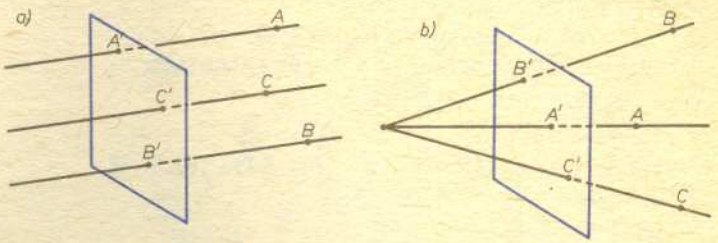
Grafika komputerowa

Dr Michał JANKOWSKI

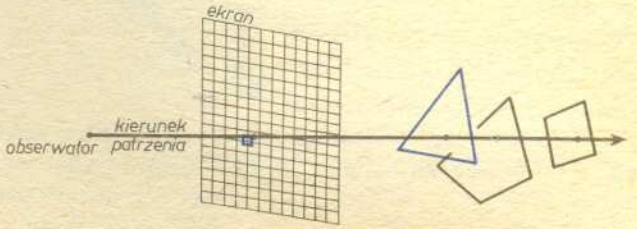
Dla większości młodych ludzi kontakt z informatyką zaczyna się od gier komputerowych. Z czasem nawet najciekawsze z nich opatrują się. Pojawia się chęć rysowania samemu. Najlepiej, żeby tworzone na ekranie monitora obrazki były możliwie realistyczne — kolorowe, uwzględniające oświetlenie i cieniowanie. Jeszcze fajniej, gdy coś się rusza (i to nie raz na godzinę, ale tak szybko, jak w normalnym filmie animowanym). Czy posiadacze Spectrum, Atari, a nawet IBM PC mają szansę osiągnąć takie wyniki?

Zacznijmy od skromnego początku. Załóżmy, że chcemy rysować (kreskami albo kolorowymi plamami) sceny przestrzenne — trójwymiarowe, zbudowane ze ścian będących wielokątami. Mogą to być np. układanki z klocków będących prostopadłościanami, graniastosłupami czy ostrosłupami. Zakładamy przy tym, że ściany nie przecinają się (mogą tylko mieć wspólne krawędzie).



Rys. 1

Rysowaną scenę opisujemy podając współrzędne (x, y, z) wierzchołków wielokątów (= ścian), każda krawędź określona jest przez numery wierzchołków, które łączy; wreszcie ściana definiowana jest numerami uporządkowanych krawędzi. Mając taki opis musimy dla narysowania wykonać rzutowanie, tzn. odwzorować trójwymiarową scenę na płaszczyznę ekranu monitora. Zasady rzutów: równoległego i środkowego ilustrują rysunki 1a i 1b, potrzebne zaś wzory i dokładne wyjaśnienia można znaleźć np. w niedawno wydanej przez WNT książce Iana Angella *Wprowadzenie do grafiki komputerowej*.

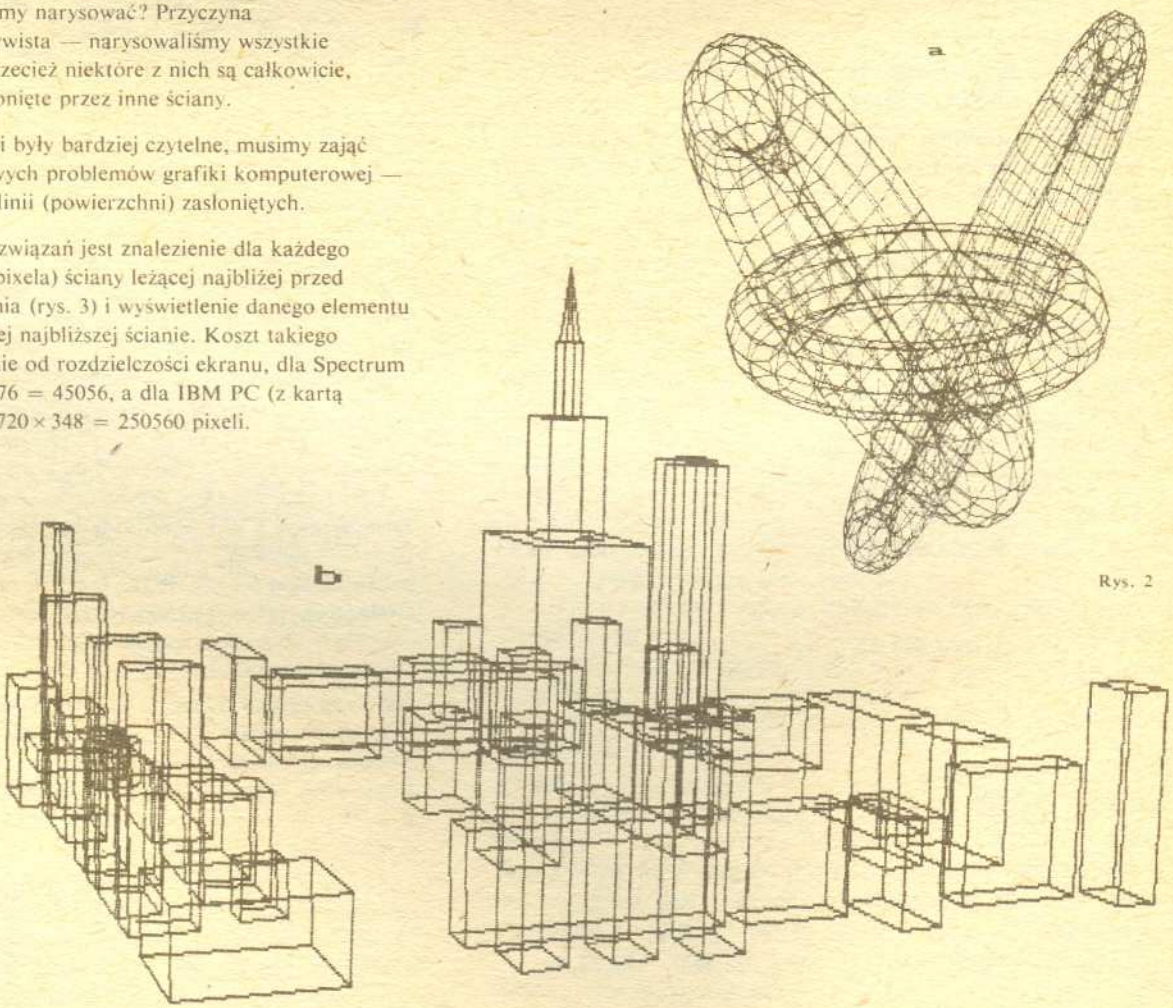


Rys. 3

Po zrzutowaniu wszystkich wierzchołków postępowanie jest proste: łączymy kreskami rzuty końców krawędzi albo wypełniamy kolorem (czy rastrem) wielokąty będące rzutami ścian. W ten sposób otrzymujemy takie rysunki jak 2a i 2b. Prawda, że trudno domyślić się, co chcieliśmy narysować? Przyczyna niepowodzenia jest oczywista — narysowaliśmy wszystkie krawędzie (ściany), a przecież niektóre z nich są całkowicie, a pewne częściowo zasłonięte przez inne ściany.

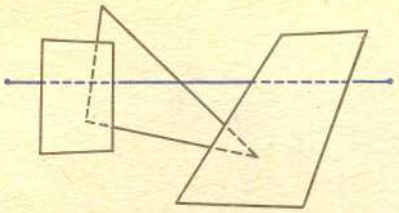
Jeśli chcemy, by rysunki były bardziej czytelne, musimy zająć się jednym z podstawowych problemów grafiki komputerowej — zadaniem wyznaczania linii (powierzchni) zasłoniętych.

Jednym z prostszych rozwiązań jest znalezienie dla każdego elementu ekranu (tzw. pixela) ściany leżącej najbliżej przed nim w kierunku patrzenia (rys. 3) i wyświetlenie danego elementu kolorem przypisanym tej najbliższej ścianie. Koszt takiego algorytmu zależy głównie od rozdzielczości ekranu, dla Spectrum musimy zbadać $256 \times 176 = 45056$, a dla IBM PC (z kartą graficzną Hercules) aż $720 \times 348 = 250560$ pixeli.



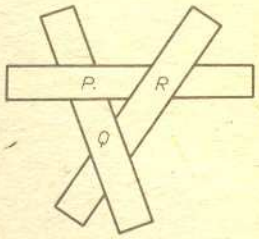
Rys. 2

Gdy chcemy uzyskać rysunek „druciany" (taki jak np. 4), to możemy zastosować algorytm Ricciego (dokładny jego opis i program w Fortranie można znaleźć w rozdziale 8 książki Angella). Zasada tego algorytmu sprowadza się do porównywania rzutu każdej krawędzi z rzutami wszystkich ścian. Sprawdza się, jakie fragmenty krawędzi (rys. 5) są zasłonięte (tzn. leżą za jedną ze ścian). Rozwiązanie jest „brutalne" — badamy położenie wszystkich krawędzi względem wszystkich ścian. Koszt takiego algorytmu na ogół jest proporcjonalny do n^2, gdzie n oznacza liczbę krawędzi lub ścian (przykładowo scena z rysunku 4 zbudowana jest z 1000 czworokątów).



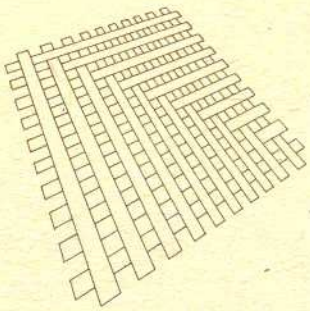
Rys. 5

Gdy rysujemy nie kreskami, ale kolorowymi plamami, to najprościej byłoby ponumerować ściany, czyli ustawić je np. w takiej kolejności: jako pierwszą weźmy ścianę, która żadnej innej nie zasłania, a każda następna niech zasłania tylko poprzednie (już ustawione). Jeśli takie ustawienie jest wykonalne, to możemy wyświetlać na ekranie zadanymi kolorami wielokąty będące rzutami kolejnych ścian. Nowe ściany będą tym samym zamalowywały (zasłaniały) fragmenty wcześniej narysowanych. Ale znowu, tak jak w algorytmie Ricciego, koszt będzie proporcjonalny do n^2 (dla rysunku 6 jest $n = 640$) działań.



Rys. 7

Czy można rozwiązać zadanie wyznaczenia linii (powierzchni) zasłoniętych taniej, a tym samym szybciej? Okazuje się, że dla dowolnych scen trójwymiarowych, zbudowanych z wielokątów, odpowiedź jest negatywna. Może wydać się to dziwne — ustawienie ścian to nic innego jak ich sortowanie, a przecież np. n liczb umiemy uporządkować znacznie taniej niż kosztem rzędu n^2 działań.

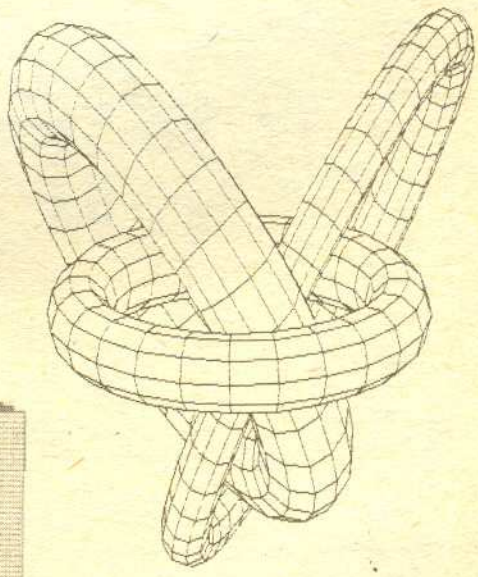


Rys. 8

To prawda — różnica polega na tym, że relacja zasłaniania ścian nie jest porządkiem. Jeśli ściana P zasłania Q, a Q zasłania R, to wcale nie oznacza, że P zasłania R (relacja nie jest przechodnia). Natomiast już przy trzech ścianach może wystąpić tzw. zakleszczenie wzajemnych zasłonięć: P zasłania Q, Q zasłania R, a R zasłania P (rys. 7) — w takim przypadku nie możemy wybrać ściany, która żadnej innej nie zasłania! Przyjrzyjmy się dokładniej rysunkowi 8, ta scena zbudowana jest z n linii, $n/4$ ścian, a liczba widocznych fragmentów ścian jest rzędu n^2.

Wniosek jest chyba pesymistyczny. Rozstrzyganie, co jest widoczne, a co zasłonięte w ogólnym przypadku wymaga sporej liczby działań. Badanie oświetlenia jest jeszcze kosztowniejsze. Mimo tego grafika komputerowa nawet na Spectrum może sprawić frajdę. Ale o tym kiedy indziej.

Rys. 4



Rys. 6