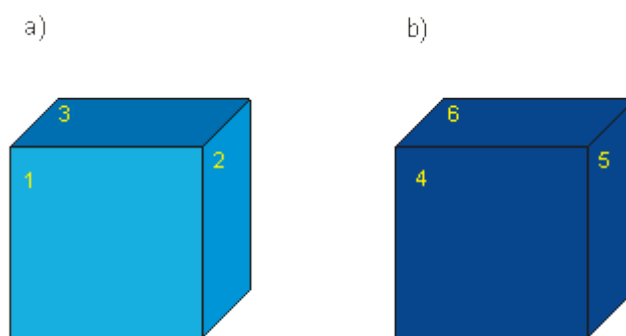
Kolejny problem, który wymaga rozwiązania w fazie renderingu polega na odpowiednim pocieniowaniu wszystkich wyświetlanych powierzchni, co faktycznie oznacza obliczenie barwy każdego piksela wyświetlanej powierzchni. Bierze się przy tym pod uwagę atrybuty wyświetlanej powierzchni (barwa oryginału, materiał powierzchni itp.) oraz sposób oświetlenia. Zależnie od potrzeb korzysta się przy tym z różnych metod.

W najprostszym przypadku możemy ograniczyć się do cieniowania płaskiego polegającego na tym, że każdy wielokąt aproksymujący powierzchnię boczną obiektu jest pokrywany jednolitą barwą. Wyjaśnijmy sposób rozwiązania problemu cieniowania na przykładzie sześcianu oświetlonego odległym źródłem światła o natężeniu I_p przy czym promienie świetlne są do siebie równoległe. W takim przypadku jasność poszczególnych ścianek zależy od kąta padania promieni na daną ściankę. Jasność ścianki wyznacza się w sposób zilustrowany na rysunku XI.1 - jasność ścianki jest równa iloczynowi natężenia źródła światła i kosinusa kąta między wektorem N normalnym do płaszczyzny ścianki a wektorem L skierowanym do źródła światła.



Rys. XI.1. Zależność wartości oświetlenia ścianki od kąta padania promieni świetlnych

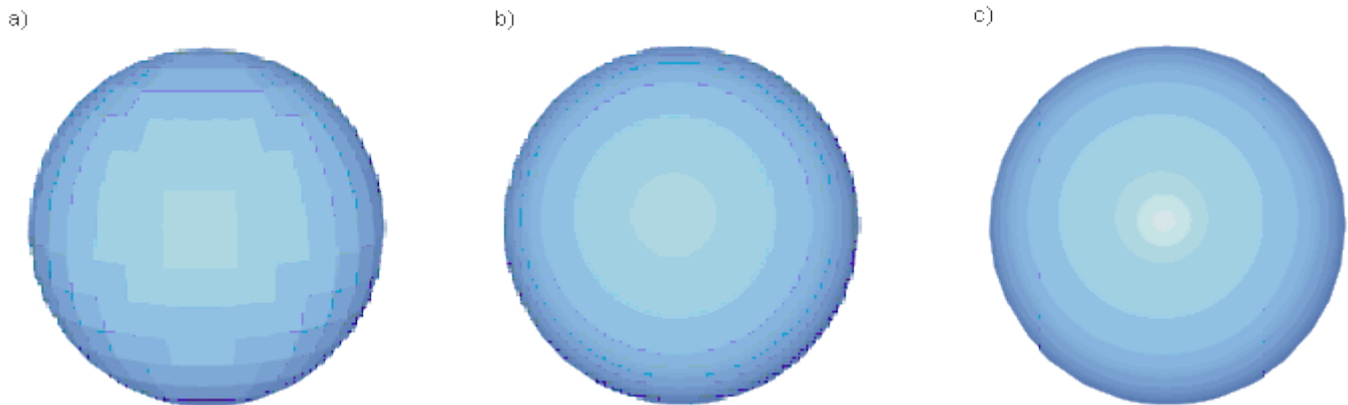
Założmy dla przykładu, że chcemy wyświetlić niebieski sześcian oświetlony odległym źródłem światła, przy czym dopuszczamy możliwość oglądania sześcianu z różnych miejsc. W tym celu w pierwszej fazie należy określić ścianki widoczne ze źródła światła i określić ich jasności. Pozostałym ściankom należy przypisać jasność wynikającą z oświetlenia przez światło otoczenia (gdyby nie uwzględnić światła otoczenia to ścianki niewidoczne ze źródła światła powinny otrzymać barwę czarną, co raczej jest niezgodne z naszą intuicją). Wartość światła otoczenia należy również uwzględnić przy wyznaczaniu jasności ścianek widocznych ze źródła światła. W drugiej fazie można określić położenie obserwatora, znaleźć ścianki widoczne z tego punktu i wyświetlić pocieniowany sześcian. Na rysunku XI.2 pokazano widok sześcianu dla dwóch położeń obserwatora.



Rys. XI.2. Sześcian oświetlony odległym źródłem światła. a) Widok dla obserwatora znajdującego się w położeniu, z którego widać bezpośrednio trzy ścianki oświetlone przez źródło światła, przy czym światło pada na ścianki 1, 2 i 3 pod różnymi kątami, b) widok dla obserwatora znajdującego się po przeciwnej stronie sześcianu niż źródło światła, w punkcie, z którego widać trzy ścianki, które nie są bezpośrednio oświetlone przez źródło światła. Barwę ścianek 4, 5 i 6 określa przyjęte światło otoczenia

Model cieniowania płaskiego jest dobry w odniesieniu do prostych obiektów, których powierzchnie boczne składają się z niewielkiej liczby wielokątów. W przypadku gdy krzywoliniowe powierzchnie boczne obiektów są aproksymowane wielokątami metoda cieniowania płaskiego nie daje dobrych rezultatów - jeżeli każdy wielokąt jest pokryty jednolitą barwą, to widoczna jest struktura

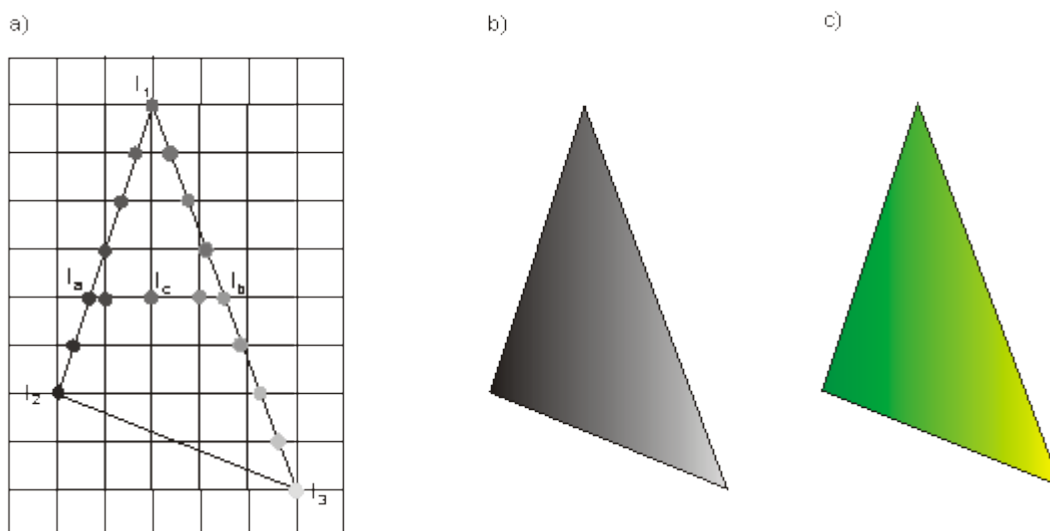
wielokątowa (por. rysunek XI.3). Efekt ten można usunąć stosując metodę cieniowania Gouraud. W metodzie tej, przy dostatecznie dużej liczbie dostępnych barw, każdy piksel wielokąta może mieć inną barwę.



Rys. XI.3. Przykłady cieniowania bryły. a) Metoda cieniowania płaskiego, b) metoda Gouraud, c) metoda Phonga (na środkowej części powierzchni kuli widoczne jest rozświetlenie)

Założmy, że powierzchnia boczna bryły jest aproksymowana trójkątami i weźmy pod uwagę jeden trójkąt. Założmy ponadto, że znamy rzut tego trójkąta na płaszczyznę ekranu oraz, że znamy barwy (wartości jasności) trzech wierzchołków trójkąta (barwy te zostały wyznaczone w przestrzeni 3D przed rzutowaniem; sposób wyznaczania barw punktów powierzchni 3D jest omawiany dalej). Zadanie polega na wyznaczeniu barwy każdego piksela należącego do trójkąta.

Weźmy pod uwagę lewą krawędź trójkąta (por. rysunek XI.4). Oznaczmy wartości jasności barw wierzchołkowych przez I_1 i I_2 . Krawędź ta jest przecinana przez linie rastra. Znając wartości I_1 i I_2 możemy dla każdego punktu przecięcia krawędzi z linią rastra znaleźć jasność pośrednią, korzystając z metody interpolacji liniowej. Podobnie, dla prawej krawędzi trójkąta o jasnościach wierzchołkowych I_1 i I_3 możemy znaleźć barwy w punktach pośrednich krawędzi. Z kolei, weźmy pod uwagę fragment wiersza rastra należący do trójkąta. Ponieważ znamy już wartości jasności na końcach tego odcinka (I_a oraz I_b na rysunku XI.4), to możemy podobnie jak poprzednio znaleźć wartości pośrednie dla poszczególnych pikseli należących do odcinka rastra (na przykład wartość I_c na rysunku). Postępując w ten sposób dla innych wierszy rastra możemy znaleźć wartość jasności dla każdego piksela należącego do wnętrza trójkąta. W efekcie można uzyskać gładkie cieniowanie powierzchni trójkąta. Przykład takiego cieniowania jest pokazany na rysunku XI.4 dla wersji czarno-białej i kolorowej.



Rys. XI.4. Metoda Gouraud. a) Ilustracja działania metody, b,c) przykłady uzyskiwanych efektów

Pewnym ograniczeniem metody Gouraud jest to, że zestaw barw jakie można uzyskać wewnątrz wypełnianego trójkąta jest ograniczony przez zestaw barw wierzchołkowych i na przykład nie można uzyskać efektu rozświetlenia wewnątrz trójkąta. Zasygnalizujemy, że problem ten pozwala rozwiązać metoda Phonga, w której prowadzi się podwójną interpolację podobnie jak w metodzie Gouraud, z tym, że dla wartości wektorów normalnych a nie dla barw. Dopiero po określeniu wektora normalnego dla konkretnego piksela wyznacza się jego barwę. Przykład efektu działania cieniowania Phonga pokazano na rysunku XI.3c.

Omówione wyżej metody określania barw poszczególnych pikseli są stosowane do wielokątów, które zostały już rzutowane na płaszczyznę ekranu. Zastanówmy się teraz jak można wyznaczać jasność dowolnego punktu powierzchni zewnętrznej bryły, a więc jeszcze przed rzutowaniem. Najczęściej korzysta się tu z równania podanego przez Phonga lub jego modyfikacji. Zakładamy, że znamy wektor normalny do powierzchni w rozważanym punkcie oraz wektor skierowany do odległego punkowego źródła światła (por. rysunek XI.5).



Rys. XI.5. Ilustracja do sposobu obliczania jasności punktu w przestrzeni 3D. a) Wpływ oświetlenia przez punktowe źródło światła, b) efekt odbicia zwierciadlanego

Phong w swoim modelu uwzględnił trzy elementy mające wpływ na jasność punktu: światło otoczenia I_a , światło pochodzące bezpośrednio ze źródła światła I_p oraz światło wynikające z odbicia zwierciadlanego od powierzchni I_s . O świetle otoczenia mówiliśmy wcześniej. Podobnie pokazywaliśmy już, że wpływ światła ze źródła światła I_p docierającego bezpośrednio do powierzchni uwzględnia się biorąc

pod uwagę kąt między promieniem światła a normalną do powierzchni (a dokładniej kosinus kąta).

W przypadku efektu odbicia zwierciadlanego bierze się pod uwagę odchylenie rzeczywistego kierunku obserwacji punktu odbicia od teoretycznego kierunku promienia odbitego (por. kąt β na rysunku XI.5b). W obliczeniach faktycznie uwzględnia się n -tą potęgę kosinusa kąta β . Wartość n dobiera się eksperymentalnie z uwzględnieniem właściwości materiału powierzchni odbijającej.

Podstawowe równanie Phong'a ma następującą postać:

$$I = k_a I_a + k_d I_p \cos \alpha + k_s I_s \cos^n \beta$$

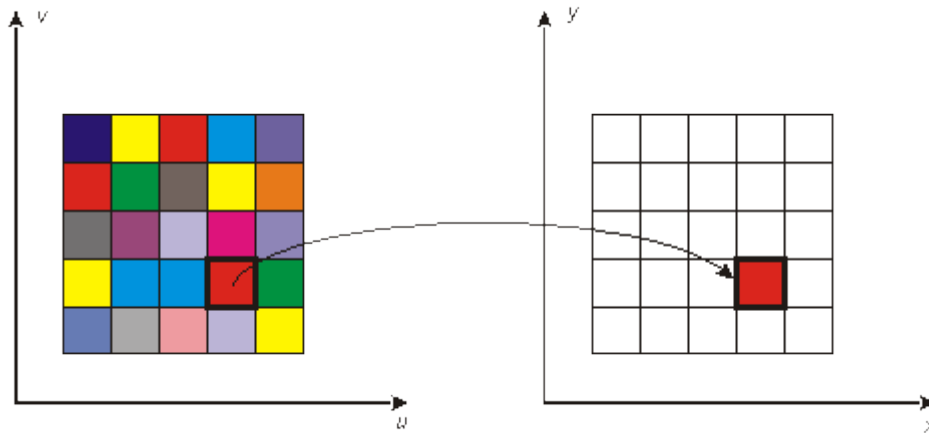
W równaniu tym współczynniki k_a , k_d oraz k_s są współczynnikami wagowymi o wartościach z przedziału od 0 do 1. Współczynniki te umożliwiają regulowanie stopnia wpływu poszczególnych źródeł światła, a pośrednio modelowanie rodzaju materiału, z którego jest wykonana powierzchnia.

W przypadku gdy w scenie występuje większa liczba źródeł światła sumuje się ich wpływ na jasność punktu (należy zwracać uwagę na to żeby nie przekroczyć dopuszczalnej wartości jasności). Jeżeli rozważany obiekt jest półprzezroczysty, to w równaniu może się pojawić czwarty czynnik umożliwiający uwzględnienie efektów załamania światła. W przypadku światel barwnych można korzystać z równania Phong'a niezależnie dla każdej składowej R, G i B.

2. Teksturowanie

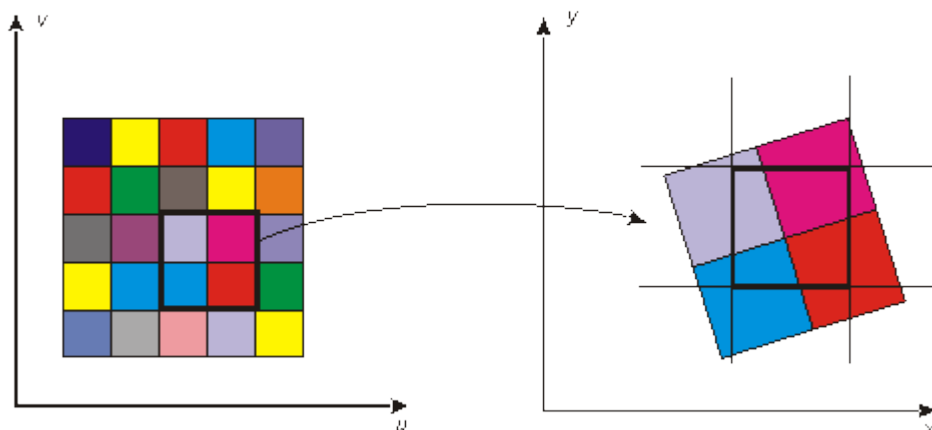
Często chcemy żeby powierzchnie zewnętrzne brył były pokryte jakimiś wzorami. Dla realizacji takich zadań w grafice najczęściej korzysta się z koncepcji teksturowania powierzchni. Ogólnie koncepcja ta polega na tym, że najpierw definiuje się odpowiedni wzór - teksturę, a następnie odwzorowuje się taki wzór na powierzchnię obiektu.

Teksturę może stanowić dowolna mapa bitowa. W szczególności może to być obraz uzyskany na przykład ze skanera albo z aparatu cyfrowego. Tekstura zwykle jest definiowana w układzie współrzędnych u, v . Elementarny fragment tekstury jest określany jako tekseł. W najprostszym przypadku, gdy obszar tekstury jest zgodny z obszarem powierzchni płaskiej, na którą należy nanieść teksturę, zadanie jest proste i sprowadza się do odwzorowania poszczególnych tekseli na odpowiednie piksele obrazu w układzie współrzędnych x, y tak jak na rysunku XI.6. Jeżeli jednak obszary tekstury i obrazu różnią się, to konieczne jest znalezienie odpowiedniej funkcji odwzorowującej.



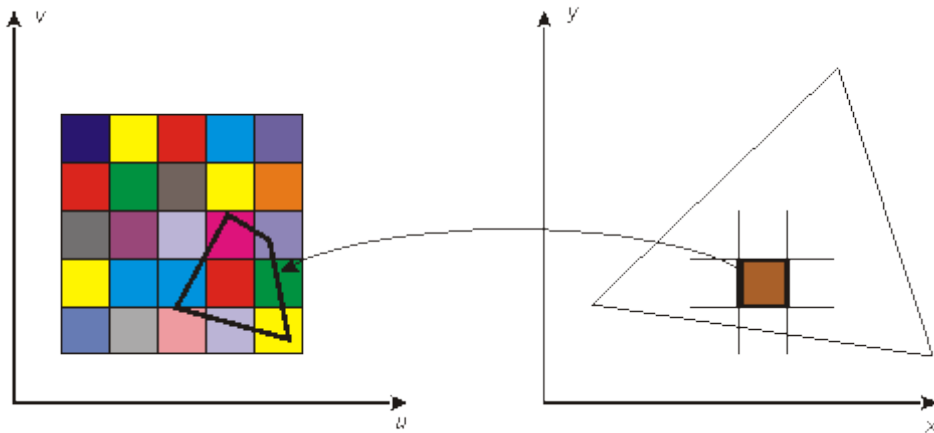
Rys. XI.6. Podstawowa koncepcja teksturowania. W najprostszym przypadku jeden tekseł jest odwzorowywany na jeden piksel

Przyjrzyjmy się problemowi odwzorowania z punktu widzenia pojedynczych pikseli. W ogólnym przypadku może wystąpić taka sytuacja, że po odwzorowaniu tekseleli na docelową powierzchnię, pojedynczy piksel zostanie przykryty przez kilka tekseleli, tak jak na przykład na rysunku. XI.7. Wtedy barwę piksela trzeba wyznaczyć biorąc pod uwagę wielkość wpływu każdego z tekseleli przykrywających piksel. Tego typu postępowanie jest określane jako odwzorowanie tekstury wprost.



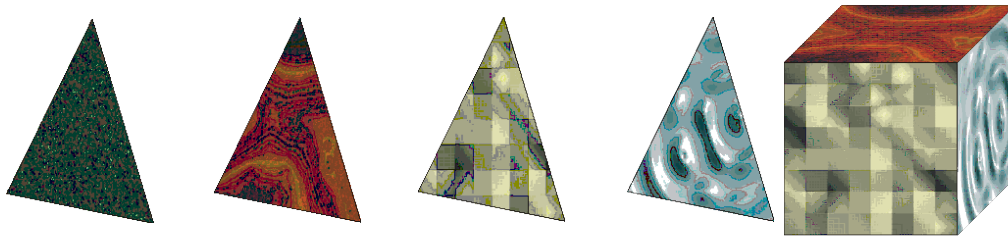
Rys. XI.7. Teksturowanie metodą odwzorowania wprost. Barwa piksela jest określona przez kilka tekseleli. Wpływ poszczególnych tekseleli zależy od stopnia pokrycia piksela przez odwzorowane teksele

Możliwe jest również stosowanie odwzorowania odwrotnego. Tym razem odwzorowuje się piksel na powierzchnię tekstury i barwę piksela określa się biorąc pod uwagę teksele, które znajdują się w obrębie obszaru reprezentującego piksel. Ilustruje to rysunek XI.8.



Rys. XI.8. Tekstutowanie metodą odwzorowania odwrotnego. Barwa piksela jest określona przez kilka tekseli. Wpływ poszczególnych tekseli zależy od stopnia pokrycia odwzorowanego piksela przez tekselę

Na rysunku XI.9 pokazano kilka przykładów tekstutowania.



Rys. XI.9. Przykłady tekstutowania

Podsumowanie

W tym wykładzie poznaliśmy różne metody cieniowania. Poznaliśmy prostą metodę cieniowania ciągłego, metody cieniowania Gouraud i Phonga. Pokazaliśmy również w jaki sposób, korzystając z równania Phonga, można wyznaczać barwę punktu powierzchni przed rzutowaniem na płaszczyznę - przyda nam się to w następnym wykładzie. Poznaliśmy także koncepcję tekstutowania.

Przykładowe pytania i problemy do rozwiązania

1. Wyjaśnić na czym polega różnica między cieniowaniem płaskim a cieniowaniem metodą Gouraud. Co można powiedzieć o złożoności obliczeniowej obu metod?
2. Podać jakie operacje trzeba wykonać, żeby uzyskać na ekranie obraz zielonej kostki sześciennej oświetlonej odległym źródłem światła (promienie światła są do siebie równoległe). Obserwator może znajdować się w dowolnym punkcie poza sześcianem.
3. Zwrócić uwagę na dwa różne problemy, których rozwiązanie zaproponował Phong: cieniowanie powierzchni wielokąta po rzutowaniu na płaszczyznę metodą Phonga oraz wyznaczanie barwy punktu powierzchni 3D za pomocą równania Phonga.
4. Porównać metody odwzorowania tekstury: wprost i odwrotną.

Wykład 12: Grafika 3D. Metoda śledzenia promieni

Streszczenie

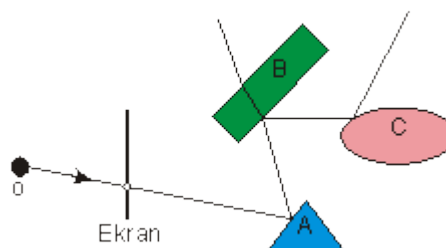
Każdy z algorytmów poznanych w poprzednich dwóch wykładach realizuje pewien etap obliczeń realizowanych w fazie renderingu. Metody te umożliwiają wyświetlanie scen lepiej lub gorzej odzwierciedlających rzeczywistość, ale za to stosunkowo szybko. W tym wykładzie poznamy metodę, która umożliwia generowanie obrazów o bardzo dobrej jakości jednak kosztem dużej złożoności obliczeniowej. Jest to tak zwana metoda śledzenia promieni.

1. Koncepcja metody śledzenia promieni

W metodzie śledzenia promieni przyjęta jest inna koncepcja realizacji procesu renderingu sceny niż poprzednio, gdzie cały proces renderingu składał się z szeregu kroków realizujących poszczególne operacje w odniesieniu do poszczególnych obiektów sceny: rzutowanie z uwzględnieniem bryły widzenia, eliminowanie elementów niewidocznych, cieniowanie itd. Tym razem pomysł polega na tym żeby obliczenia prowadzić niezależnie w odniesieniu do każdego piksela tworzonego obrazu.

Istota metody polega na tym, że z punktu obserwacji (oka obserwatora) prowadzi się promienie przechodzące przez poszczególne piksele ekranu i biegnące w stronę sceny. Analizowany jest bieg każdego promienia w scenie i następnie określa się barwę przypisywaną odpowiedniemu pikselowi. W ten sposób kompletuje się informację o barwie wszystkich pikseli tworzących obraz.

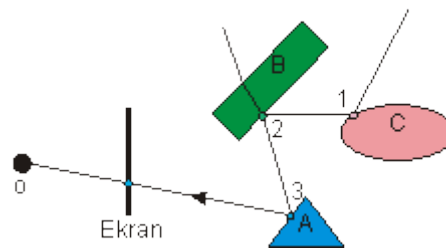
Rozważmy proces analizy dla jednego piksela. Na rysunku XII.1 pokazano przykładowy promień biegnący w stronę sceny. Promień ten po napotkaniu pierwszego obiektu A odbija się i biegnie do momentu napotkania kolejnego obiektu B. Na rysunku XII.1 założono, że obiekt B jest półprzezroczysty i promień ulega zarówno odbiciu jak i załamaniu. Promień odbity z kolei biegnie do najbliższego obiektu C gdzie ponownie jest odbijany. Proces śledzenia można kontynuować dalej. W praktyce proces śledzenia ogranicza się do dwóch lub trzech kolejnych odbić.



Rys. XII.1. Przykładowy bieg promienia w metodzie śledzenia promieni

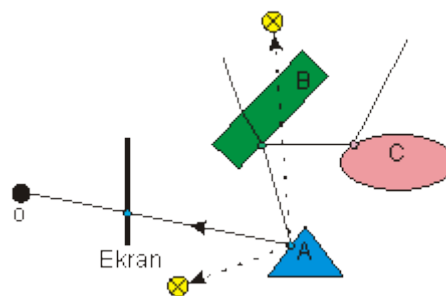
Po znalezieniu drogi promienia w scenie, w drugiej fazie wyznacza się barwę jaką należy przypisać pikselowi. Niewątpliwie piksel powinien uzyskać barwę jaką

będzie widział obserwator patrząc przez umowny otwór w ekranie w miejscu gdzie jest piksel. W przykładzie z rysunku XII.1 obserwator będzie widział punkt powierzchni obiektu A i barwa tego punktu powinna być przypisana pikselowi. Powstaje więc problem określenia barwy tego punktu. Można to zrobić korzystając z równania Phong'a. Trzeba tu wziąć pod uwagę fakt, iż barwa punktu będzie uzależniona zarówno od bezpośredniego oświetlenia przez źródła światła znajdujące się w scenie jak też i oświetlenia pośredniego pochodzącego od sąsiednich obiektów. W przykładzie z rysunku XII.1 oznacza to, że powinniśmy najpierw wyznaczyć barwę punktu na powierzchni obiektu C, potem barwę punktu na powierzchni B (z uwzględnieniem wpływu promienia załamane) i dopiero na końcu barwę punktu na powierzchni A (por. rysunek XII.2).



Rys. XII.2. Wyznaczanie barwy piksela w metodzie śledzenia promieni

Przy wyznaczaniu barw poszczególnych punktów istotne jest również uwzględnienie tego czy dany punkt jest oświetlany przez określone źródło światła czy też leży w cieniu rzucanym przez inny obiekt. W tym celu korzysta się z pomocniczych promieni (tak zwanych promieni cieni) prowadzonych z analizowanego punktu w kierunku każdego źródła światła. Ilustruje to rysunek XII.3.



Rys. XII.3. Promienie cieni w metodzie śledzenia promieni

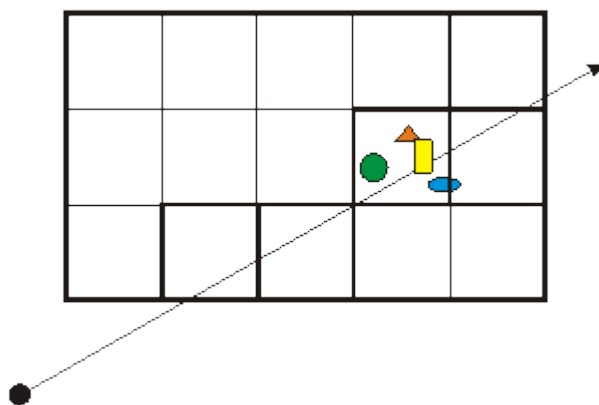
Zauważmy, że metoda śledzenia promieni rozwiązuje wiele problemów, o których mówiliśmy wcześniej: problem rzutowania z uwzględnieniem bryły widzenia, problem widoczności, problem cieniowania czy wreszcie problem cieni rzucanych przez obiekty. Jest to metoda bardzo efektywna jeśli chodzi o jakość uzyskiwanych obrazów. Obrazy uzyskiwane za pomocą tej metody są określane mianem fotorealistycznych. Ich jakość niejednokrotnie nie odbiega od jakości dobrych zdjęć.

2. Sposoby przyspieszania obliczeń

Jednak istotnym ograniczeniem metody śledzenia promieni jest jej duża złożoność obliczeniowa. Liczba promieni wymagających analizy jest równa liczbie pikseli, co przy rozdzielczości ekranu na przykład 1024x768 oznacza konieczność prześledzenia prawie 800 tysięcy promieni. Szukając sposobów przyspieszenia działania metody zbadano, które obliczenia związane z realizacją metody są najbardziej pracochłonne. Okazało się, że większość czasu zajmują obliczenia związane z wyznaczaniem przecięć promieni z obiektami i poszukiwaniem pierwszego napotkanego obiektu. Wiedząc o tym opracowano kilka sposobów zmniejszania nakładów obliczeniowych.

Jeden z najczęściej stosowanych sposobów polega na tym, że złożone obiekty występujące w scenie wstępnie otacza się bryłami, dla których łatwo sprawdza się przecięcie z promieniem. Są to najczęściej kule albo sześciany. Jeżeli trzeba sprawdzić czy promień przecina dany obiekt, to najpierw sprawdza się czy przecina on otaczającą kulę. Jeżeli nie to nie podejmuje się próby przecinania promienia z obiektem. Dopiero jeżeli promień przetnie kulę to wchodzi się w pracochłonną procedurę znalezienia przecięcia promienia z obiektem (oczywiście mimo wszystko wynik może być negatywny).

Inny sposób polega na tym, że całą scenę otacza się prostopadłościanem. Następnie prostopadłościan jest dzielony na pewną liczbę mniejszych prostopadłościanów. Z kolei dla każdego takiego mniejszego prostopadłościanu określa się listę obiektów sceny, które w całości lub częściowo należą do niego. Analizując teraz bieg promienia można znaleźć te prostopadłościany, przez które przechodzi promień i w dalszych obliczeniach można się ograniczyć tylko do obiektów, które są związane z wybranymi prostopadłościanami. Z pozostałymi obiektami promień na pewno nie przetnie się. Metodę tę poglądowo wyjaśnia rysunek XII.4, na którym dla uproszczenia pokazano przypadek dwuwymiarowy.



Rys. XII.4. Poglądowe wyjaśnienie metody podziału przestrzeni sceny na prostopadłościany (widok z góry)

Dzięki zastosowaniu różnych metod przyspieszania obliczeń oraz dzięki ogromnemu wzrostowi szybkości sprzętu obliczeniowego metoda śledzenia promieni jest już dostępna na sprzęcie powszechnego użytku i obrazy dobrej jakości można uzyskiwać w przeciągu kilku czy kilkunastu minut. Należy jednak pamiętać, że jest to jednak metoda bardzo czasochłonna i powinna być stosowana wtedy, gdy zależy nam na obrazach o jakości fotorealistycznej. W innych przypadkach należy

korzystać z metod prostszych, takich jakie były omawiane na wcześniejszych wykładach.

Podsumowanie

Metoda śledzenia promieni omówiona w tym wykładzie jest jedną z najlepszych znanych metod tworzenia obrazów fotorealistycznych. Niestety wciąż czasy obliczeń są zbyt długie by można było korzystać z tej metody "na co dzień". Ale po pierwsze trzeba pamiętać, że nie zawsze jest nam potrzebna jakość fotorealistyczna, a po drugie przyrost mocy obliczeniowej jest tak szybki, że być może sytuacja niedługo ulegnie radykalnej zmianie - jeszcze nie tak dawno tych, którzy proponowali stosowanie metody śledzenia promieni uważano za fantastów.

Przykładowe pytania i problemy do rozwiązania

1. Czy korzystając z metody śledzenia promieni musimy wyznaczać bryłę widzenia?
2. Czy metoda śledzenia promieni rozwiązuje problem cieniowania? A jeżeli tak, to w jaki sposób?
3. W jaki sposób można znaleźć przecięcie promienia z kulą? (Zagadnienie to nie było omawiane w wykładzie, ale po przypomnieniu sobie wykładów z matematyki nie powinno nastręczać kłopotów.)
4. Zastanów się na tym w jaki sposób można rozwiązać problem znajdowania pierwszej napotkanej bryły na drodze promienia biegnącego w stronę sceny zawierającej kilkadziesiąt brył.

Wykład 13: Animacja

Streszczenie

Jednym z najważniejszych zastosowań grafiki komputerowej jest przemysł filmowy produkujący filmy animowane. Tutaj konieczne jest generowanie i wyświetlanie całych sekwencji obrazów. W trakcie wykładu poznamy pewne metody wykorzystywane do tworzenia animacji.

1. Podstawy animacji

Dotychczas zajmowaliśmy się tworzeniem pojedynczego - statycznego obrazu. Jednak otaczająca nas rzeczywistość nie jest statyczna i metody grafiki komputerowej muszą umożliwić również rozwiązanie problemu wizualizacji zmian zachodzących w funkcji czasu. Zajmuje się tym animacja komputerowa. W animacji generowane są zestawy klatek (obrazów, ramek) odzwierciedlających wygląd sceny w kolejnych, dyskretnych chwilach. Z kolei, dzięki pewnej bezwładności naszego systemu wzrokowego, przy dostatecznie szybkim wyświetlaniu kolejnych klatek uzyskuje się wrażenie ciągłości zmian zachodzących w funkcji czasu.

Warto tu od razu zaznaczyć, że animacja nie ogranicza się jedynie do ruchu - każda zmiana występująca między kolejnymi klatkami jest pewną formą animacji. Tak więc na przykład, zmiana kształtu obiektu czy zmiana wyglądu obiektu to też są

pewne formy animacji. Zauważmy również, że przy wyświetlaniu sekwencji obrazów, kolejne obrazy można traktować jako próbki rzeczywistej ciągłej sekwencji zdarzeń, pobierane w określonych odstępach czasu.

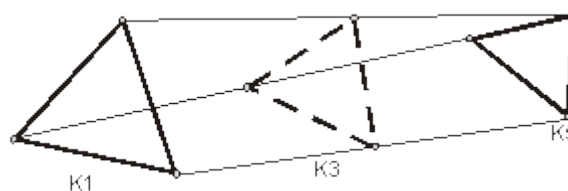
Przy wyświetlaniu sekwencji obrazów możemy spotkać się z dwoma parametrami: z częstotliwością odświeżania i z częstotliwością uaktualniania. Ten pierwszy parametr określa z jaką częstotliwością są wyświetlane kolejne klatki. Natomiast drugi parametr mówi o tym jak często są tworzone nowe obrazy. Wartości obu tych parametrów nie muszą być takie same.

Częstotliwość wyświetlania, tak jak to było już omawiane wcześniej, powinna być na tyle duża żeby nie pojawiał się efekt migotania. Natomiast częstotliwość tworzenia nowych klatek powinna być taka żeby zapewnić wrażenie ciągłości zmian zachodzących w scenie. Okazuje się, że przeciętny obserwator akceptuje znacznie mniejsze częstotliwości wyświetlania kolejnych obrazów niż to ma miejsce w przypadku efektu migotania. Akceptowalny ruch można uzyskać już przy wyświetlaniu kilku czy kilkunastu obrazów na sekundę. Przypomnijmy, że w kinie wyświetla się 24 klatki na sekundę. Stąd też, jeżeli system nie jest w stanie obliczać nowych obrazów tak szybko, żeby można je było wyświetlać z przyjętą częstotliwością odświeżania ekranu stosuje się rozwiązanie polegające na tym, że jeden obraz jest wyświetlany na przykład w dwóch albo w trzech kolejnych ramkach. Inne rozwiązanie polega na tym, że sekwencja klatek tworzących animację jest tworzona off-line i można ją wyświetlać z wymaganą częstotliwością.

Rozróżnia się animację wspomaganą komputerem i animację tworzoną komputerowo. W tym pierwszym przypadku chodzi o komputerową realizację klasycznego procesu animacji. W drugim przypadku animator pracuje w syntetycznym trójwymiarowym środowisku i określa ruch zarówno obserwatora (kamery) jak i obiektów 3D.

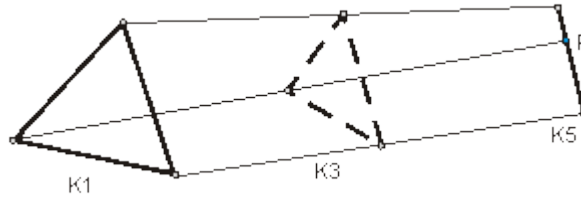
Wiele początkowych systemów animacji komputerowej działało na podobnej zasadzie jak w klasycznej animacji, kiedy to główny animator rysował ramki kluczowe dla animowanej sekwencji a asystenci rysowali na tej podstawie ramki pośrednie. W animacji komputerowej ramki pośrednie są wyznaczane automatycznie za pomocą odpowiedniej procedury, najczęściej interpolacji.

Na rysunku XIII.1 pokazano przykład prostej interpolacji między dwoma figurami (trójkątami). Najpierw łączy się odcinkami odpowiadające sobie punkty (wierzchołki) obu figur. Następnie wyznacza się metodą interpolacji położenia punktów dla klatek pośrednich. Dla uproszczenia, na rysunku XIII.1 pokazano tylko jedną klatkę pośrednią K3 leżącą w połowie drogi między klatkami K1 i K5.



Rys. XIII.1. Przykład wyznaczania klatki pośredniej

Problem trochę komplikuje się kiedy nie ma pełnej odpowiedniości między punktami w obu wyjściowych ramkach kluczowych. Wtedy trzeba wprowadzić punkty pomocnicze, tak żeby uzyskać pełny zestaw par punktów. Przykład postępowania jest pokazany na rysunku XIII.2.



Rys. XIII.2. Przykład wprowadzenie pomocniczego punktu P dla potrzeb wyznaczenia klatki pośredniej

W powyższych przykładach pokazano najprostszy przypadek, kiedy poszczególne punkty przemieszczają się po odcinkach i w dodatku ze stałą prędkością, co w efekcie pozwalało stosować zwykłą interpolację liniową pomiędzy poszczególnymi parami punktów. W bardziej złożonych problemach poszczególne punkty mogą się przemieszczać po dowolnych liniach (ścieżkach), i w dodatku różnych dla poszczególnych punktów. Zmienna może być również prędkość przemieszczania się punktów po tych ścieżkach. Na przykład, można określić, że obiekt, który w ramce K1 jest w jakimś położeniu, powinien do ramki K10 przyspieszać, potem utrzymywać stałą prędkość do ramki K20 a następnie zwalniać do ramki K30 kiedy znajdzie się w końcowym położeniu. Możliwe jest również wprowadzanie przestojów w poruszaniu się po krzywej.

W takich przypadkach określenie parametrów dla klatek pośrednich komplikuje się i z reguły wymaga stosowania odpowiednich narzędzi (programów) wspomagających, które pozwalają na przykład opisywać zmiany parametrów proceduralnie albo za pomocą określania odpowiednich reguł i ograniczeń. Zmiany te, podkreślmy raz jeszcze, mogą dotyczyć nie tylko zmian położenia ale również zmian kształtu czy wyglądu w funkcji czasu.

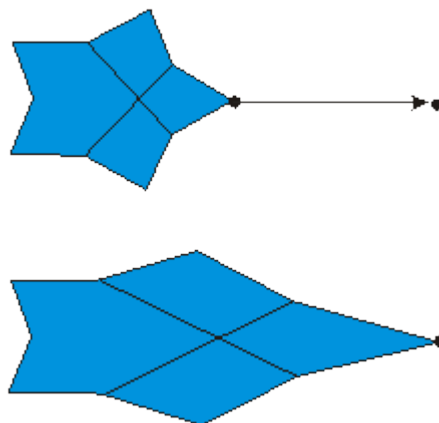
W animacji często istotne jest również określenie orientacji obiektu. Dotyczy to na przykład sytuacji kiedy obserwator (kamera) porusza się po pewnym torze krzywoliniowym i należy zapewnić to, żeby kierunek patrzenia był zawsze zgodny z kierunkiem przemieszczania się. W innej sytuacji może być istotne, żeby kierunek patrzenia obserwatora poruszającego się po określonym torze zawsze przechodził przez ustalony punkt.

Animacji mogą podlegać obiekty, których zachowanie musi być zgodne z prawami fizyki. Naturalnie w takich przypadkach należy korzystać z praw fizyki. Niemniej, niejednokrotnie wiąże się to ze złożonymi i czasochłonnymi obliczeniami. Dlatego też często korzysta się z obliczeń przybliżonych, kierując się zasadą, dość powszechnie stosowaną w grafice komputerowej, że liczy się końcowy efekt wizualny, natomiast to jak został on osiągnięty nie jest sprawą krytyczną. Oczywiście, mimo stosowania uproszczonych obliczeń, należy przestrzegać zasady, że efekt końcowy powinien być zgodny z intuicją obserwatora, a więc niesprzeczny z tym co spotyka się w rzeczywistości.

2. Warping i morfing

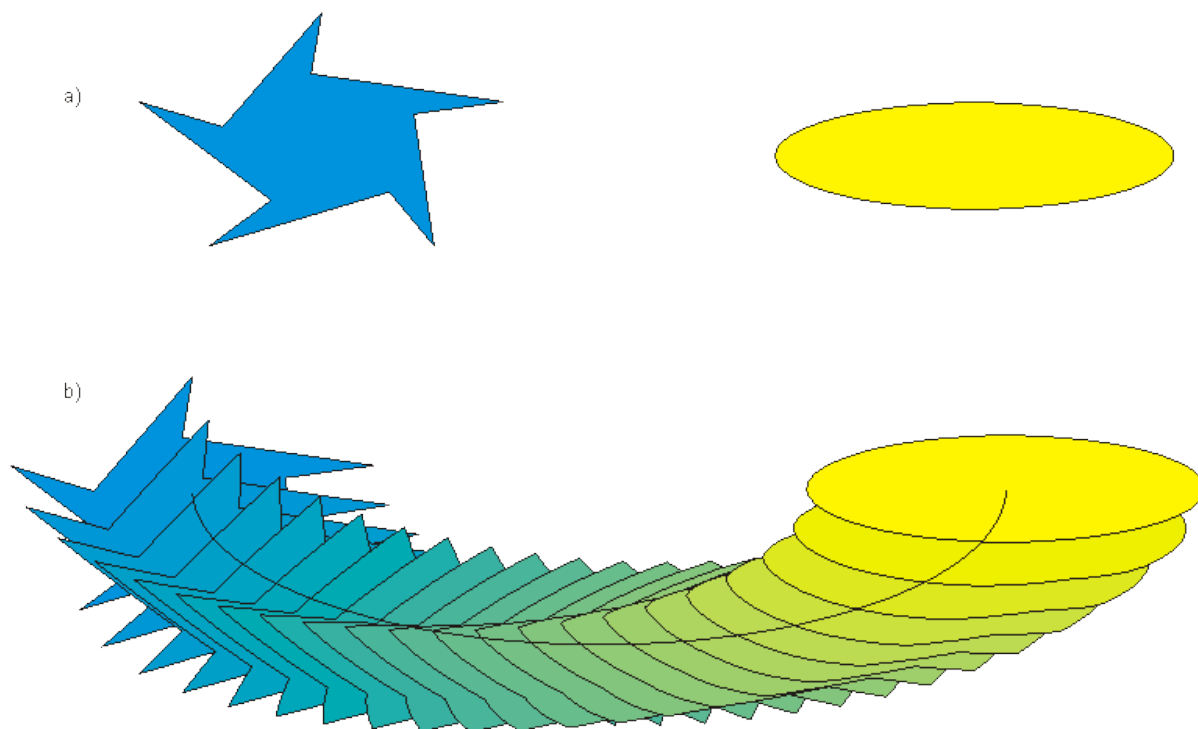
Jedną z technik animacji jest deformowanie (modyfikowanie) obiektów polegające na zmianie ich kształtów (używa się tu określenia warping). W najprostszym przypadku można tu wykorzystać odpowiednie transformacje w odniesieniu do wierzchołków figur czy brył albo w odniesieniu do punktów sterujących krzywymi (na przykład krzywymi Béziera) czy powierzchni krzywoliniowych.

Prosty sposób modyfikowania kształtu obiektu polega na przesunięciu jednego wierzchołka obiektu i propagowaniu przesunięcia sąsiednich wierzchołków po powierzchni z uwzględnieniem stopniowego zmniejszania przesunięcia wierzchołków. Przykład takiego postępowania pokazano na rysunku XIII.3.



Rys. XIII.3. Przykład modyfikowania kształtu obiektu

W powyższym przykładzie mieliśmy do czynienia cały czas z tym samym obiektem - zmieniał się tylko jego kształt. W innym rodzaju animacji możemy spotkać się z przejściem od jednego obiektu do innego obiektu. Mówimy wtedy o operacji morfingu albo o metamorfozie. Na rysunku XIII.4 pokazano prosty przykład morfingu, w którym ma miejsce stopniowe przejście od figury niebieskiej do figury żółtej wzdłuż zadanej krzywej.



Rys. XIII.4. Przykład morfingu między obiektami pokazanymi na rysunku a) wzdłuż zadanej krzywej b)

Istniejące programy graficzne umożliwiają uzyskiwanie znacznie bardziej złożonych animacji niż to co zostało pokazane powyżej. Dla przykładu możemy obejrzeć animację (rysunek XIII.5) wykonaną za pomocą programu 3D Studio Max, z którym zetkniemy się w czasie zajęć laboratoryjnych. Przedstawiona animacja zawiera 100 klatek. W celu obejrzenia tej przykładowej animacji należy użyć myszki.

Podsumowanie

Wykład poświęcony był wyjaśnieniu na czym polega animacja i zasygnalizowaniu gdzie i w jakim zakresie mogą być wykorzystywane komputery do tworzenia animacji. Zwrócono również uwagę na niektóre problemy pojawiające się przy tworzeniu animacji. Dodajmy jeszcze, że na ogół obrazom animowanym towarzyszą dźwięki. Ten obszar zagadnień jednak wykracza poza ramy kursu.

Przykładowe pytania i problemy do rozwiązania

1. Wymienić różne efekty, które mogą występować w ramach animacji.
2. Proszę wyjaśnić zasadę tworzenia klatek pośrednich.
3. W ramce kluczowej k punkt A ma współrzędne $(2,3)$. Ten sam punkt A w ramce kluczowej $k+1$ ma współrzędne $(24,27)$. Załóżmy, że między ramkami kluczowymi ma być 10 ramek pośrednich. Punkt A porusza się ruchem jednostajnym po linii prostej. Proszę podać współrzędne punktu A dla czwartej ramki pośredniej.
4. Proszę wyjaśnić różnicę między warpingiem a morfingiem.

Wykład 14: Przetwarzanie obrazów

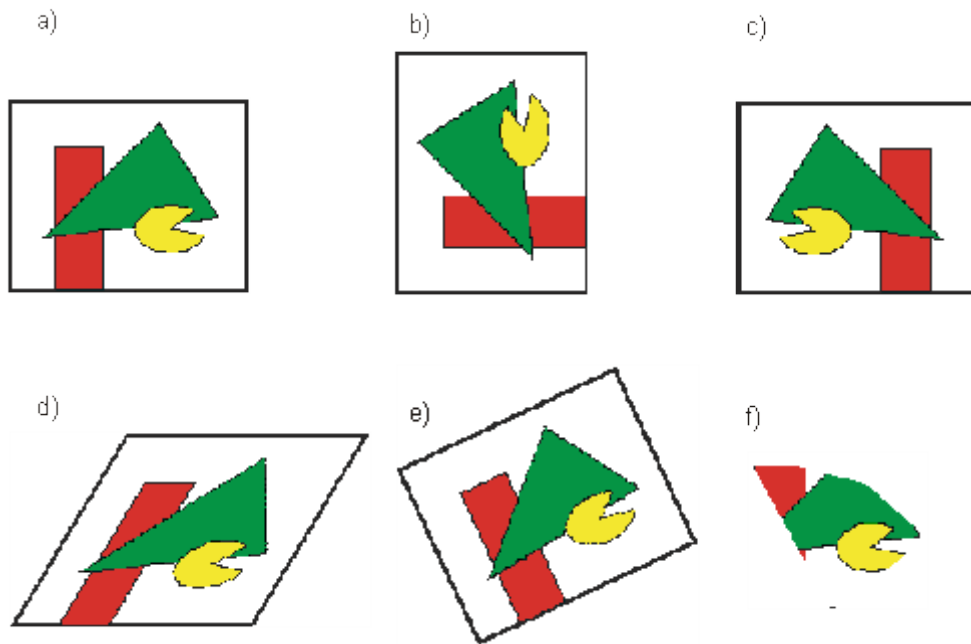
1. Wprowadzenie

O ile grafika komputerowa zajmuje się tworzeniem nowych obrazów, to w przetwarzaniu obrazów zakłada się, że jest gotowy obraz w postaci mapy bitowej i chodzi o przekształcenie tego obrazu do innej postaci. Pozwala to niejednokrotnie odpowiednio zmodyfikować bądź ulepszyć obraz uzyskany metodami grafiki komputerowej. Ponadto coraz częściej spotykamy się z metodami przetwarzania obrazów w codziennej praktyce, na przykład przy obróbce zdjęć cyfrowych czy przy korygowaniu obrazów uzyskanych za pomocą skanera. Stąd istotne jest poznanie, przynajmniej w pewnym zakresie, wybranych metod przetwarzania obrazów. Zagadnieniu temu jest poświęcony niniejszy wykład.

W przetwarzaniu obrazów punktem wyjścia jest istniejący obraz źródłowy w postaci mapy bitowej. Celem jest uzyskanie innej wersji tego obrazu, z jakichś względów lepszej z punktu widzenia użytkownika. Znanych jest wiele różnych metod przetwarzania obrazów. Ogólnie można wyróżnić wśród nich kilka podstawowych grup. Są to: przekształcenia geometryczne, przekształcenia punktowe, przekształcenia z wykorzystaniem filtrów, przekształcenia widmowe, przekształcenia morfologiczne. Tutaj ograniczymy się do scharakteryzowania trzech pierwszych grup.

2. Przekształcenia geometryczne

Przekształcenia geometryczne dotyczą z reguły całych obrazów. Typowe przekształcenia to przesunięcia, obroty, odbicia. Zaliczymy tu również operacje kadrowania pozwalające wybierać z obrazu wybrany fragment. Na rysunku pokazano przykłady działania takich operacji. Często operacje mogą być wykonywane w odniesieniu do wybranych fragmentów obrazu. Do wskazywania odpowiednich fragmentów służą różnego rodzaju maski.



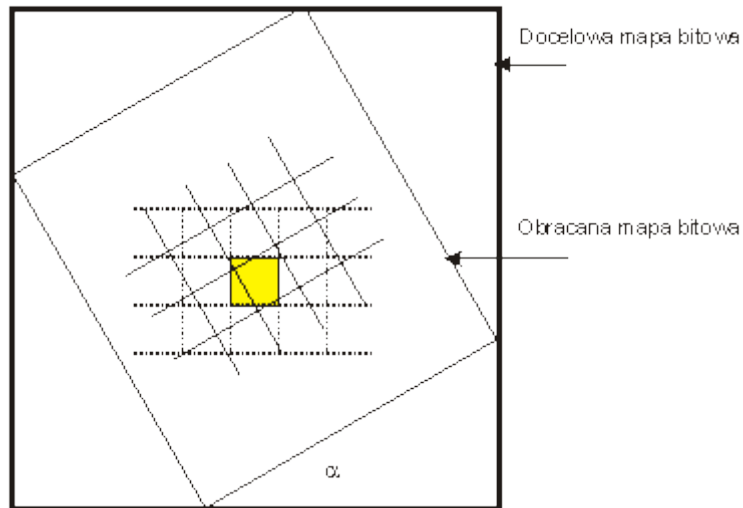
Rys. XIV.1. Przykłady operacji na całej mapie bitowej. a) Oryginał, b) obrót o 90° , c) odbijanie w poziomie, d) pochylanie w kierunku osi x, e) obrót o kąt α , f) kadrowanie

Niektóre z operacji wykonywanych w odniesieniu do całego obrazu (bądź wybranego fragmentu) są łatwe algorytmicznie, inne są bardziej skomplikowane. Proponuję zastanowić się chwilę nad realizacją operacji obracania obrazu o 90° przeciwnie do ruchu wskazówek zegara, tak jak na rysunku XIV.1b. Można chyba zauważyć, że operacja taka może być zrealizowana w następujący sposób: zamieniamy kolejność pikseli w każdym wierszu a następnie zamieniamy miejscami wiersze i kolumny (por. rysunek XIV.2).

$$\begin{bmatrix} 1 & 2 & 9 \\ 3 & 4 & 10 \\ 5 & 6 & 11 \\ 7 & 8 & 12 \end{bmatrix} \begin{bmatrix} 9 & 10 & 11 & 12 \\ 2 & 4 & 6 & 8 \\ 1 & 3 & 5 & 7 \end{bmatrix}$$

Rys. XIV.2. Obrót macierzy pikseli o 90° przeciwnie do kierunku ruchu wskazówek zegara

Obrót mapy bitowej o inny kąt niż 90° albo 180° jest bardziej złożony. Ogólną procedurę ilustruje rysunek XIV.3. Najpierw dokonujemy obrotu macierzy pikseli o zadany kąt. Następnie określamy docelową siatkę pikseli i dla każdego docelowego piksela znajdujemy te piksele pierwotnej macierzy, które są pokryte przez rozpatrywany piksel docelowy. Z kolei wyznaczamy stopień pokrycia każdego przykrytego piksela. Ten stopień pokrycia określa z jaką wagą należy wziąć intensywność przykrytego piksela przy obliczaniu wypadkowej barwy docelowego piksela.

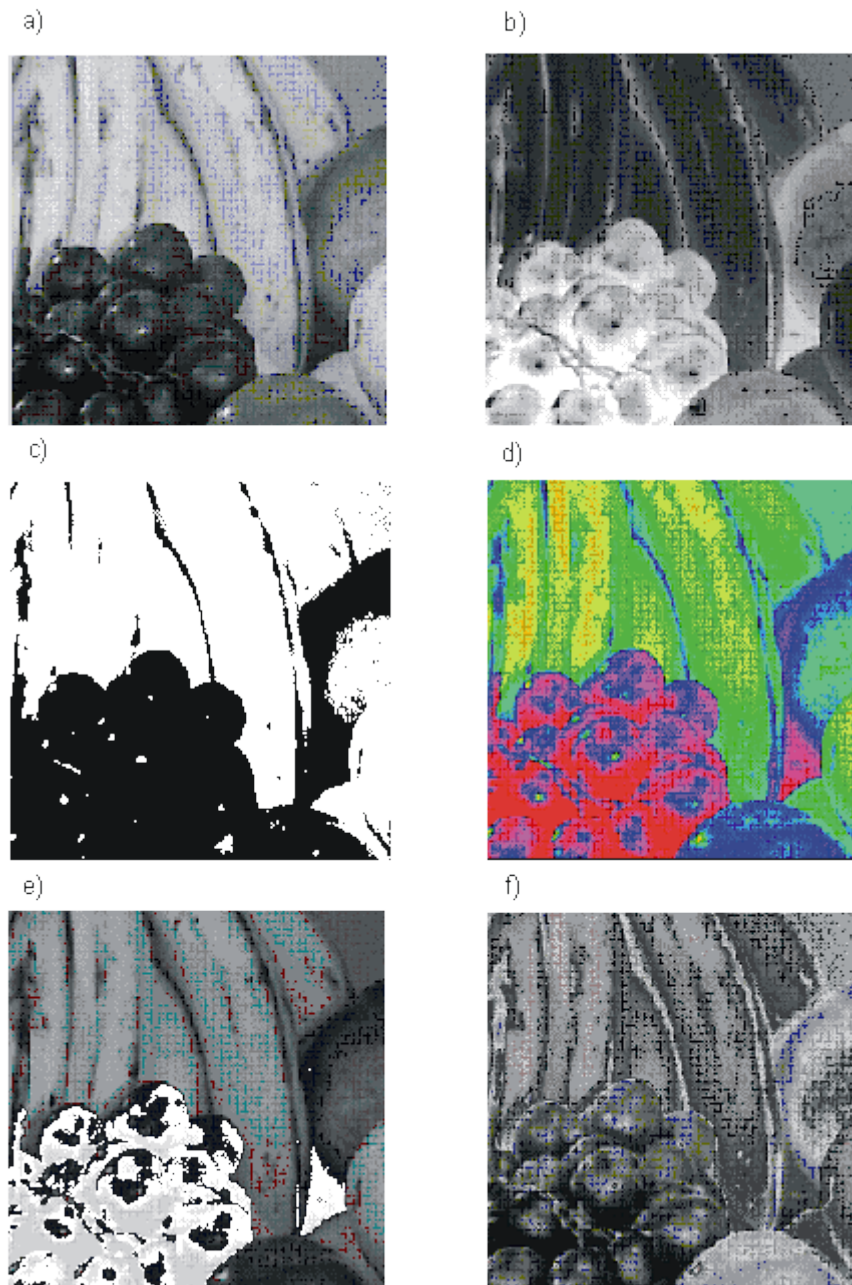


Rys. XIV.3. Obrót mapy bitowej o kąt α

Zwróćmy uwagę, że przy przekształceniach tego typu, jak na przykład obroty mapy bitowej, musimy wykonać operacje w odniesieniu do poszczególnych pikseli obrazu początkowego, podczas gdy w przypadku opisu wektorowego obrazu dokonywaliśmy przekształceń w odniesieniu do poszczególnych obiektów, a w szczególności do wierzchołków obiektów bądź punktów sterujących.

3. Przekształcenia punktowe

Przekształcenia punktowe obejmują operacje dotyczące poszczególnych pikseli obrazu niezależnie od stanu pikseli sąsiednich. Przykłady takich operacji to tworzenie negatywu obrazu, zmiana obrazu przy wykorzystaniu odpowiedniej charakterystyki przejściowej, progowanie (binaryzacja), pseudokolorowanie. Przykłady działania operacji pokazano na rysunku XIV.4.



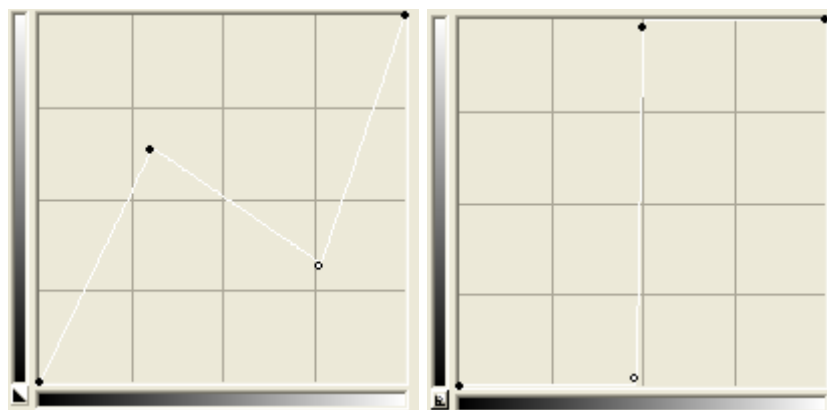
Rys. XIV.4. Przykłady ilustrujące działanie wybranych operacji jednopunktowych: (a) oryginał, (b) negatyw, (c) progowanie, (d) pseudokolorowanie, (e) zamiana poziomów jasności, (f) charakterystyka przejściowa

Przy przetwarzaniu jednopunktowym wykonuje się pewną operację w odniesieniu do każdego piksela niezależnie. Sposób wykonania operacji na ogół opisuje się albo poprzez podanie tablicy przekodowania wartości poszczególnych pikseli albo poprzez określenie funkcji przypisującej nowe wartości pikselom o określonych jasnościach.

Tablica przekodowania jest stosowana na przykład przy operacjach zmieniania barw w obrazie. Każdy element takiej tablicy określa nową barwę jaką należy zastąpić określona barwę w oryginalnym obrazie. Na przykład, pikselowi o odcieniu szarości 12 należy przypisać barwę zieloną, a pikselowi o odcieniu szarości 127 należy

przypisać barwę czerwoną itd. Tego typu przetwarzanie obrazu jest określane jako pseudokolorowanie. Tablica zmiany wartości pikseli może określać wyłącznie wymianę wartości poszczególnych pikseli w ramach pierwotnego zbioru wartości pikseli występujących w obrazie. Tego typu operację wykonano na rysunku XIV.4e.

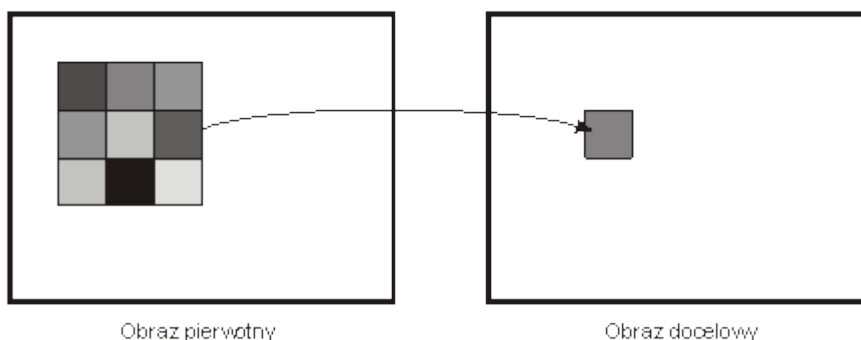
Wymagany rodzaj przekształcenia można określać za pomocą funkcji przypisujących nowe wartości poszczególnym wartościom oryginalnym. Na przykład w celu uzyskania negatywu obrazu z odcieniami szarości można wykorzystać funkcję typu $L_{\diamond} = 255 - L$, gdzie jasności pikseli: pierwotna L i nowa L_{\diamond} są zapisane dwójkowo w skali od 0 do 255. Stosowane mogą być różne funkcje: potęgowe, wykładnicze, logarytmiczne, monotoniczne i niemonotoniczne, bądź dobrane indywidualnie dla potrzeb konkretnego zadania przetwarzania. Na przykład obrazek z rysunek XIV.4f został uzyskany z obrazka z rysunku XIV.4a za pomocą funkcji takiej jak na rysunku XIV.5a. W szczególnym przypadku funkcja może być taka jak na rysunku XIV.5b. Wtedy wszystkie poziomy jasności mniejsze od wartości progowej, przy której następuje zmiana poziomu funkcji zostają zamienione na barwę czarną a wszystkie poziomy o wartości większej od wartości progowej są zamieniane na barwę białą. W takim przypadku uzyskujemy obraz czarno-biały a wykonywane przekształcenie jest określane jako progowanie obrazu (por. rysunek XIV.4c).



Rys. XIV.5. Przykłady funkcji określających sposób zmiany jasności pikseli. Na osi poziomej są wartości jasności przed przekształceniem, na osi pionowej wartości jasności pikseli po przekształceniu. a) Funkcja wykorzystana do uzyskania obrazu z rysunku XIV.4f, b) funkcja progowa

4. Przekształcenia wykorzystujące filtry

Kolejną grupę przekształceń tworzą przekształcenia wykorzystujące sąsiedztwo pikseli. Operacje tego typu wykonuje się stosując odpowiednie filtry czy też maski. Istotę operacji wyjaśnia rysunek XIV.6. Operację wykonuje się w odniesieniu do każdego piksela pierwotnego obrazu. Wynik operacji zapisuje się w nowym docelowym obrazie.



Rys. XIV.6. Ilustracja metody przetwarzania z wykorzystaniem filtra 3×3 . Na podstawie informacji o pikselach tworzących sąsiedztwo danego piksela wyznaczana jest jasność piksela w docelowym obrazie

Maska (filtr) jest to tablica elementów, najczęściej kwadratowa o wymiarach 3×3 , 5×5 itp. Poszczególne elementy tablicy stanowią współczynniki wagowe, przez które mnoży się odpowiednie wartości pikseli z sąsiedztwa analizowanego piksela. Wartość nowego piksela wyznacza się z zależności

$$p_{x,y} = \frac{\sum_{i,j=-m}^m w_{ij} p_{x+i,y+j}}{\sum_{i,j=-m}^m w_{ij}},$$

gdzie współczynniki w_{ij} są elementami filtra kwadratowego o boku $2m + 1$, $p_{x,y}$ jest przetwarzanym pikselem, natomiast $p_{x+i,y+i}$ są pikselami sąsiedztwa w kwadracie o boku $2m + 1$. Oczywiście, jeżeli suma współczynników wagowych filtra jest równa 0 albo 1, to wynikową wartość piksela określa tylko licznik podanego wzoru.

Zauważmy dodatkowo, że nieco odmiennie muszą być potraktowane piksele należące do pierwszego i ostatniego wiersza obrazu. Stosuje się tu różne rozwiązania. W najprostszym przypadku nie przetwarza się pikseli należących do tych wierszy. W bardziej złożonym rozwiązaniu stosuje się specjalne filtry o wymiarze 2×3 .

Rodzaj wykonywanej operacji zależy od używanej maski. Znanych jest wiele różnych filtrów. W szczególności wyróżnia się filtry dolnoprzepustowe, które pozwalają usuwać drobne zakłócenia w obrazie (uśredniać, co w efekcie prowadzi do pewnego rozmycia obrazu), górnoprzepustowe, które pozwalają uwypuklać krawędzie występujące w obrazie (wyostrzać obraz) oraz filtry wykrywające krawędzie. Na rysunku XIV.7 podano kilka przykładowych filtrów. Natomiast na rysunku XIV.8 pokazano kilka przykładów stosowania takich operacji.

a)

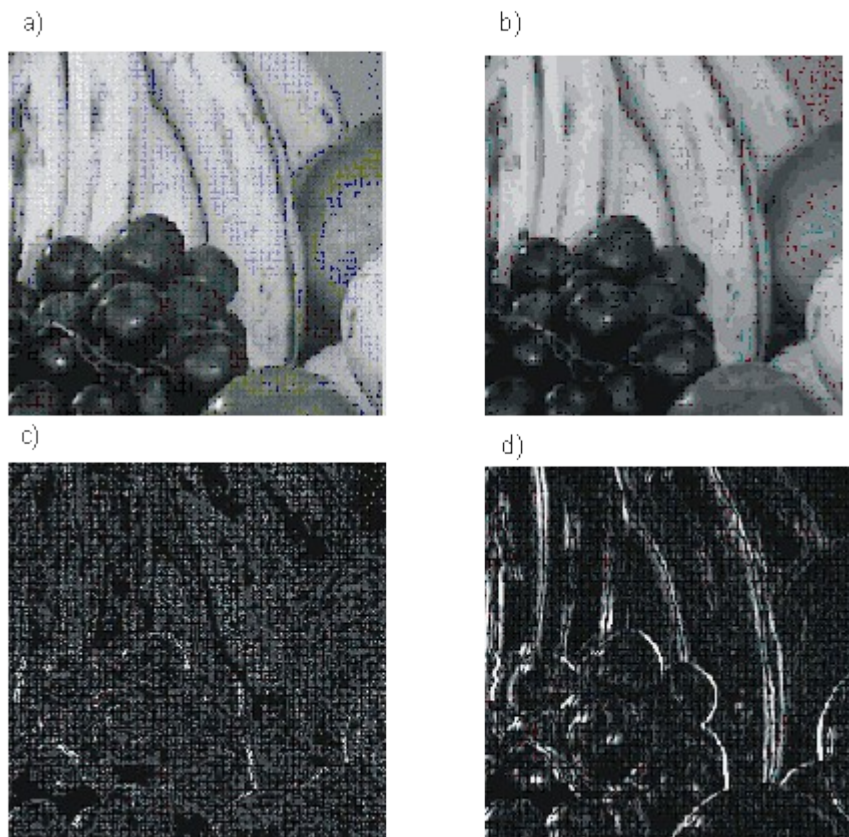
$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

b)

c)

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

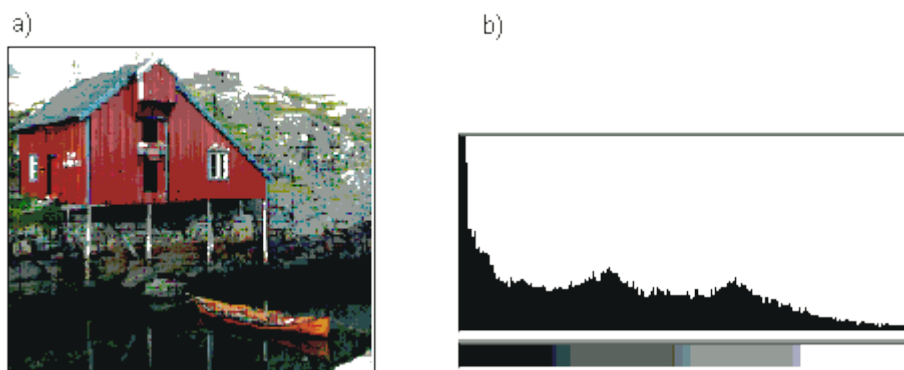
Rys. XIV.7. Przykłady filtrów: a) dolnoprzepustowy (uśredniający), b) górnoprzepustowy (wyostrzający), c) wykrywający krawędzie (Sobela)



Rys. XIV.8. Przykłady stosowania filtrów: a) Oryginał, b) filtr dolnoprzepustowy, c) filtr górnoprzepustowy, d) wykrywanie krawędzi

5. Histogram obrazu

W przetwarzaniu obrazów wykorzystuje się pojęcie histogramu obrazu. Histogram jest to wykres, w którym dla każdej barwy (odcienia szarości) występującej w obrazie podaje się liczbę pikseli o tej barwie znajdujących się w obrazie. Liczby te są reprezentowane za pomocą pionowych odcinków o odpowiednich wysokościach. Na rysunkach XIV.9a i b pokazano przykładowy obraz i jego histogram.



Rys. XIV.9. a) Obraz i b) jego histogram

Podsumowanie

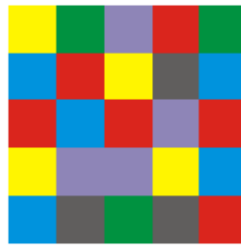
Przedstawiony w wykładzie przegląd metod przetwarzania obrazów stanowi w istocie jedynie wprowadzenie do przetwarzania obrazów. Przegląd ten pozwala jednak zorientować się w zakresie możliwości jakie dają nam te metody. Wiele z metod przetwarzania obrazów jest dostępnych w programach graficznych.

Przykładowe pytania i problemy do rozwiązania

1. Zaproponować algorytm znajdowania odbicia lustrzanego obrazu w pionie.
2. Wyjaśnić na czym polega zadanie pseudokolorowania.
3. Obliczyć wartość piksela jaką uzyskamy po zastosowaniu maski M do zestawu pikseli P , jeżeli

$$M = \begin{bmatrix} 2 & 1 & 2 \\ 2 & 4 & 2 \\ 2 & 2 & 1 \end{bmatrix} \quad P = \begin{bmatrix} 32 & 2 & 1 \\ 2 & 1 & 3 \\ 2 & 2 & 3 \end{bmatrix}$$

4. Wyjaśnić pojęcie histogramu obrazu. Proszę również znaleźć histogram dla obrazu pokazanego na rysunku. W obrazie wykorzystano 6 różnych barw.



Wykład 15: Przechowywanie obrazów

Streszczenie

Utworzone obrazy muszą być w jakiejś formie pamiętane. Służą do tego odpowiednie formaty plików. Zanim jednak przedstawimy kilka popularnych formatów plików omówimy problem kompresji - metody kompresji pozwalają zmniejszyć wielkość plików, a tym samym zmniejszyć zapotrzebowanie na pamięć i skrócić czas transmisji obrazów. Praktycznie wszystkie formaty dopuszczają stosowanie kompresji.

1. Kompresja obrazów

Gotowe obrazy są pamiętane w postaci map bitowych. Przy obecnie stosowanych rozdzielczościach obrazów 800 \times 600, 1024 x 768 czy 1280 \times 1024 i 24 bitach na piksel ilość miejsca w pamięci potrzebna na zapisanie pojedynczego obrazu jest znaczna (odpowiednio w przybliżeniu 1,44 MB, 2,36 MB, 3,93 MB). Dodatkowy problem pojawia się przy przesyłaniu obrazów - czas transmisji jest odpowiednio długi. Stąd pojawia się potrzeba stosowania kompresji obrazów, a więc zapisywania ich w takiej postaci, żeby możliwie maksymalnie zmniejszyć ilość pamięci potrzebnej do zapamiętania obrazu.

W kompresji obrazów wyróżnia się dwa podejścia. W jednym z nich zakłada się, że kompresja ma być bezstratna. Oznacza to, że obraz poddany kompresji, a następnie dekompresji będzie wyglądał dokładnie tak samo jak obraz pierwotny - w czasie procesu kompresja-dekompresja nie jest tracona żadna informacja. Założenie to istotnie ogranicza możliwość uzyskania wysokiej kompresji. W praktyce, metody bezstratne pozwalają uzyskiwać stopnie kompresji (stosunek wymaganej pojemności pamięci przed kompresją do wymaganej pojemności po kompresji) rzędu kilku.

W drugim podejściu do kompresji dopuszcza się pewną utratę informacji w procesie kompresja-dekompresja obrazu. Mówi się wtedy o kompresji stratnej. W tym przypadku można uzyskiwać wysokie stopnie kompresji rzędu kilkudziesięciu. W metodach realizujących kompresję stratną z reguły dąży się do usuwania informacji mających stosunkowo małe znaczenie dla interpretacji całego obrazu, a więc dąży się do usuwania mało istotnych szczegółów.

Dalej omówimy przykładowe metody kompresji bezstratnej i stratnej.

Bodaj najprostszą metodą kompresji bezstratnej jest metoda kodowania ciągów znana pod nazwą metoda RLE. Istota tej metody polega na wyszukiwaniu ciągów identycznych elementów i zapamiętywaniu wartości tych elementów i liczby ich powtórzeń. Załóżmy, że piksele w obrazie monochromatycznym (z odcieniami szarości) są reprezentowane za pomocą bajtów i że pewien ciąg pikseli w obrazie jest jak na rysunku XV.1a. Ten ciąg można zapisać krócej, zgodnie z podaną zasadą, tak jak na Rys. XV.1b. W każdej parze liczb pierwsza oznacza liczbę powtórzeń a druga wartość piksela. Oczywiście mając skompresowany ciąg i pamiętając zasadę kompresji można łatwo dokonać dekompresji i odtworzyć dokładnie pierwotny ciąg pikseli.



Rys. XV.1. Przykład kompresji metodą RLE. a) Informacja przed kompresją, b) informacja po kompresji

Można zauważyć, że skuteczność kompresji jest duża wtedy, gdy występują długie ciągi takich samych wartości. Jeżeli jednak pojawiają się pojedyncze wartości, to zamiast zysku może wystąpić strata. (W niektórych implementacjach stosuje się rozwiązania zwiększające efektywność w takich przypadkach.) Najczęściej, w konkretnych implementacjach, kompresji RLE poddawane są poszczególne wiersze obrazu. Nie jest to jednak regułą. Można spotkać rozwiązania, w których kodowane są kolumny obrazu albo ciągi pikseli wybieranych metodą ZigZag (por. Rys. XV.4).

Zwróćmy uwagę na pewien problem związany z kodowaniem obrazów barwnych. Jeżeli każdy piksel jest reprezentowany za pomocą trzech składowych RGB, to obraz może być zapisany jako ciąg trójek RGB reprezentujących kolejne piksele albo za pomocą trzech ciągów zawierających odpowiednio tylko składowe R albo tylko składowe B albo tylko składowe C kolejnych pikseli (por. rysunek XV.2). Można

zauważyć, że w tym drugim przypadku efektywność kompresji może być większa, co wynika stąd, że szansa na pojawianie się dłuższych ciągów identycznych wartości jest większa w obrębie ciągu tych samych składowych (tworzących tak zwane płaszczyzny odpowiednio R, G i B).

a)	R	G	B	R	G	B	R	G	B	R	G	B
	255	26	70	255	26	71	255	26	72	255	25	73

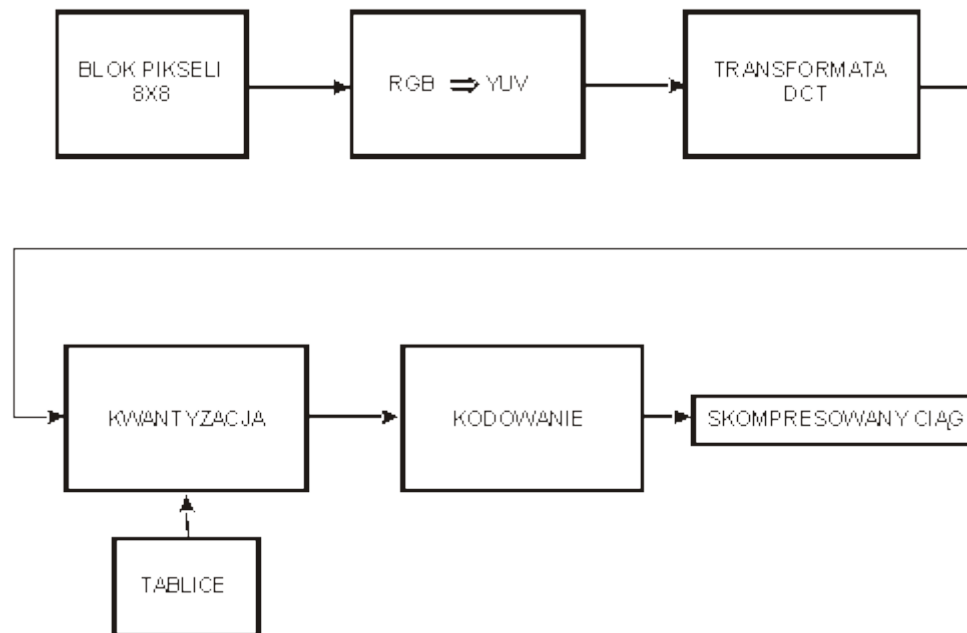
b)	R	R	R	R	G	G	G	G	B	B	B	B
	255	255	255	255	26	26	26	26	70	71	72	73

Rys. XV.2. Wpływ uporządkowania danych na efektywność kompresji. a) Uporządkowanie pikselowe, b) uporządkowanie według składowych barwy

Ogólnie metoda kompresji RLE pozwala uzyskiwać stopnie kompresji na poziomie 2 : 1, 3 : 1. Oczywiście stopień kompresji zależy od konkretnego obrazu - dla jednych obrazów można uzyskać większy stopień kompresji, dla innych mniejszy. Dodajmy, że jest to cecha charakterystyczna każdej metody kompresji.

Spośród innych metod kompresji bezstratnych często stosowanych w praktyce należy wymienić metodę Huffmana oraz metodę LZW. Istotą metody Huffmana jest to, że stosuje się kody o zmiennej długości. Pikselom, które występują częściej w danym obrazie przyporządkowuje się krótsze kody. Z kolei metoda LZW jest metodą słownikową. Istota metody polega na tym, żeby możliwie najdłuższym ciągom powtarzających się wartości pikseli przypisywać odpowiednie kody. Metoda LZW jest objęta patentem (natomiast wcześniejsza wersja metody LZ77 nie jest objęta patentem).

Jedną z najczęściej stosowanych metod kompresji stratnych jest metoda JPEG. Ze względu na złożoność metody ograniczymy się do przedstawienia ogólnej koncepcji tej metody. Na początku obraz dzielony jest na podobrazy (bloki) o rozmiarach 8x8 pikseli. Procesowi kompresji jest poddawany każdy blok niezależnie. Proces ten obejmuje cztery kroki pokazane na rysunku XV.3.



Rys. XV.3. Schemat blokowy metody kompresji JPEG

W pierwszym kroku wykonuje się konwersję z modelu RGB do modelu Y_C, C_b . Występuje tu pierwsza kompresja, bowiem o ile w modelu RGB przeznacza się 24 bity na piksel, to w modelu Y_C, C_b tylko 16 (8 bitów na składową luminancji Y i po 4 bity na składowe niosące informację o barwie - wykorzystuje się tu fakt, iż oko człowieka jest bardziej czułe na zmiany jasności niż na zmiany barwy). Dalsze kroki są realizowane niezależnie dla składowych Y, C_r, C_b .

Tablica wartości Y o rozmiarze 8×8 poddawana jest transformacie kosinusowej i w efekcie uzyskuje się nowy zestaw 64 liczb - współczynników z rozwinięcia kosinusowego. Wykorzystuje się tu następującą zależność

$$s_{uv} = \frac{1}{4} C_u C_v \sum_{x=0}^7 s_{yx} \cos\left(\frac{(2x+1)u\pi}{16}\right) \cos\left(\frac{(2y+1)v\pi}{16}\right),$$

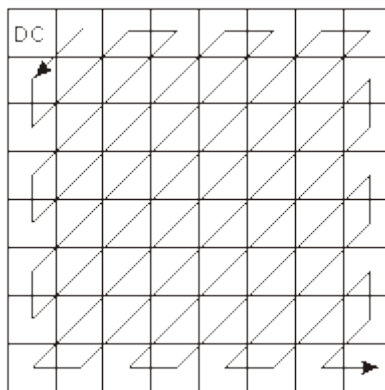
gdzie s_{yx} są to oryginalne dane z tablicy o współrzędnych yx a s_{uv} są to współczynniki uzyskane w wyniku transformacji kosinusowej, zapisywane w tablicy o współrzędnych uv . Współczynniki C_u i C_v są równe $\frac{1}{2}$ gdy $u, v = 0$ i równe 1 w przeciwnym przypadku.

W kolejnym kroku wykonywana jest tak zwana kwantyzacja. Jej celem jest zamienienie możliwie dużej liczby współczynników transformaty na zera. Generalnie chodzi o to, że wśród współczynników transformaty są takie, które niosą dużą ilość informacji o obrazie i takie, które niosą niewielką ilość informacji o obrazie. Te ostatnie można wyeliminować bez większej straty dla ogólnej jakości obrazu - ich wpływ jest na tyle mały, że obserwator może nie zauważyć zmian w obrazie związanych z ich usunięciem.

Kwantyzację realizuje się w następujący sposób. Korzystamy z pomocniczej tablicy 8×8 , zawierającej odpowiednio dobrane wartości. Kolejno dzieli się przez siebie

współczynniki z tablicy uzyskanej w wyniku transformacji przez odpowiednie współczynniki z pomocniczej tablicy (to znaczy pierwszy element tablicy współczynników transformacji dzielimy przez pierwszy element pomocniczej tablicy, potem to samo wykonujemy w stosunku do drugich współczynników itd.) i wyniki zaokrągla się w dół. Elementy pomocniczej tablicy są tak dobrane, żeby w wynikowej tablicy pojawiała się duża liczba zer. Są one z reguły zgrupowane w prawej dolnej części wynikowej tablicy. Dzięki operacji kwantyzacji uzyskuje się kompresję informacji, przy czym jest to operacja nieodwracalna.

Tablica uzyskana w wyniku kwantyzacji jest w ostatnim kroku poddawana kodowaniu metodą RLE i następnie metodą Huffmana. Kodowaniu metodą RLE poddawany jest ciąg wartości z wynikowej tablicy przeglądany w porządku ZigZag jak na rysunku XV.4. Taki porządek jest przyjęty ze względu na to, że pozwala uzyskać długi ciąg zer zgrupowanych w prawej dolnej części tablicy. Zauważmy, że do ciągu nie wchodzi element z lewego górnego rogu oznaczony jako DC. Jest on kodowany wspólnie z odpowiednimi elementami dla innych bloków obrazu 8x8, z których składa się cały obraz poddawany kompresji.



Rys. XV.4. Przeglądanie elementów tablicy w porządku ZigZag

Przy dekompresji obrazu postępuje się w odwrotnej kolejności jak przy kompresji. Najpierw ma miejsce odkodowanie ciągu właściwe dla metody Huffmana, a potem dla metody RLE. Odtworzona tablica jest poddawana odwrotnej operacji jak przy kwantyzacji (z wykorzystaniem tej samej pomocniczej tablicy), z tym, że oczywiście nie można odtworzyć wyzerowanych współczynników. Z kolei odzyskana tablica współczynników transformaty kosinusowej jest poddawana odwrotnej transformacji kosinusowej. Na końcu następuje konwersja z modelu $YCbCr$ do modelu RGB.

Kompresja metodą JPEG pozwala uzyskiwać stopnie kompresji 20:1 - 30:1 przy praktycznie niezauważalnych zniekształceniach obrazu odzyskiwanego po dekompresji. Przy większych stopniach kompresji zmiany takie są już zauważalne.

Na zakończenie dodajmy, że omówione wyżej metody kompresji są stosowane do kompresji pojedynczych obrazów. Przy kompresji sekwencji obrazów są stosowane inne metody, na przykład takie jak w standardach MPEG.

2. Formaty plików

Po wygenerowaniu obrazu powstaje problem w jaki sposób go przechowywać i przesyłać. Przyjęty sposób reprezentacji obrazu powinien umożliwiać odtworzenie obrazu po odczytaniu z pamięci lub po zakończeniu transmisji. Równocześnie, reprezentacja powinna być możliwie oszczędna, głównie ze względu na ograniczone szybkości transmisji oraz ograniczenia związane z zajętością pamięci. Metoda reprezentacji powinna być dostatecznie elastyczna po to, żeby możliwe było zapisywanie informacji o różnych obrazach, poczynając od niedużych obrazów czarno-białych a kończąc na obrazach o dużej rozdzielczości i dużej liczbie bitów poświęconych na pamiętanie informacji o barwie pojedynczego piksela.

Jak dotychczas nie opracowano jednolitej reprezentacji obrazów. Wynika to z jednej strony stąd, że trudno jest spełnić równocześnie wymienione wyżej wymagania, z natury sprzeczne ze sobą. Z drugiej strony nie bez znaczenia jest stosowanie przez różne firmy polityki zmierzającej do posiadania "swoich" standardów. Skutkiem tego jest to, że opracowano wiele różnych metod reprezentacji obrazów, tak zwanych formatów.

W każdym z formatów można na ogół wyróżnić następujące części: nagłówek, opis obrazu, określoną reprezentację obrazu oraz informacje dodatkowe. W nagłówku najczęściej są podawane ogólne dane o formacie, w szczególności pozwalające zidentyfikować format, o pochodzeniu obrazu, autorze itd. W części poświęconej opisowi obrazu są dane charakteryzujące obraz jako całość, a więc informacje o wielkości obrazu (rozdzielczości), liczbie bitów/piksel, sposobie reprezentacji informacji o barwie, rodzaju użytej kompresji itd. Część zawierająca reprezentację obrazu obejmuje informacje o obrazie zapisane zgodnie z przyjętą konwencją w danym formacie. Informacje dodatkowe mogą dotyczyć różnych aspektów związanych z możliwościami wyświetlenia czy reprodukcji obrazu - w szczególności, mogą tu być zamieszczane informacje o profilu urządzenia, na którym obraz został uzyskany.

Niektóre z opracowanych formatów są stosowane w ograniczonym zakresie, inne zdobyły większą popularność. Niżej ograniczymy się do scharakteryzowania kilku najpopularniejszych formatów, podając każdorazowo kilka najbardziej charakterystycznych informacji. Więcej uwagi poświęcimy formatowi PNG, który zdobywa coraz większą popularność w aplikacjach internetowych. Omawiane formaty to z reguły formaty, w których obraz reprezentowany jest w postaci mapy bitowej (formaty rastrowe, bitmapowe). Zasygnalizujemy jednak, że są również formaty wektorowe, które umożliwiają przechowywanie informacji o obrazie w postaci ciągu poleceń graficznych i danych, które pozwalają utworzyć obraz w postaci pikselowej. Ponadto zasygnalizujemy istnienie formatów pozwalających na reprezentowanie sekwencji obrazów tworzących animację (na przykład avi).

Wobec istnienia wielu różnych formatów powstaje oczywiście problem konwersji między tymi formatami. Problem ten w praktyce rozwiązuje się na dwa sposoby. Pierwsze rozwiązanie polega na tym, że producenci oprogramowania graficznego wyposażają swoje produkty w możliwości importowania obrazów zapisanych w kilku wybranych formatach oraz w możliwości eksportowania obrazów w kilku wybranych formatach. Drugie rozwiązanie polega na tym, że udostępniane są niezależne programy konwersji między wybranymi formatami. Warto tu zwrócić uwagę na to, że o ile problem konwersji między formatami przeznaczonymi do zapisywania

obrazów bitmapowych jest stosunkowo prosty, to w przypadku formatów przechowujących informację w postaci wektorowej realizacja konwersji jest bardziej złożona.

W praktyce stosuje się wiele różnych formatów zapisu obrazów. Do najczęściej stosowanych należą GIF, PNG, JPG. Każdy z tych formatów ma specyficzne cechy, które powodują, że jest on stosowany w odpowiednich aplikacjach.

Format TIFF. Format TIFF (Tag Image File Format) jest bardzo elastyczny. Umożliwia on pamiętanie informacji o obrazach bitmapowych o różnych rozdzielczościach, zarówno czarno-białych jak i kolorowych (do 24 bitów/piksel). Dopuszczalne jest korzystanie z różnych modeli barw (RGB, CMYK, YC_bC_r itd.). Można przechowywać informację o przezroczystości piksela. Dopuszczalne są różne metody kompresji (w tym brak kompresji, kompresja RLE, LZW, JPEG i inne). Można również dołączać dodatkowe informacje o obrazie umożliwiające jego reprodukcję na różnych urządzeniach.

Format GIF. W formacie GIF (Graphics Interchange Format) wykorzystuje się kompresję bezstratną LZW (chronioną patentem). Format ten dopuszcza przechowywanie informacji o obrazach barwnych, w których wykorzystuje się co najwyżej 256 różnych barw (8 bitów/piksel). Format ten umożliwia określanie przezroczystości tła oraz pamiętanie w pliku wielu obrazów, co stwarza możliwość pamiętania prostych animacji. Format GIF zyskał dużą popularność jednak jego rozwój ograniczają aspekty prawne związane z koniecznością wnoszenia opłat licencyjnych.

Format PNG. Odpowiedzią środowiska internetowego na ograniczenia związane ze stosowaniem formatu GIF było opracowanie formatu PNG (Portable Network Graphics). Format ten zapewnia dobrą kompresję bezstratną, nie jest objęty żadnymi zastrzeżeniami patentowymi i stąd coraz częściej wypiera format GIF.

Sygnatura bloku składa się z 8 bajtów o wartościach 137, 80, 78, 71, 13, 10, 26, 10 odpowiadających znakom ASCII: 137, P, N, G, RETURN, LINEFEED, CTRL/Z, RETURN. Ciąg P, N, G identyfikuje format. Pozostałe znaki są istotne dla dekodera. Dzięki nim np. można odróżnić plik tekstowy od pliku PNG i ewentualnie zatrzymać drukowanie pliku.

W formacie PNG wykorzystywana jest kompresja bezstratna bazująca na metodzie kompresji LZ77, która nie jest chroniona patentami. Po dokonaniu kompresji metodą LZ77 otrzymany ciąg poddawany jest kompresji metodą Huffmana.

Format umożliwia kompresję obrazów, w których dla zapamiętania informacji o pikselu wykorzystuje się do 48 bitów. Dopuszczalny jest kanał umożliwiający sterowanie przezroczystością. Precyzja próbek (liczba bitów dla reprezentowania składowej barwy) może być 1-, 2-, 4-, 8- lub 16- bitowa.

Format PNG dopuszcza pięć metod reprezentowania koloru piksela w obrazie. Pierwszy sposób polega na reprezentowaniu barwy w postaci trójki RGB. Każdy piksel jest reprezentowany przez trzy składowe 8- albo 16- bitowe (przy pamiętaniu obowiązuje kolejność R, G, B).

Drugi sposób pozwala na używanie palety kolorów. W takim przypadku informacja o konkretnym pikselu faktycznie jest indeksem do palety kolorów (która musi być dołączona do pliku).

W przypadku obrazów monochromatycznych można korzystać ze skali szarości.

Efekt przezroczystości obrazu można uwzględnić korzystając z kanału α , który umożliwia łączenie obrazu z tłem. W tym przypadku, obok wartości RGB pamiętana jest wartość pomocniczego parametru tak zwanego parametru α . Liczba bitów poświęcana na zapamiętanie wartości składowych R, G, B i α jest taka sama (8 albo 16). Jeżeli wartość parametru α jest równa zero, to piksel jest w pełni przezroczysty, a więc tło jest w pełni widoczne. Przy maksymalnej wartości parametru α tło jest całkowicie zastąpione przez obraz. W pozostałych przypadkach barwa piksela B jest mieszana z barwą tła T (na zasadzie $(\alpha) B + (1 - \alpha) T$).

Ostatni sposób reprezentowania koloru polega na uwzględnieniu współczynnika α w przypadku obrazów monochromatycznych. Podaje się wtedy wartość piksela oraz wartość współczynnika α .

W formacie PNG przewidziano możliwość rozwiązywania problemu przenoszenia obrazów między różnymi urządzeniami. W tym celu pamiętane są informacje pozwalające odtworzyć kolory, które faktycznie były użyte przy tworzeniu źródłowego obrazu (profil urządzenia). W szczególności, pamiętane są parametry luminoforów monitora i informacja o bieli monitora. Pamiętana jest również wartość γ systemu użytego do utworzenia obrazu.



Format PNG pozwala opisywać jedynie obrazy statyczne. Ograniczenie to ma szansę być usunięte w tworzonym formacie MNG. Warto zwrócić uwagę na fakt, że w formacie PNG wbudowane są mechanizmy kontroli poprawności danych w procesie kompresji i dekompresji. Jest to szczególnie istotne z punktu widzenia odporności na błędy transmisji.

Format JPG. Wyżej omówione formaty dopuszczają kompresję bezstratną. Spośród formatów wykorzystujących kompresję stratną największą popularność zyskał format JPG, który pozwala zapisywać obrazy skompresowane metodą JPEG. Format ten wykorzystywany jest najczęściej do przechowywania obrazów pełnokolorowych (24 bity/piksel), w tym do przechowywania i przesyłania zdjęć.

Podsumowanie

W wykładzie poruszyliśmy problem przechowywania obrazów - utworzony obraz, często dużym nakładem pracy, musi zostać zapamiętany w postaci jakiegoś pliku. Forma zapisania obrazu nie jest obojętna. Obraz musi dać się odtworzyć zarówno bezpośrednio twórcy obrazu jak i innym użytkownikom, do których plik z obrazem zostanie przesłany. Problem ten pozwalają rozwiązać różne formaty plików, które na ogół dopuszczają stosowanie kompresji obrazu. W trakcie wykładu poznaliśmy kilka podstawowych metod kompresji, zarówno bezstratnych jak i stratnych.

Przykładowe pytania i problemy do rozwiązania

1. Wyjaśnić na czym polega problem kompresji i porównać metody kompresji bezstratnej i stratnej.
2. Wyjaśnić dlaczego w końcowej fazie kompresji metodą JPEG wykorzystuje się przegładanie tablicy elementów metodą ZigZag.
3. Zaproponować prosty format pliku do pamiętania obrazów z odcieniami szarości. Założyć, że pamiętane będą obrazy o rozdzielczości 800  600  8.
4. Wymienić podstawowe cechy formatu PNG.

Informacje końcowe

W poszczególnych wykładach poznaliśmy podstawowe pojęcia i metody grafiki komputerowej. Wszystkie poznane algorytmy praktycznie są wykorzystywane w różnego rodzaju programach graficznych. Znajomość tych algorytmów pozwala lepiej zrozumieć działanie poszczególnych narzędzi spotykanych w programach graficznych, ich możliwości i ograniczenia. Ponadto stwarza to możliwość samodzielnego pisania oprogramowania graficznego. Mimo ogromnej oferty rynkowej wciąż jest jeszcze wiele do zrobienia, a równocześnie istnieje stałe zapotrzebowanie na nowe oprogramowanie o bogatszych możliwościach funkcjonalnych, bardziej przyjazne dla użytkownika i przystosowane do konkretnych potrzeb użytkownika.

Na rynku dostępnych jest kilkaset różnych programów graficznych, poczynając od darmowych a kończąc na drogich systemach komercyjnych. Możliwości funkcjonalne programów są bardzo zróżnicowane. Wybór konkretnego programu zależy od zastosowania. Przy prostych zastosowaniach z reguły jeden z wielu dostępnych programów w zupełności wystarczy. Przy bardziej zaawansowanych zastosowaniach czasami trzeba korzystać z kilku programów, wykorzystując unikatowe możliwości poszczególnych programów.

Klasyfikacja programów graficznych nie jest prosta z tego względu, że wiele z nich posiada tak różnorodne funkcje, że wiele programów można przypisywać do różnych kategorii. W tym kontekście sensowny wydaje się być podział ze względu na główny obszar zastosowań. W szczególności wyróżnić można (w nawiasach podano przykładowe programy): niskopoziomowe biblioteki graficzne, programy do tworzenia rysunków wektorowych, programy do przetwarzania obrazów, programy do wizualizacji danych, narzędzia do modelowania, narzędzia do renderingu, narzędzia do animacji, narzędzia dla tworzenia grafiki umieszczanej w sieci, konwertery formatów plików graficznych itd.

Ze zrozumiałych względów nie ma sensu omawianie tutaj poszczególnych programów. Omówienie każdego z programów wymaga niezależnego obszernego opracowania, a ponadto warto opisywać program jedynie wtedy gdy ma się możliwość korzystania z niego. Jedyny wyjątek uczynimy w odniesieniu do programów bibliotecznych niskiego poziomu, które udostępniają zestawy poleceń graficznych lub funkcji. Z bibliotek tych można korzystać z poziomu różnych języków programowania (C, Java itd.). Zapewniają one interfejs programowy do sprzętu graficznego.

Najpopularniejszą biblioteką jest OpenGL (Open Graphics Library). Zawiera ona ponad 100 procedur i funkcji, które umożliwiają programiście specyfikowanie obiektów i operacji potrzebnych dla tworzenia obrazów. Zakres dostępnych funkcji rozciąga się od rysowania pojedynczego punktu, odcinka czy wypełnionego wielokąta do teksturowania krzywoliniowych powierzchni NURBS. Dostępne funkcje pozwalają realizować oświetlenie i cieniowanie, animację i różne efekty specjalne. Wiele narzędzi graficznych i aplikacji naukowych zostało opracowanych z wykorzystaniem tej biblioteki. Wiele funkcji biblioteki OpenGL jest implementowanych sprzętowo. Można spotkać się z określeniem, że OpenGL jest "programowym interfejsem sprzętu graficznego".

Drugą pod względem popularności jest biblioteka Direct3D. Jest ona szczególnie popularna w programowaniu gier dla komputerów PC. Jest to biblioteka graficzna niskiego poziomu na platformę Windows. Zbiór funkcji jest porównywalny ze zbiorem funkcji udostępnianych przez bibliotekę OpenGL. Zarówno Direct3D jak i OpenGL są wspierane przez producentów kart graficznych. Direct3D wchodzi w skład DirectX - zbioru API dostępnych jako obiekty COM (DirectDraw, DirectSound, DirectPlay, Direct3D, DirectInput).

Dostępne są również biblioteki graficzne wysokiego poziomu (np. OpenInventor). Umożliwiają one na przykład konstrukcje scen, import i eksport plików 3D, operowanie na obiektach i wyświetlanie. Biblioteki wysokiego poziomu są często określane jako narzędzia do animacji, do symulacji czy sztucznej rzeczywistości.

Podsumowując część wykładową kursu mam nadzieję, że przedstawiony materiał był zrozumiały i jego poznanie pozwoliło zrozumieć o co chodzi w grafice komputerowej ale równocześnie nasunęło refleksję, że wiele rzeczy jest wciąż niejasnych. Taka refleksja jest oczywiście w pełni uzasadniona. Grafika komputerowa jest już obecnie bardzo rozległą dziedziną wiedzy i nie sposób jest przedstawić wszystkie problemy w ramach jednego kursu. Równocześnie jest to dziedzina wciąż rozwijająca się - stale pojawiają się nowe metody, umożliwiające tworzenie obrazów o coraz lepszej jakości i coraz szybciej. Wciąż też pojawiają się nowe obszary zastosowań, które wymagają opracowywania następnych rozwiązań.

Sądzę również, że dołączone do poszczególnych wykładów pytania i problemy do samodzielnego rozwiązania pozwoliły każdorazowo zweryfikować czy dany wykład został dobrze zrozumiany. Wszystkie pytania i problemy zostały tak dobrane, żeby można było znaleźć odpowiedzi na nie albo bezpośrednio w wykładzie albo żeby można było na podstawie znajomości wykładów dojść samodzielnie do poprawnego rozwiązania. Gdyby jednak pojawiły się trudności ze znalezieniem poprawnych odpowiedzi, bądź gdyby pewne fragmenty wykładów były mimo wszystko niezrozumiałe, to proszę o kontakt za pomocą poczty elektronicznej (jza@ii.pw.edu.pl). Będę również zobowiązany za zwrócenie uwagi na ewentualne zauważone usterki oraz za wszelkie sugestie, które pozwolą ulepszyć kurs.

Z pewnością przedstawiony w wykładach materiał teoretyczny przyda się również w czasie zajęć laboratoryjnych w ramach których poznamy praktycznie wiele narzędzi stosowanych w różnych programach graficznych, w tym z pewnością sporo nowych narzędzi, o których w trakcie wykładu nie było mowy.

Ponieważ, jak to już było sygnalizowane na początku kursu, czeka nas egzamin z materiału wykładowego, niżej podany jest fragment jednego z wcześniej przeprowadzanych egzaminów.

Przykładowe tematy egzaminacyjne

W każdym z tematów od 1 do 10 podane są trzy stwierdzenia. Niektóre z tych stwierdzeń są poprawne, inne nie. Proszę w odpowiednich polach wpisać TAK gdy stwierdzenie jest poprawne albo NIE gdy stwierdzenie jest niepoprawne. Za każdy z tematów 1-10 uzyskuje się 3 punkty, jeżeli związane z tematem trzy odpowiedzi zostały zaznaczone poprawnie. Za każdy z tematów 11 i 12 można uzyskać po 3 punkty. Za temat 13 można uzyskać do 4 punktów. Minimum dla uzyskania oceny pozytywnej wynosi 21 punktów.

1. Podstawowe algorytmy techniki rastrowej	
a. Dwa przecinające się odcinki mogą mieć więcej niż jeden piksel wspólny	
b. Odcinek o współrzędnych końców (2,1), (5,6) narysowany przy wykorzystaniu algorytmu Bresenhama składa się z 6 pikseli	
c. Aliasing jest skutkiem skończonej rozdzielczości rastra	
2. Barwa w grafice komputerowej	
a. W modelu RGB, przy reprezentacji barwy za pomocą 24 bitów, barwa czarna ma współrzędne (255, 255, 255)	
b. W systemie full color (pełnokolorowym) każda składowa barwy jest reprezentowana za pomocą 10 bitów	
c. W modelu CIE XYZ barwy nasycone znajdują się na obwiedni wykresu chromatyczności we współrzędnych xy	
3. Podstawowe przekształcenia geometryczne	
a. Kwadrat o współrzędnych wierzchołków (0, 0), (1, 0), (1, 1), (0, 1) został poddany operacji skalowania ze współczynnikami $S_x = 2$, $S_y = 3$ i przesunięty o wektor (3, 2). Współrzędne jednego z wierzchołków otrzymanej figury to (2, 5)	

b. Punkt o współrzędnych jednorodnych (5, 4, 3,1) w układzie współrzędnych xyz ma współrzędne (5, 4, 3)	
c. W przekształceniu typu obrót, punkt wokół którego następuje obrót musi pokrywać się z początkiem układu współrzędnych	

11. Wymienić trzy właściwości krzywych Béziera
12. Wyjaśnić różnicę między cieniowaniem płaskim a cieniowaniem Gouraud
13. Naszkicować sześcian oświetlony odległym źródłem światła (promienie padają równolegle).