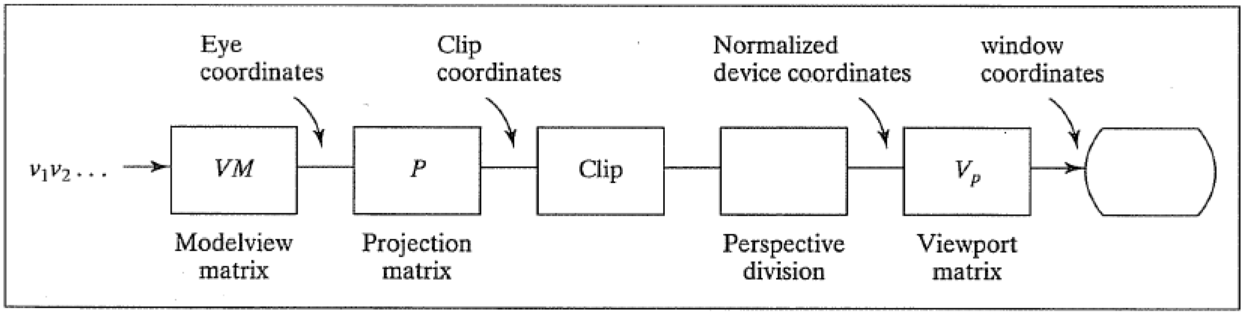
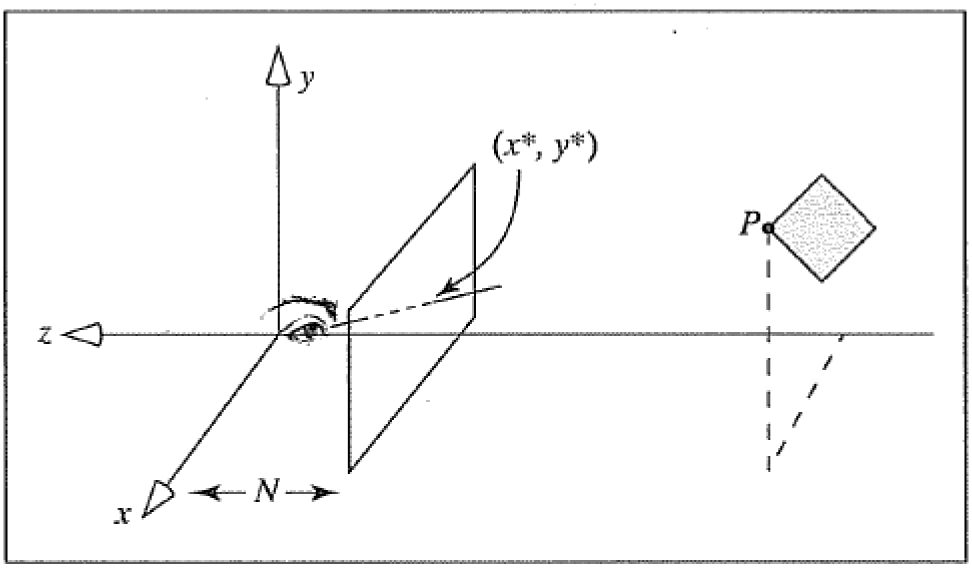
بسمه تعالی

افکنش پرسپکتیو اشیاء سه بعدی

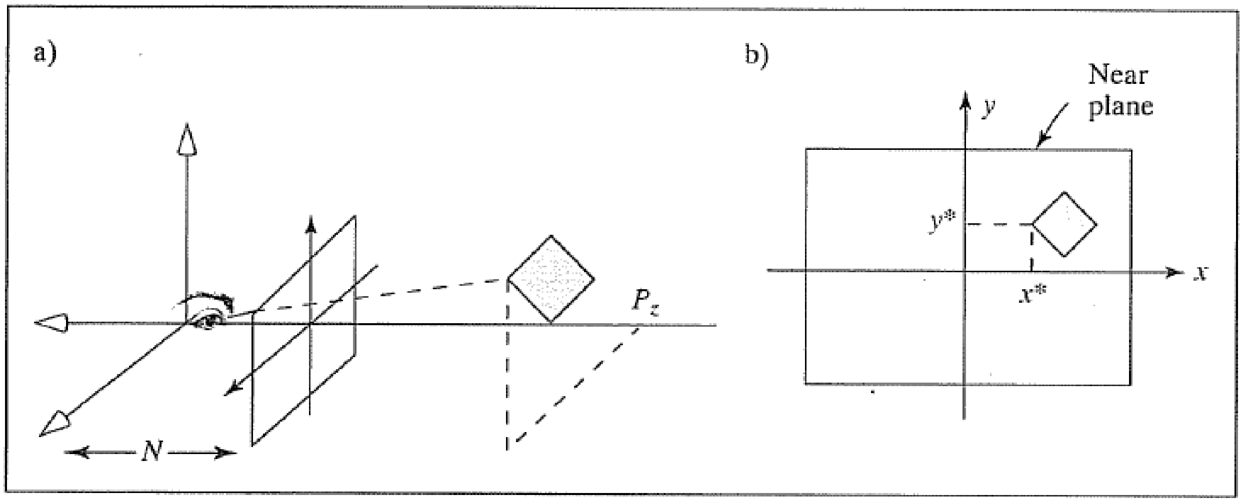


شکل ۱ نگرش جدید به خط لولهٔ OpenGL.

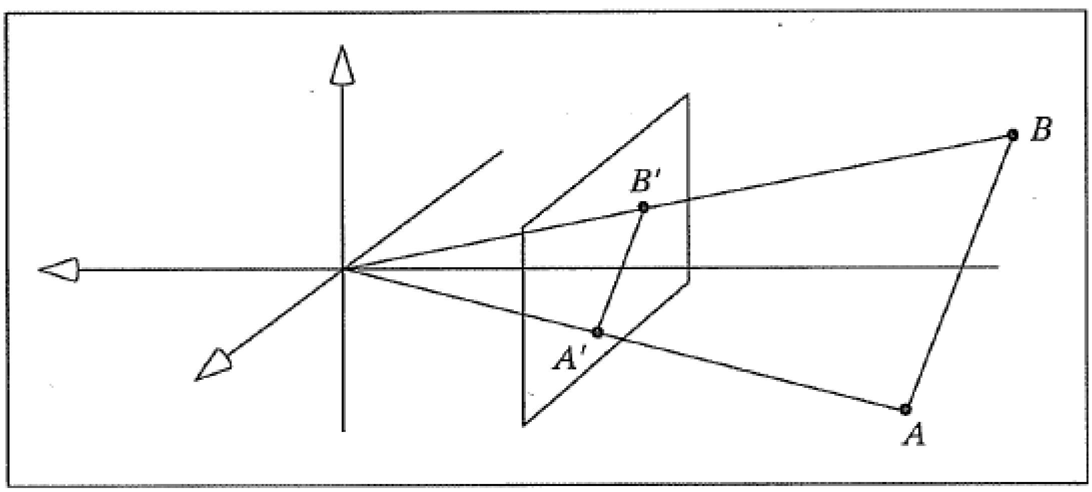


شکل ۲ افکنش پرسپکتیو نقطه ای که در دستگاه مختصات چشم بیان شده است.

$$(x^*, y^*) = \left(N\frac{P_x}{-P_z}, N\frac{P_y}{-P_z}\right) \qquad \text{(the projection of } P\text{)}$$



شکل 3 **در نظر گرفتن یک دستگاه مختصات بر روی صفحهٔ تصویر و بدست آوردن افکنش یک نقطه.**



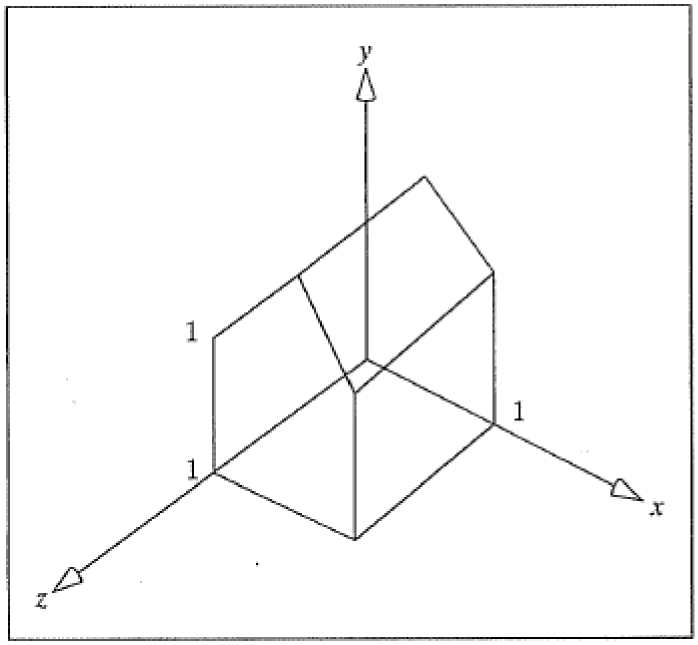
شکل 4 **افکنش یک خط راست، یک خط راست است.**

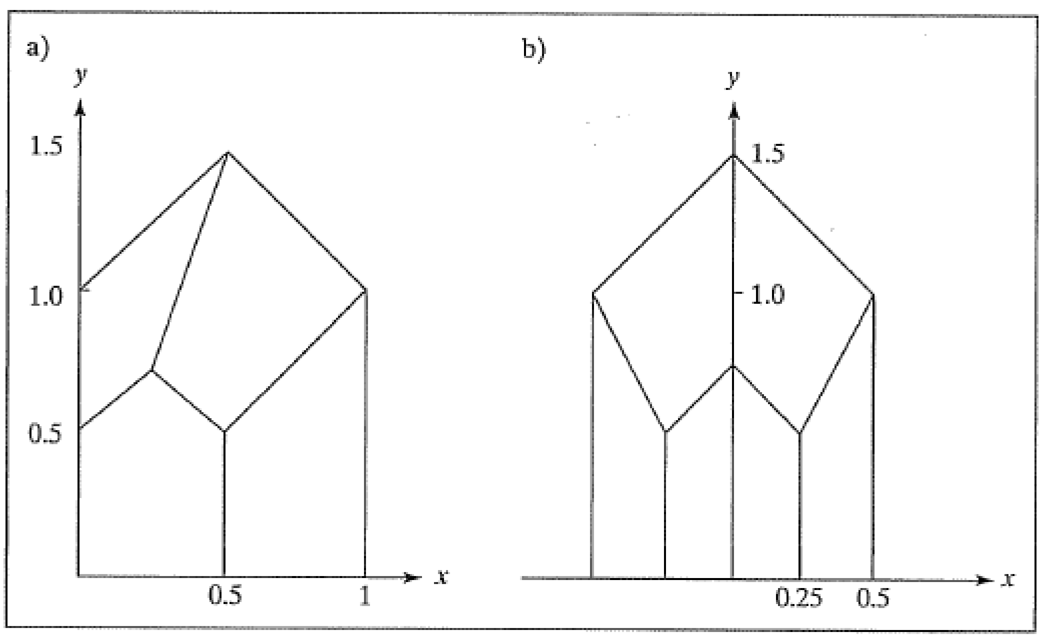
مثال: سه افکنش یک خانه

نگرش اوّل: چشم در (0,0,2) نگاه به (0,0,0)، u=(1,0,0)، صفحهٔ جلو N=1 (نسبت به چشم)

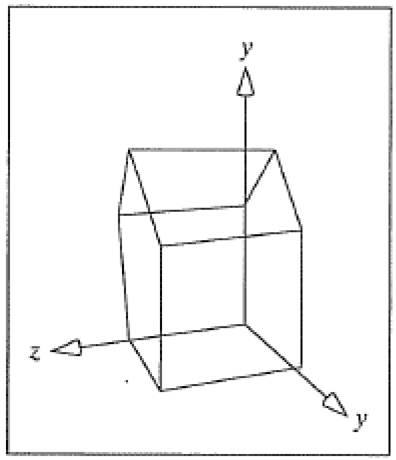
نگرش دوّم: چشم در (0.5,0,2)

نگرش سوّم: چشم در (2,5,2)





شکل ۵ افکنشهای خانه برای نگرشهای ۱ و ۲.



شکل ٦ نگرش سوّم خانه.

افکنش پرسپکتیو خط

فرض: خطی از نقطهٔ A=(Ax،Ay،Az) در جهت c=(cx،cy،cz) در مختصات دوربین داشته باشیم.

شکل پارامتری $P(t) = A + \mathbf{c}t$

شکل پارامتری افکنش خط

$$p(t) = \left(N\frac{A_x + c_x t}{-A_z - c_z t}, N\frac{A_y + c_y t}{-A_z - c_z t}\right)$$

اگر خط در فضا موازی با صفحهٔ تصویر باشد cz=0

$$p(t) = \frac{N}{-A_z}(A_x + c_x t, A_y + c_y t)$$

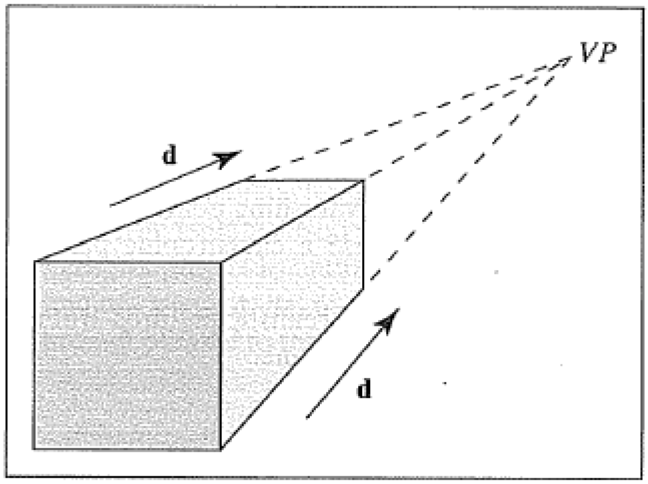
این شکل پارامتری خطی با شیب cy/cx می باشد. این شیب وابسته به مکان خط نیست. بنابر این همهٔ خطوط با شیب c با این شیب افکنده می شوند. بنابر این افکنش آنها نیز موازی است.

اگر c موازی نباشد مثلاً cz<0

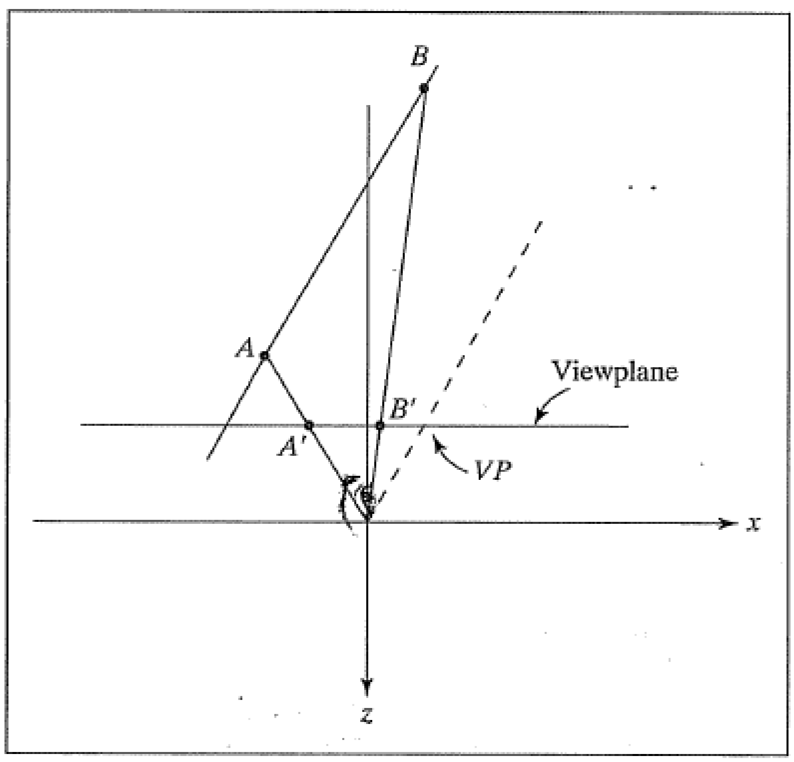
وقتی t به سمت بی نهایت برود

$$p(\infty) = \left(N\frac{c_x}{-c_z}, N\frac{c_y}{-c_z}\right) \qquad \text{(the vanishing point for the line)}$$

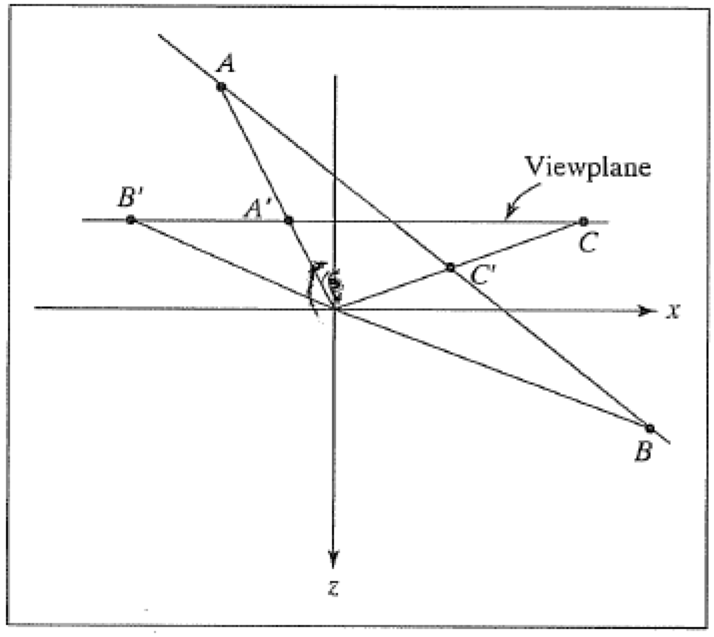
نقطهٔ محو شدن خط بدست می آید.



شکل ۷ نقطهٔ محو شدن برای خطوط موازی.



شکل 8 هندسهٔ نقطهٔمحو شدن

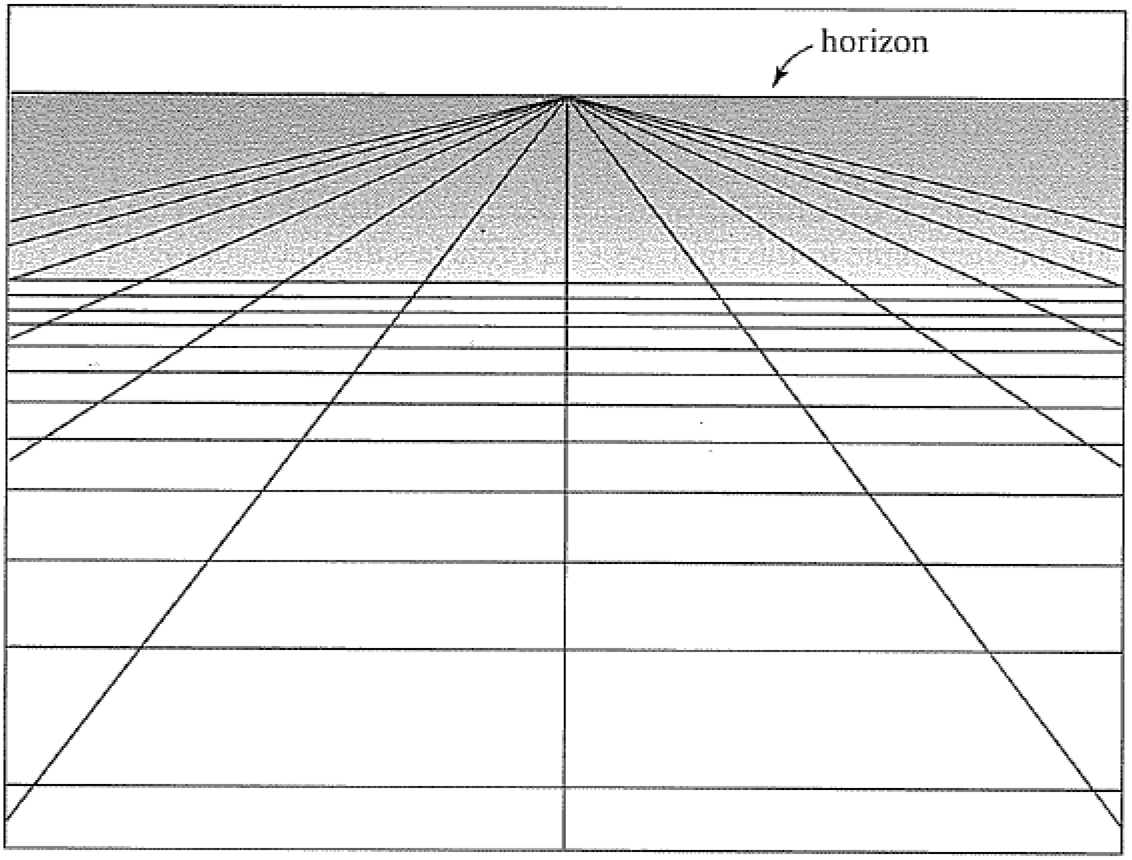


شکل 9 افکنش قطعه خط AB که B در پشت چشم قرار دارد.

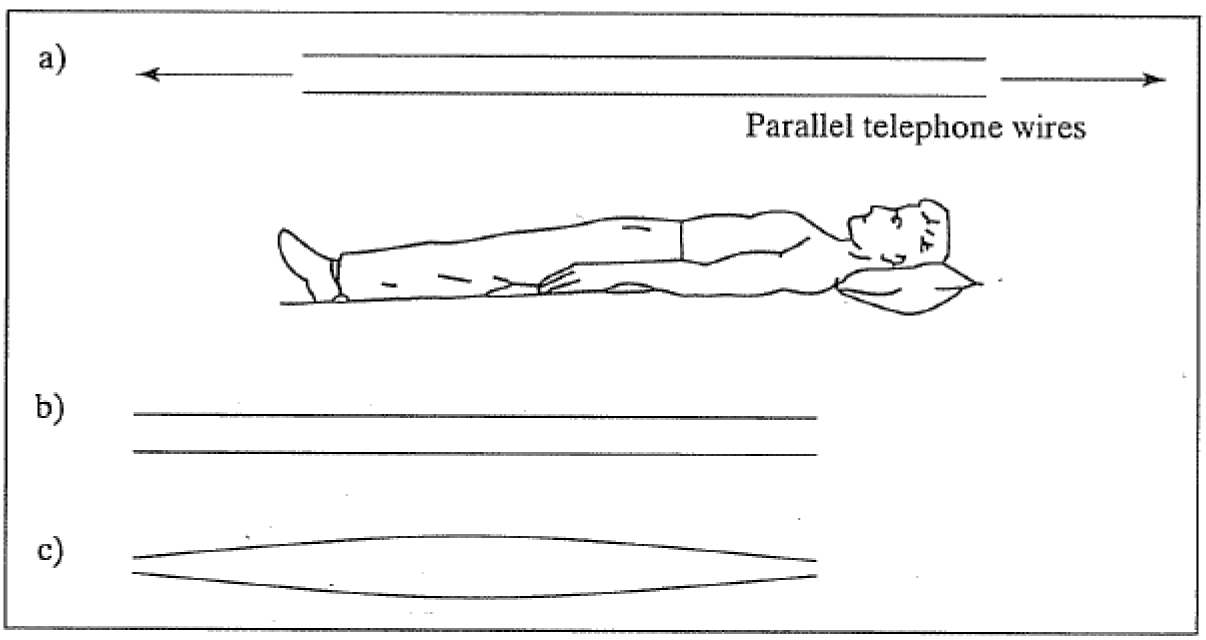
مثال: یک سری خطوط موازی در صفحهٔ xz که ا واحد نسبت به هم فاصله دارند. چشم در (0,1,0)، N=1

خطوط با x ثابت دارای شکل پارامتری در مختصات چشم بصورت (i,-1,t) که ...,i= ....,-2,-1,0,1,2 و t از صفر تا منفی بی نهایت. افکنش این خطوط (-i/t,1/t) . نقطهٔ محو شدن این خطوط در (0,0).

خطوط با z ثابت دارای شکل پارامتری در مختصات چشم بصورت (t,-1,-i) که i=1,2,...,N و t از منفی بی نهایت تا بی نهایت. افکنش این خطوط (-t/i,1/i).



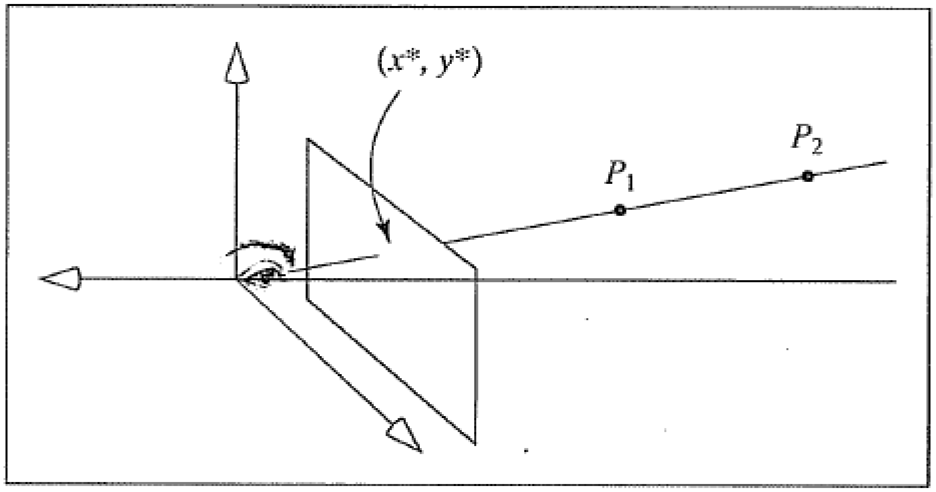
شکل ۱۰ نگرش یک شبکهٔ افقی که در صفحه xz قرار گرفته است.



شکل 11نگرش کابلهای موازی خیلی طولانی.

شبه عمق

افکنش عمق را از بین می برد. عمق واقعی نقطهٔ P برابر است با $\sqrt{P_x^2 + P_y^2 + P_z^2}$ که محاسبهٔ آن برای هر نقطهٔ مورد علاقه کند است. فقط نیاز داریم که هنگامی که دو نقطه دارای یک افکنش هستند مشخص کنیم کدام نزدیکتر است. شکل زیر دو نقطهٔ P1 و P2 را نشان می دهد که هر دو دارای افکنش یکسانی هستند. نیاز داریم که بدانیم کدامیک از این دو نقطه یکدیگر را پنهان می کنند. بنابر این برای هر نقطهٔ P که افکنش می کنیم یک شبه عمق محاسبه می کنیم.



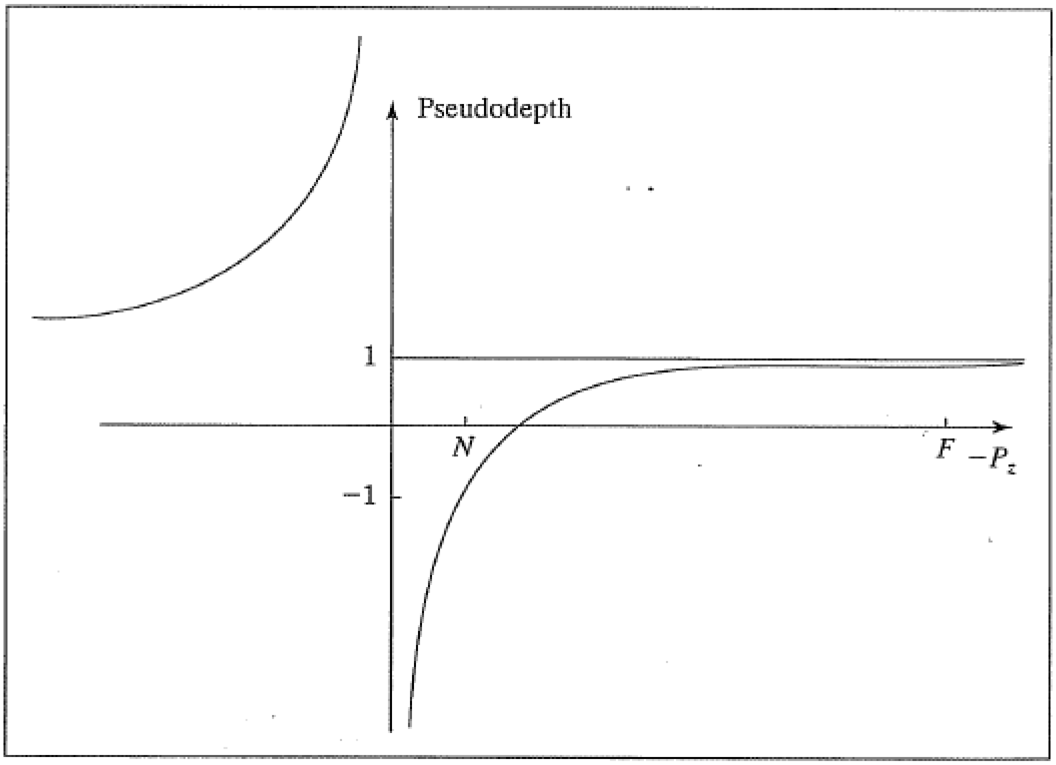
شکل 12

یک گزینه می تواند Pz– باشد. ولی بهتر است مقداری انتخاب شود که مخرجی مشابه دو مؤلفهٔ دیگر افکنش نقطه باشد.

$$(x^*, y^*, z^*) = \left(N\frac{P_x}{-P_z}, N\frac{P_y}{-P_z}, \frac{aP_z + b}{-P_z}\right)$$

a و b را به گونه ای انتخاب می کنیم که وقتی Pz=–N است مقدار ۱- و هنگامی که Pz=–F است مقدار ۱ داشته باشد.

$$a = -\frac{F+N}{F-N}, b = \frac{-2FN}{F-N}$$



استفاده از مختصات همگن

$$\begin{pmatrix} N & 0 & 0 & 0 \\ 0 & N & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & -1 & 0 \end{pmatrix} \quad \text{(the projection matrix—version 1)}$$

$$\begin{pmatrix} N & 0 & 0 & 0 \\ 0 & N & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & -1 & 0 \end{pmatrix}\begin{pmatrix} wP_x \\ wP_y \\ wP_z \\ w \end{pmatrix} = \begin{pmatrix} wNP_x \\ wNP_y \\ w(aP_z + b) \\ -wP_z \end{pmatrix}$$

نقطه در حالت دکارتی

$$\left(N\frac{P_x}{-P_z}, N\frac{P_y}{-P_z}, \frac{aP_z + b}{-P_z}\right)$$

تبدیل پرسپکتیو یک نقطهٔ سه بعدی P را به نقطهٔ سه بعدی دیگری که با معادلهٔ زیر داده می شود تبدیل می کند.

$$\left(N\frac{P_x}{-P_z}, N\frac{P_y}{-P_z}, \frac{aP_z + b}{-P_z}\right)$$

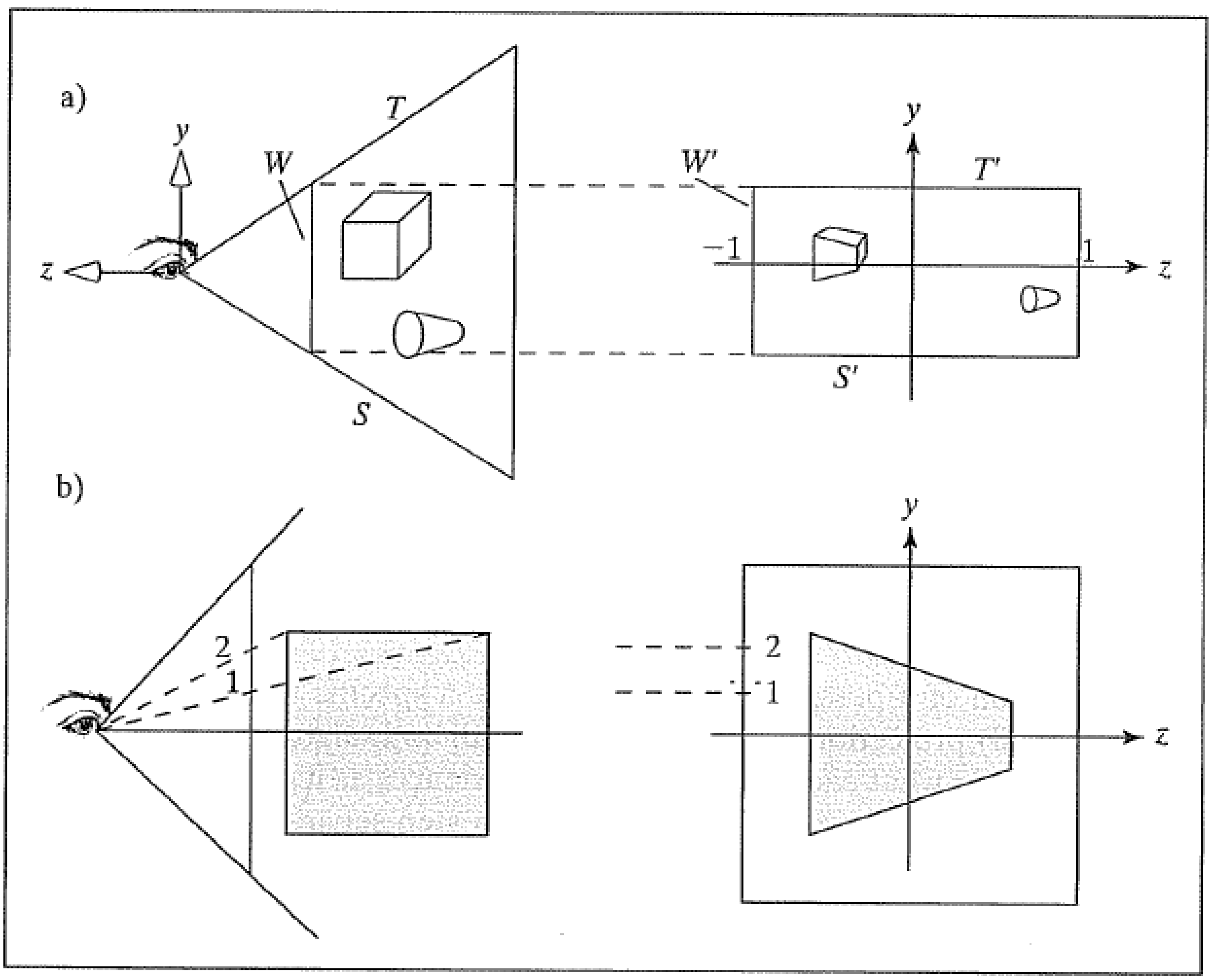
تبدیل پرسپکتیو هرم سربریدهٔ حجم دید پرسپکتیو را به یک مکعب مستطیل تبدیل می کند.

صفحه بالا به صفحهٔ y=top تبدیل می شود.

صفحهٔ کف به صفحهٔ y=bott تبدیل می شود.

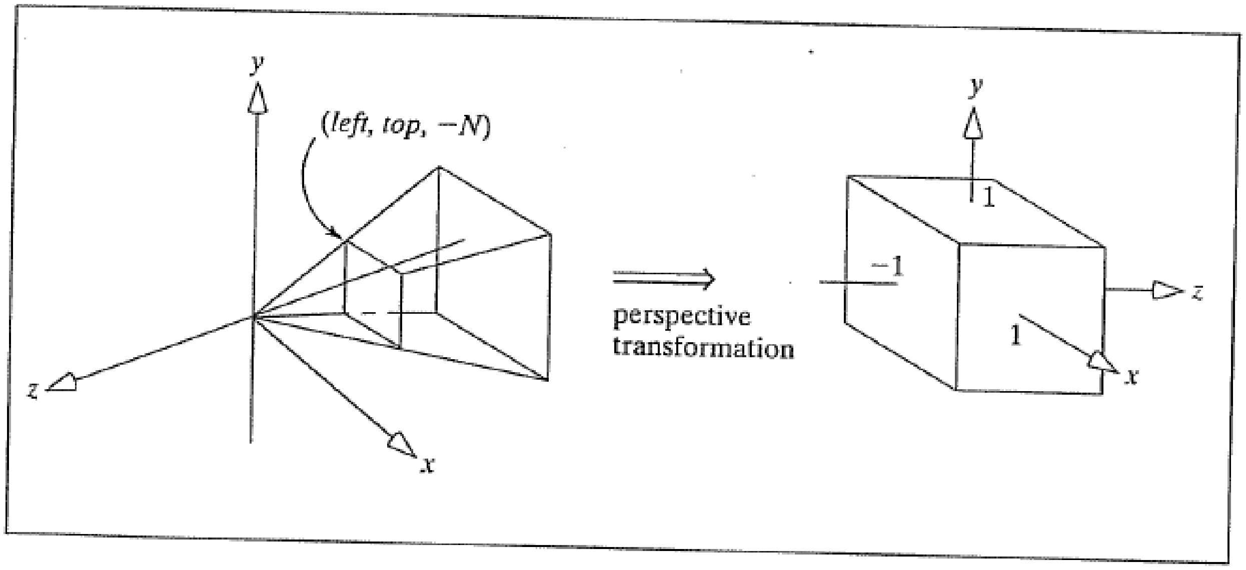
صفحهٔ چپ به صفحهٔ x=left تبدیل می شود.

صفحهٔ راست به صفحهٔ x=right تبدیل می شود.



شکل ۱۳

تبدیل به حجم دید قانونی.

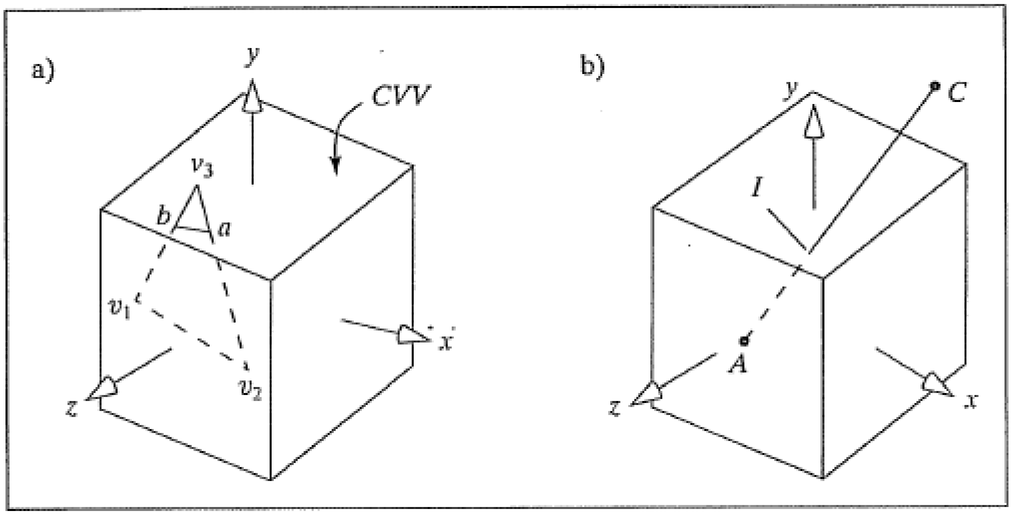


شکل 14 **جزئیات تبدیل پرسپکتیو.**

$$R = \begin{pmatrix} \frac{2N}{right - left} & 0 & \frac{right + left}{right - left} & 0 \\ 0 & \frac{2N}{top - bott} & \frac{top + bott}{top - bott} & 0 \\ 0 & 0 & \frac{-(F + N)}{F - N} & \frac{-2FN}{F - N} \\ 0 & 0 & -1 & 0 \end{pmatrix} \text{(the projection matrix)}$$

برش

پس از عبور رئوس از ماتریس افکنش عمل برش در مقابل حجم دید قانونی انجام می گیرد. حجم دید قانونی باعث می شود که عمل برش بصورت کارآتری انجام گیرد.



شکل ۱۵ برش در مقابل حجم دید قانونی.

بایستی قادر باشیم که چک کنیم که یک نقطه در داخل یا خارج یک صفحه قرار دارد. برای مثال، صفحهٔ x = -1 را در نظر بگیرید. نقطهٔ A در سمت راست (داخل) این صفحه قرار می گیرد اگر

$$\frac{a_x}{a_w} > -1 \quad \text{or} \quad a_x > -a_w \quad \text{or} \quad (a_w + a_x) > 0$$

بطور مشابه A داخل صفحهٔ x = 1 است اگر

$$\frac{a_x}{a_w} < 1 \quad \text{or} \quad (a_w - a_x) > 0$$

این شرایط شرایط مرزی نقطهٔ A نامیده می شوند و شش شرایط مرزی که وجود دارند در جدول زیر آورده شده است.

Boundary Coordinate	*Homogeneous Value*	*Clip Plane*
BC_0	$w + x$	$X = -1$
BC_1	$w - x$	$X = 1$
BC_2	$w + y$	$Y = -1$
BC_3	$w - y$	$Y = 1$
BC_4	$w + z$	$Z = -1$
BC_5	$w - z$	$Z = 1$

این شش کمیت برای A و C هر دو محاسبه می شوند. اگر همهٔ شش کمیت مثبت باشند نقطه داخل حجم دید قانونی قرار می گیرد. اگر هر یک منفی باشد نقطه در خارج CCV قرار دارد. اگر A و C هر دو داخل باشند پذیرش ساده و اگر A و C هر دو خارج باشند رد ساده انجام می گیرد. در غیر اینصورت برش خط انجام می گیرد. هر یک از شش صفحه بترتیب بررسی می شوند.

اگر لبه وارد می شود: $t_{in} = \max(old\, t_{in}, t_{hit})$

اگر لبه خارج می شود: $t_{out} = \min(old\, t_{out}, t_{hit})$

محاسبهٔ زمان برخورد لبه با یک صفحه ساده است.

$$\text{edge}(t) = (a_x + (c_x - a_a)t, a_y + (c_y - a_y)t, a_z + (c_z - a_z)t, a_w + (c_w - a_w)t)$$

برای محاسبهٔ برخورد لبه با صفحهٔ X=1 مؤلفهٔ x برابر ۱ قرار داده می شود.

$$\frac{a_x + (c_x - a_x)t}{a_w + (c_w - a_w)t} = 1$$

و از روی آن می توان t را بدست آورد.

$$t = \frac{a_w - a_x}{(a_w - a_x) - (c_w - c_x)}$$

الگوریتم Liang–Barsky

```
int clipEdge(Point4& A, Point4& C)
{
   double tIn = 0.0, tOut = 1.0, tHit;
   double aBC[6], cBC[6];
   int aOutcode = 0, cOutcode = 0;
   <.. find BC's for A and C ..>
   <.. form outcodes for A and C ..>

   if((aOutcode & cOutcode) != 0) // trivial reject
        return 0;
   if((aOutcode | cOutcode) == 0) // trivial accept
        return 1;

   for(int i = 0; i < 6; i++) // clip against each plane
   {
      if(cBC[i] < 0) // exits: C is outside
      {
       tHit = aBC[i]/(aBC[i] - cBC[i]);
       tOut = MIN(tOut,tHit);
      }
      else if(aBC[i] < 0) //enters: A is outside
      {
       tHit = aBC[i]/(aBC[i] - cBC[i]);
       tIn = MAX(tIn, tHit);
      }
   if(tIn > tOut) return 0; //CI is empty early out
}
// update the endpoints as necessary
Point4 tmp;
if(aOutcode != 0) // A is out: tIn has changed
{ // find updated A, (but don't change it yet)
   tmp.x = A.x + tIn * (C.x - A.x);
   tmp.y = A.y + tIn * (C.y - A.y);
   tmp.z = A.z + tIn * (C.z - A.z);
   tmp.w = A.w + tIn * (C.w - A.w);
}
if(cOutcode != 0) // C is out: tOut has changed
{ // update C (using original value of A)
   C.x = A.x + tOut * (C.x - A.x);
   C.y = A.y + tOut * (C.y - A.y);
   C.z = A.z + tOut * (C.z - A.z);
   C.w = A.w + tOut * (C.w - A.w);
}
A = tmp; // now update A
return 1; // some of the edge lies inside the CVV
}
```

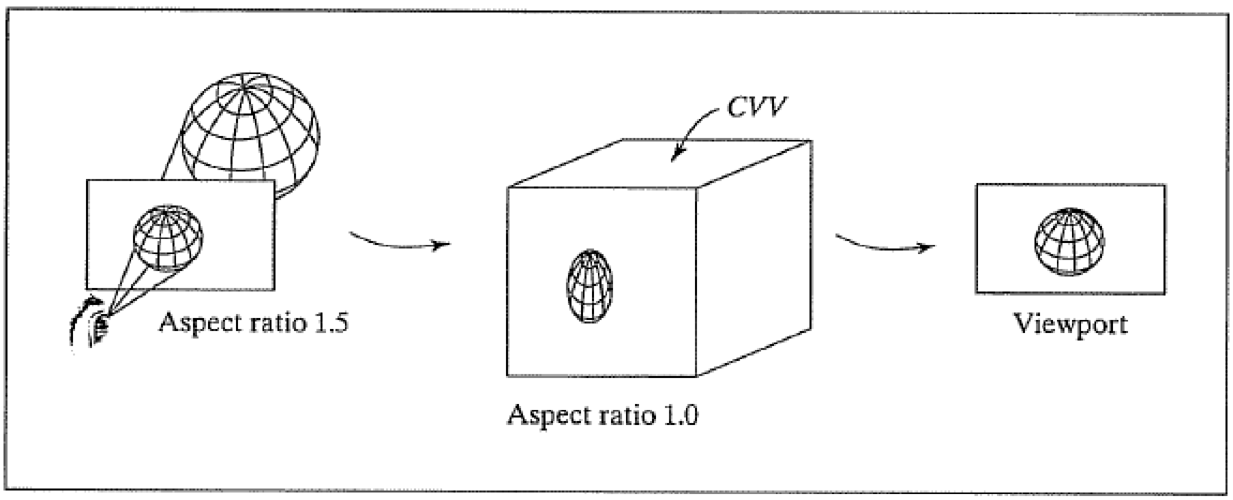
کدبرونی یک کد شش بیتی است. بیت iام کدبرونی A صفر خواهد بود اگر aBC[i]>0 و در غیر اینصورت ۱ خواهد بود.

در حلقه ای که لبه مقابل هر وجه آزموده می شود حداکثر یکی از شرایط مرزی می تواند منفی باشد. اگر A دارای یک شرط مرزی منفی باشد، لبه باید در نقطهٔ تلاقی وارد شونده باشد. اگر C دارای یک شرط مرزی منفی باشد، لبه باید در نقطهٔ تلاقی خارج شونده باشد. هر زمان که tIn یا tOut به هنگام می شود یک خروج زودرس در نظر گرفته می شود اگر tIn بزرگتر از tOut شده باشد.

هنگامی که همهٔ صفحه ها آزموده شدند یکی یا هر دوی tIn و tOut تغییر یافته اند. A به A+(C-A)tIn به هنگام سازی می شود اگر tIn تغییر کرده باشد، و C به A+(C-A)tOut به هنگام سازی می شود اگر tOut تغییر کرده باشد.

تبدیل به بندردید

پس از برش عمل تقسیم پرسپکتیو انجام می شود. بدنبال آن تبدیل به بندردید انجام می گیرد. تبدیل بندردید شبه عمق را از بازهٔ ۱- تا ۱ را به بازهٔ ۰ تا ۱ می نگارد.

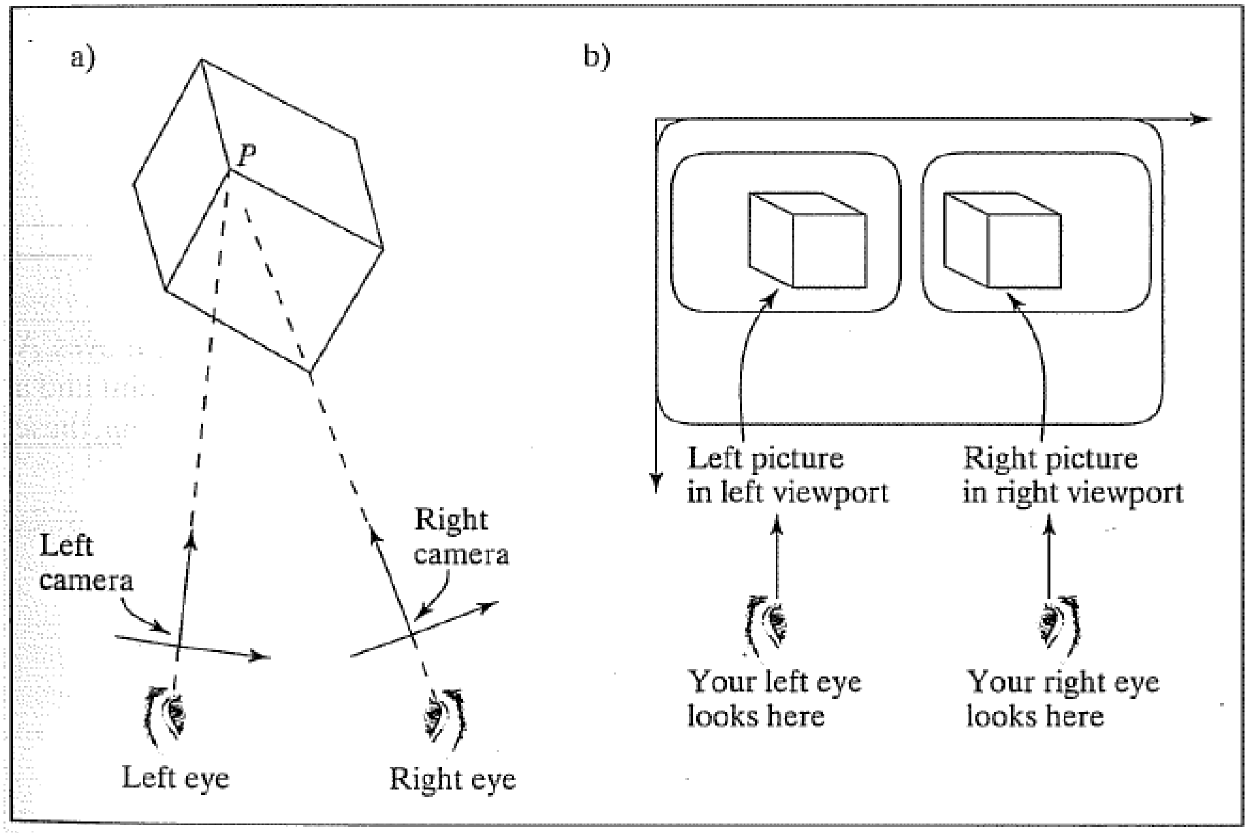


شکل 16 **تبدیل بندردید نسبتها را بازیابی می کند.**

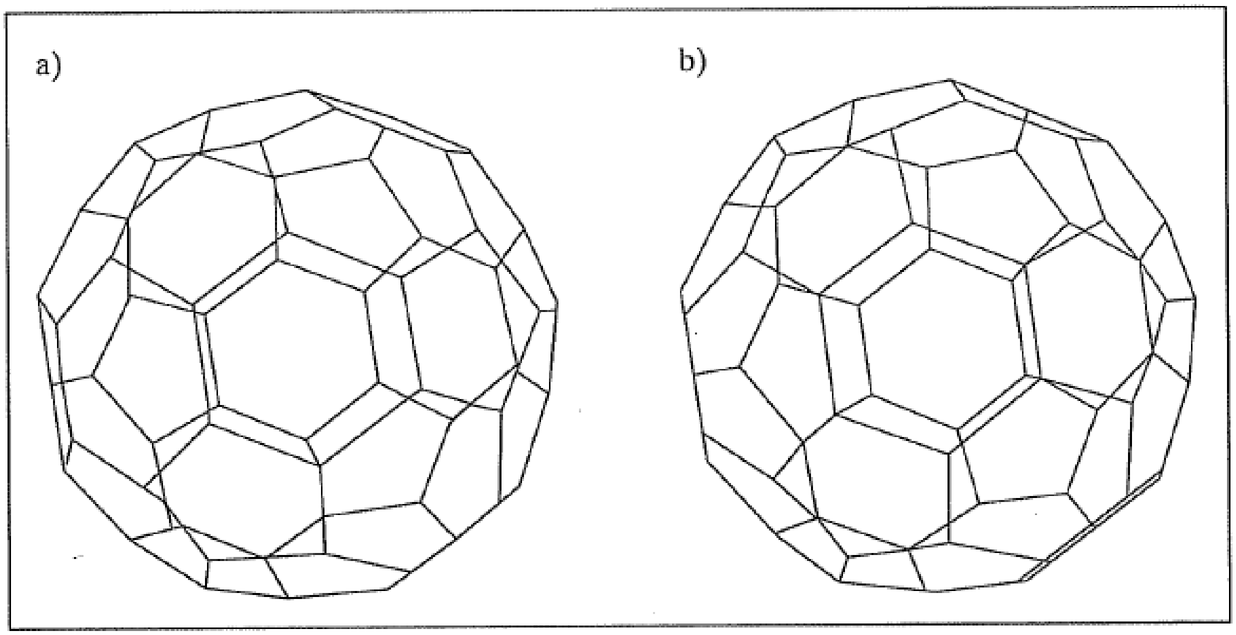
ماتریس افکنش برای افکنش موازی ارتگرافیک که توسط OpenGL استفاده می شود.

$$\begin{pmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{-2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{(OpenGL projection matrix for orthographic projection)}$$

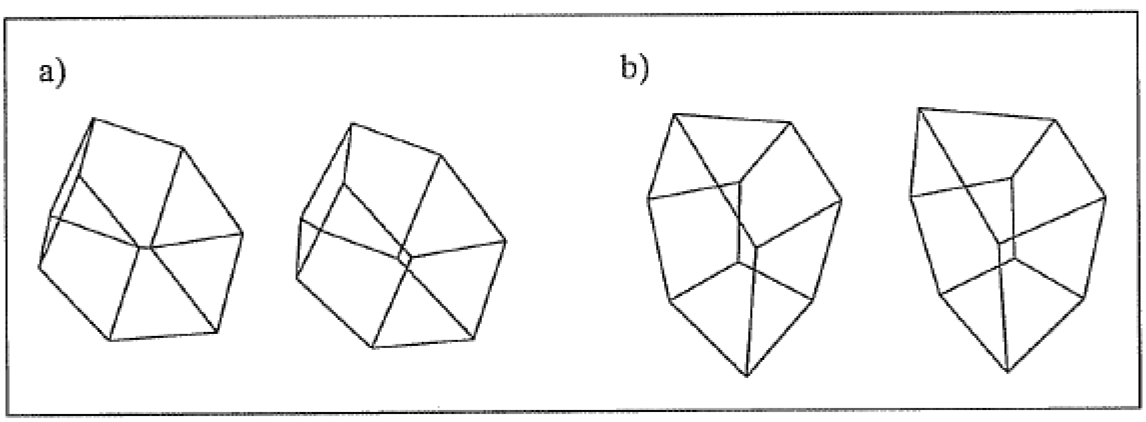
استریو



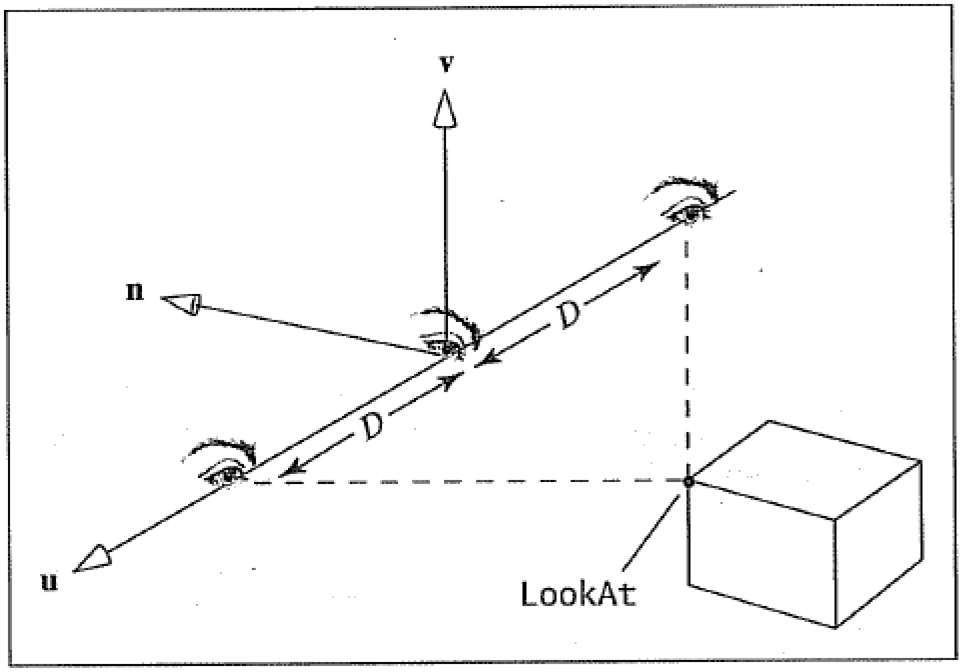
شکل 17 **ایجاد نگرش استریو.**



شکل 18 **نگرش استریو یک توپ.**



شکل 19 **دو نگرش استریو از یک خانه.**



شکل 20 **قراردهی مکانهای دو چشم برای نگرش استریو.**

تمرین

۱- تحقیق کنید که رابطهٔ زیر نشان دهندهٔ معادلهٔ خطی به شیب cy/cx می باشد.

$$p(t) = \frac{N}{-A_z}(A_x + c_x t, A_y + c_y t)$$

۲- بررسی نمائید که نسخهٔ دوّم ماتریس افکنش پرسپکتیو یک حجم دید پرسپکتیو را به یک حجم دید قانونی تبدیل می نماید.

۳- ماتریس تبدیل حجم دید موازی ارتگرافیک به حجم دید قانونی را بدست آورید.

۴- پاره خط AC را در نظر بگیرید. مختصات این دو نقطه عبارت است از: A=(-2,1.5,0) و C=(0,-2,0). کد برونی این دو نقطه را بدست آورید. سپس چگونگی برش این پاره خط توسط الگوریتم لایانگ- بارسکی را دنبال نمائید.