

# Projeto de Iniciação Científica - Relatório 1

## Visualização Unidimensional do Espaço Tridimensional

### Prólogo para Visualização Bidimensional do Espaço Quadridimensional

Paulo Roberto Rodrigues da Silva Filho

Felipe Acker (Orientador)

2023-06-28

## Contents

<b>1</b>	<b>Introdução</b>	<b>1</b>
<b>2</b>	<b>Modelo Mínimo - Ambiente Bidimensional em Tela Monodimensional</b>	<b>2</b>
2.1	Iluminação Paralela . . . . .	2
2.1.1	Modelamento de Intensidade de Iluminação . . . . .	2
2.1.2	Abertura de Visualização e Varredura de Pontos . . . . .	4
2.1.3	Identificação, Iluminação e Coloração do Pixel - cálculos . . . . .	5
2.2	Iluminação Radial . . . . .	8
<b>3</b>	<b>Modelo Ampliado (1) - Ambiente Tridimensional em Tela Monodimensional</b>	<b>9</b>
<b>4</b>	<b>Modelo Ampliado (2) - Ambiente Quadridimensional em Tela bidimensional</b>	<b>12</b>
<b>5</b>	<b>Modelo Ampliado (3) - Ambiente N-Dimensional em Tela bidimensional</b>	<b>13</b>
<b>6</b>	<b>Conclusões e Próximos Passos</b>	<b>13</b>

## 1 Introdução

Atualmente, os algoritmos de Ray-Tracing, visualização e radiosidade, para projeção de espaços 3D em 2D estão dominados e tecnologicamente avançados, já tendo mesmo implementações em Hardware, através de placas de vídeo 3D *[adicionar referências]*. Entretanto, tais técnicas implementam a representação do espaço tridimensional no espaço bidimensional, reduzindo apenas uma dimensão de representação, e apenas para o caso particular de  $\mathbb{R}^3$ . Entretanto, não há tecnologias consistentes que permitam a redução de  $\mathbb{R}^3$  para  $\mathbb{R}$ , ou de  $\mathbb{R}^4$  para  $\mathbb{R}^2$  - objetivo final desse projeto.

Assim, foi necessário desenvolver a tecnologia e os algoritmos para essas representações da estaca zero. Esse relatório visa apresentar os cálculos necessários para prover tal renderização de objetos tridimensionais, em projeção unidimensional, utilizando o modelamento físico de olho, apresentado na proposta e na renderização de objetos quadridimensionais em projeção bidimensional.

O processo de irradiação luminosa e focalização é estendido do caso tridimensional para o caso quadridimensional, enquanto o processo de captura de imagem é reduzido do caso bidimensional para o caso unidimensional, nas projeções de  $\mathbb{R}^3$  para  $\mathbb{R}$ , ou do caso tridimensional para o caso bidimensional, nas projeções de  $\mathbb{R}^4$  para  $\mathbb{R}^2$ .

Em um primeiro momento, é entendido o processo de renderização de objetos bidimensionais em uma tela monodimensional, com a utilização de sombras, com o cálculo de radiosidade, mas sem a utilização completa de *ray-tracing*, ou seja, não serão consideradas superfícies espelhadas. A partir daí, esse algoritmo é estendido para a renderização de objetos tridimensionais em tela monodimensional - que

é o objetivo planejado para esse projeto - e, por último, esse algoritmo é estendido para a renderização de objetos quadridimensionais em telas bidimensionais, o que já seria o objetivo final do projeto.

## 2 Modelo Mínimo - Ambiente Bidimensional em Tela Monodimensional

O modelo mínimo é a renderização de objetos bidimensionais em uma tela unidimensional. Esse modelo de renderização é **inferior** ao que já é implementado atualmente em software e em hardware, pelas placas 3D, mas é a base do modelo de renderização quadridimensional - portanto, entendê-lo é fundamental para a implementação de modelos de renderização que fujam do padronizado, que é a Renderização de Ambientes Tridimensionais em Telas Bidimensionais.

Para haver renderização, é necessário haver **iluminação** e **captura**. A captura é feita pelo olho, é já uma característica da renderização assumida por definição. A iluminação, apesar de obrigatória para a renderização, não é imediatamente considerada pelo senso comum. Entretanto, renderização sem um modelo de iluminação adequada apresentaria figuras achatadas e sem volume aparente, que permitiria discernir sobre as formas representadas na renderização.

Usamos dois modelos de iluminação: **(1)** Iluminação Paralela, adequada para ambientes externos e **(2)** Iluminação radial, adequada para ambientes internos. Em ambos os casos adicionamos um componente de iluminação difusa. Não é apresentado o algoritmo de *tinting*, então todas as fontes luminosas são brancas e não é apresentado o algoritmo para a utilização de mais de uma fonte luminosa. Essas omissões são corrigidas posteriormente, em uma seção específica para elas.

### 2.1 Iluminação Paralela

Na iluminação paralela de ambiente bidimensional, a fonte de luz é representada como uma reta que emite luz na direção perpendicular de sua linha - ou seja, nas suas normais, em todos os pontos. A estrutura de visualização possui seus elementos apresentando nas Figuras 1 e 2. A iluminação paralela assume que não há decaimento de intensidade, mas esse modelo não é necessariamente verdadeiro em todos os cenários. Para um melhor entendimento do modelamento da intensidade luminosa, a seção a seguir, 2.1.1, apresenta o racional físico/geométrico da intensidade, e um modelo estendido de como lidar com a intensidade luminosa.

#### 2.1.1 Modelamento de Intensidade de Iluminação

A intensidade luminosa é parte integrante do cálculo da radiosidade. No nosso mundo real (que é um mundo tridimensional), a intensidade luminosa representa a dispersão da **energia** dispersada por uma fonte de luz, no tempo (ou seja, é uma medida de potência), por uma superfície. A unidade de intensidade luminosa é a **Candela**. *[adicionar referências]*.

Uma das características fundamentais do decaimento luminoso, em função da distância, é que se assume que a luz se dispersa pelo ângulo sólido no qual ela se propaga a partir de sua fonte central. Assim, nem de uma fonte de luz paralela, nem de fonte de luz coerente (como no caso dos *lasers*), haveria decaimento de intensidade, mas em focos de luz ordinários, o decaimento seria com o quadrado da distância, pois essa é a taxa de crescimento da área da superfície do ângulo sólido, na medida em que nos distanciamos do centro de emissão da luz.

Para a renderização de imagens, a intensidade máxima de iluminação representa que nenhuma das cores da linha, superfície ou variedade iluminada possui qualquer tipo de desconto de brilho, por conta da distância da fonte de luz. Em ambientes bidimensionais, a taxa de decaimento da intensidade é dada linearmente, pois o ângulo plano representa um comprimento unidimensional. Em ambientes tridimensionais, a taxa de decaimento cai com o quadrado da distância e em ambientes quadridimensionais, estaríamos falando de um ângulo representado por uma 3-variedade e a taxa de decaimento seria com o cubo da distância.

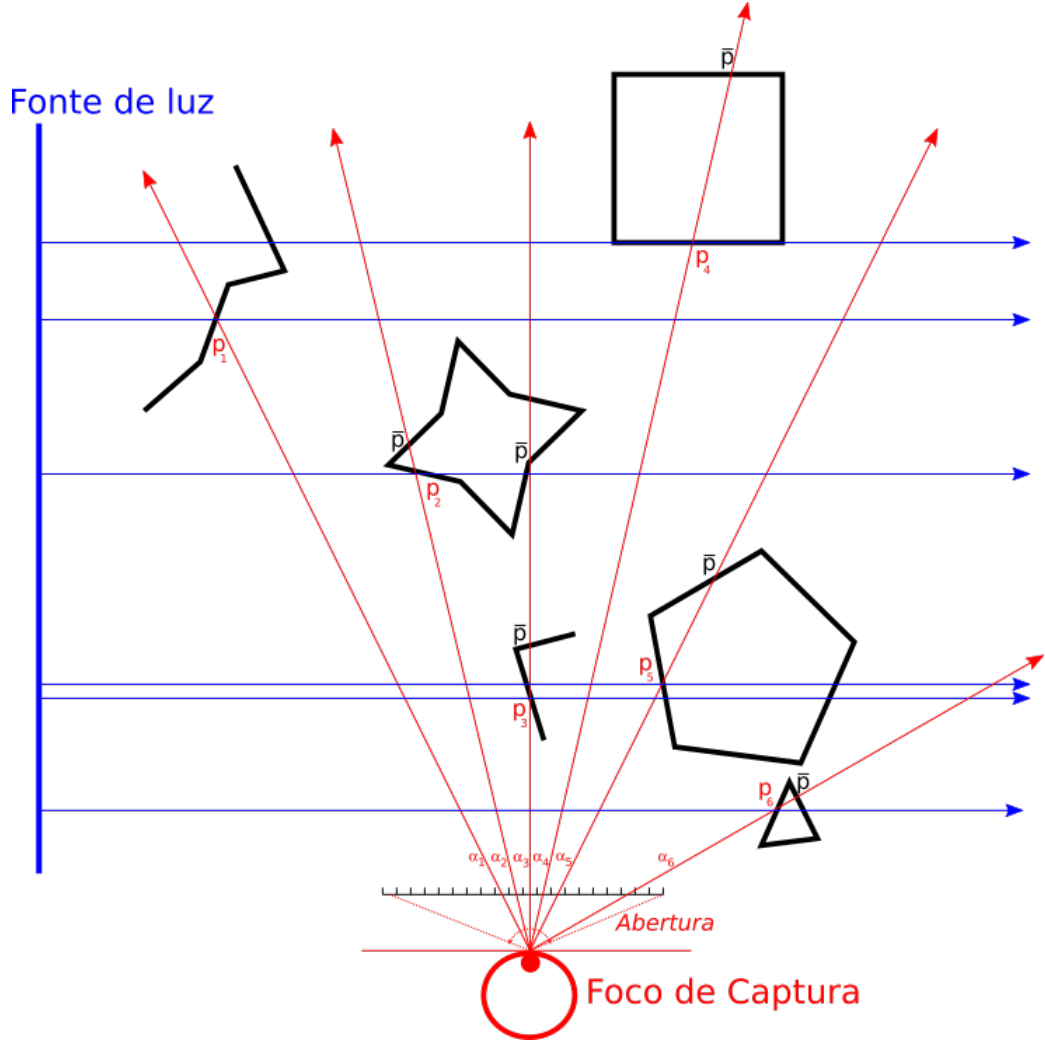


Figura 1: Identificando pontos de renderização. No inferior da figura temos o foco de captura, interceptado por uma tela unidimensional com vinte pixels de tamanho. À esquerda, temos uma fonte de luz paralela. Essa fonte de luz, na figura, está na vertical, mas qualquer direção do plano pode ser utilizada por tal fonte - desde que ela esteja longe o suficiente para nunca estar entre o foco de captura e os objetos visualizados.

De forma geral, a intensidade luminosa utilizada para a renderização de imagens pode ser dada por:

$$I(D) = \frac{k}{(1 + D)^n}, n = 1, 2, 3, \dots \quad (1)$$

Assumindo que à distância 1 (em uma unidade arbitrária), da fonte de luz, a intensidade luminosa é, por definição, igual a 1, e há o decaimento exponencial a partir daí. Se essa assunção não for feita, haveriam casos de intensidade luminosa infinita, o que não seria estritamente errado, para  $D = 0$ , mas muito inconveniente, para fins de representação.

O valor  $k$  é uma constante de intensidade arbitrária (sendo o valor padrão  $k = 2$ , enquanto o valor de  $n$ , acima, pode ser modulado, dependendo da conveniência da implementação, enquanto  $D$  é, necessariamente, a distância entre o ponto visualizado e a fonte de luz. Podemos ter universos bidimensionais que desejamos que o decaimento da intensidade luminosa não seja linear, podendo ser quadrática, para esse universo bidimensional representar um corte de um universo tridimensional maior, ou o decaimento da intensidade luminosa pode ser linear, em um universo tridimensional, no qual a intensidade deixa de ser um aspecto geométrico da luz, passando a seguir alguma lei alternativa que seja conveniente para a representação visual deste mundo. Assim, a dimensão de decaimento da intensidade luminosa fica escolhida arbitrariamente caso a caso, segundo a conveniência da implementação do mundo.

### 2.1.2 Abertura de Visualização e Varredura de Pontos

A Figura 1 mostra o exemplo da captura de seis pixels por uma tela unidimensional, em um ambiente bidimensional, com fonte de luz paralela (ou seja, luz ambiente). Esse é o modelo mínimo, renderizando um ambiente bidimensional em uma tela unidimensional. Tal exemplo é utilizado para exemplificar os seguintes conceitos:

1. Ângulo de abertura de visão - Aplicação do **Critério da Abertura**
2. Fonte de Luz - Aplicação do **Critério da Direção**
3. Radiosidade - Aplicação do **Critério da Distância e da Direção**, mas para a Fonte de Luz
4. Pontos visíveis e não visíveis - Aplicação do **Critério da Distância**

Na Figura 1 mostra-se, logo acima do foco de captura, a tela unidimensional, com vinte pixels. Cada pixel é renderizado de acordo com as interseções dos raios de visão (representados pelas setas vermelhas) com os objetos que tais raios atravessam. Cada raio renderiza apenas **o ponto mais próximo** ao foco de captura. Essa condição faz parte do **Critério da Distância**. Assim, para os raios de visão  $\alpha_1, \dots, \alpha_6$ , são renderizados os pontos  $p_1, \dots, p_6$ , na tela. Esses pontos são aqueles mais próximos do foco de captura, e o que estiver para além deles, não é renderizável - ou seja, é invisível para o Foco de Captura. Assim, todos os pontos representados por  $\bar{p}$  são invisíveis ao foco de captura e, portanto, não são renderizados.

Nas figuras 1 e 4, o foco de captura está centralizado no décimo-primeiro pixel, da esquerda para a direita. Isso é apenas para propósito de apresentação. Rigorosamente, foco deve estar localizado exatamente no meio da tela, mas isso tornaria a representação no desenho mais difícil. De toda forma, a posição da tela perante o foco de captura é totalmente arbitrário e pode mudar implementação a implementação. Nas outras figuras, a representação de tela está mais alinhada com o esperado para uma implementação real.

A renderização do ponto na tela, em si, é apenas a ativação e desativação de uma determinada cor naquele ponto de tela, ou **pixel**. O pixel não tem dimensão, e ou ele é integralmente representado, ou não - ficando apenas um ponto preto em sua posição<sup>1</sup>. Se o raio de visão identifica quais pontos da tela serão ativados ou não, qual cor será ativada depende da cor do objeto representado e da radiosidade do ponto desse objeto que está sendo representado em tela.

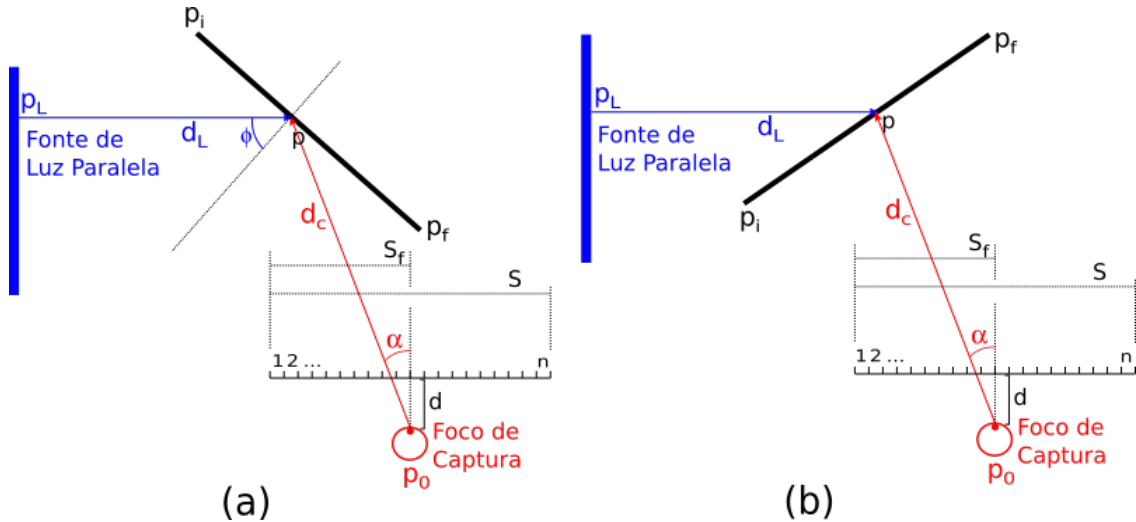


Figura 2: Identificando pontos de renderização.

<sup>1</sup>Essa regra não é tão rigorosa assim. Existem técnicas de *anti-aliasing* que podem flexibilizar um pouco essas condições - entretanto, *anti-aliasing* está além do escopo desse relatório e desse projeto.

O primeiro critério a ser considerado para haver iluminação de um ponto visível por estar diretamente exposto ao raio de visão é verificar se ele também está diretamente exposto a um raio de luz proveniente da fonte. No caso da fonte de luz paralela, considera-se uma reta paralela, mas normal à Fonte de Luz, que estará posicionada em um local infinitamente<sup>2</sup> distante, passando pelo ponto visualizado. Se esse ponto for o primeiro irradiado pelo raio de luz (ou se for a primeira interseção com algum objeto que seja atravessado por tal raio, considerando sua direção), então o ponto é iluminado. Observando a Figura 1, percebe-se que os pontos  $p_1$ ,  $p_2$ ,  $p_3$  e  $p_6$  são iluminados. Já o ponto  $p_5$  não é iluminado porque seu raio de luz é interrompido pelo objeto que contém o ponto  $p_3$ , estando, portanto, em sombra. Já o ponto  $p_4$  está em um segmento paralelo aos raios de luz, não recebendo iluminação, também. Outra justificativa de  $p_4$  não ser iluminado é porque o primeiro ponto do segmento ao qual ele pertence é que é iluminado, e todos os outros pontos depois desse, considerando a direção do raio de luz, estão em sombra.

Mas o ponto iluminado por ser o primeiro a ser irradiado precisa refletir o raio de luz recebido em direção ao foco de captura, para dizer que ele, o ponto visualizado, está sendo realmente iluminado. Assim, na Figura 2 são mostradas as duas possibilidades de identificação de se um ponto visível por estar diretamente exposto ao raio de visão receberá iluminação ou não. Em **(a)**, tanto o raio de visão, quanto o raio de iluminação estão diretamente acessando o ponto  $p$ , visualizado, e o ponto  $p_l$  e  $p_0$  estão do mesmo lado da reta  $p_i-p_f$ , que representa o objeto visualizado, que contém o ponto  $p$ . Nesse caso, o ponto  $p$  é iluminado. Em **(b)**, o ponto  $p_l$  e o ponto  $p_0$  estão em lados opostos da reta  $p_i-p_f$ . Já nesse caso, o ponto  $p$  não é iluminado, porque ele reflete a luz recebida para uma direção fora do foco de captura. Assim, o ponto  $p$  aparece em sombra.

### 2.1.3 Identificação, Iluminação e Coloração do Pixel - cálculos

Uma vez identificado o ponto  $p_i$  a ser renderizado e se ele está iluminado ou não, deve-se calcular a cor de tal ponto. Calculada a cor, ele pode ser renderizado na tela (unidimensional). Assume-se as informações iniciais da tabela 1, em notação consistente com as figuras 1, 2, 4 e 5.

**Orientação Espacial:** Uma das informações mais relevantes para todos os cálculos apresentados aqui, em todas as dimensões apresentadas, é o fato de que os eixos de representação, com origem no ponto  $p_0$ , seguem a regra de que o primeiro eixo,  $x$ , parte de  $p_0$  para o  $+\infty$ , na direção *para frente*; e, *para trás*, ele segue para o  $-\infty$ . O próximo eixo,  $y$ , ele segue para  $+\infty$  na direção *para a esquerda*, e os outros eixos ( $z$  e  $w$ ) são orientados de acordo com a **regra da mão direita**, ou seja, o eixo  $z$  é direcionado *para cima* e os outros eixos seguem a sua orientação seja ela qual for, desde que cumpram com a regra da mão direita. Assim, assumindo a existência de um produto exterior, nessa orientação, temos os vetores unitários nos eixos  $x$ ,  $y$ ,  $z$  e  $w$ , respectivamente  $e_x$ ,  $e_y$ ,  $e_z$  e  $e_w$ , respeitando:

- $e_x \rightarrow$  Orientação Positiva
- $e_y \rightarrow$  Orientação Positiva
- $e_z \rightarrow$  Orientação Positiva
- $e_w \rightarrow$  Orientação Positiva
- $e_x \wedge e_y \rightarrow$  Orientação Positiva
- $e_y \wedge e_z \rightarrow$  Orientação Positiva
- $e_x \wedge e_y \wedge e_z \rightarrow$  Orientação Positiva
- $e_x \wedge e_y \wedge e_z \wedge e_w \rightarrow$  Orientação Positiva

Todas as outras possíveis orientações podendo ser deduzidas a partir das propriedades de *Produto Exterior*. A figura 3 apresenta a orientação, seguindo essa convenção, para um espaço tridimensional.

O algoritmo de renderização deve ser repetido **para cada ponto da tela**, sendo a ordem não importante, desde que a tela como um todo seja desenhada apenas depois que todos os pixels forem renderizados. Entretanto, por uma questão prática, escolhe-se desenhar a tela começando pelo pixel de índice  $i = 1$ , e indo até o pixel de índice  $i = n$ .

---

<sup>2</sup>Na verdade, esse foco de luz paralelo precisa estar em um ponto **convenientemente** distante, caso seja aplicado qualquer critério de decaimento de intensidade para os feixes paralelos de luz, em algum "universo" ficcional simulado, no qual as fontes paralelas de luz tenham tal comportamento.

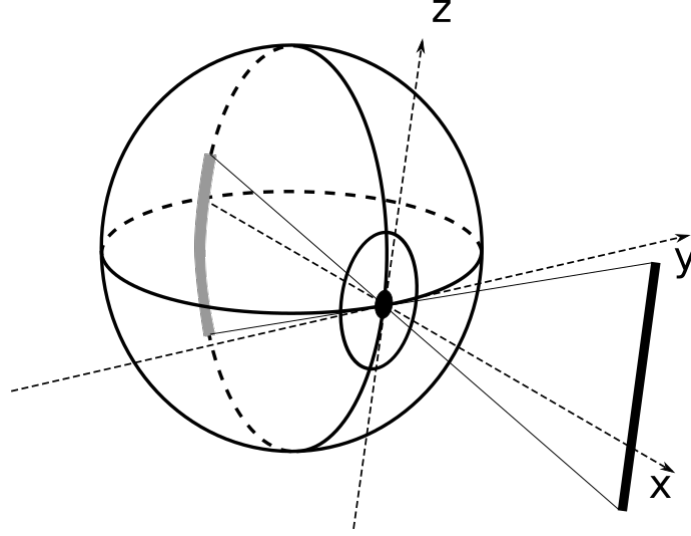


Figura 3: Representação Esquemática do Olho Tridimensional. Ver referências dessa figura na Proposta de Iniciação Científica

Então, para cada pixel  $i = 1, 2, \dots, n$ , deve-se implementar o seguinte algoritmo:

**1 - Identificação do ponto de renderização do pixel:** Cada pixel deve renderizar um ponto de acordo com o **Critério da Direção**: renderiza-se o ponto de algum segmento  $p_i-p_f$  que intercepte a reta que passa pelo centro do pixel e pela origem (o ponto  $p_0$ , ou foco de captura) e que esteja mais próximo da origem. Todos os outros pontos que cumprirem a condição de estarem na reta determinada por  $p_0$  e  $P_i$  serão ignorados.

Assim deve-se calcular:

- Reta de todos os pontos  $r_{\text{pixel}} : r_{\text{pixel}} = (P_i - P_0) \cdot t$ ,  $P_i = d \cdot e_x + S_f \cdot e_y$ ,  $S_f = \frac{S}{2} - \frac{S}{n} \cdot i_i + \frac{S}{2n}$
- Retas de todos os segmentos renderizáveis: Para todo  $p_i$ ,  $p_f$  representado no espaço a ser renderizado;  $p_i$  e  $p_f$  representando segmentos renderizáveis, calcular as suas retas  $r_{\text{segmento}} : r_{\text{segmento}} = (p_i + p_f \cdot t)$
- Pontos de interseção de  $r_{\text{pixel}}$  e de  $r_{\text{segmento}}$ . Para calcular tal interseção, basta resolver os sistemas de equação de  $r_{\text{pixel}}$  com cada  $r_{\text{segmento}}$  e verificar se o valor no eixo  $x$  de cada ponto encontrado está entre o valor de  $x_{p_i}$  e  $x_{p_f}$  correspondente a cada segmento cuja reta foi usada para se fazer tal interseção.
- Distância de cada ponto de interseção encontrado em relação à origem.

De todos os pontos identificados e com distância calculada, deve-se renderizar apenas aquele cuja distância em relação à origem for o menor. Esta é uma aplicação do **Critério da Distância**. Todos os outros pontos estão escondidos **atrás** deste.

**2 - Verificação se o ponto está iluminado ou em sombra:** Uma vez identificado o ponto, precisamos identificar o vetor que liga o Foco de Captura a esse ponto, e, também tal ponto à Fonte de Luz. Um vetor normal ao segmento,  $(p_i-p_f)^\perp$ , passando pelo ponto  $p$  identificado, também é necessário. Se os vetores  $p_0-p$  e  $p_L-p$  tiverem origem do mesmo lado do segmento  $p_i-p_f$ , então:

$$\frac{\langle p_0-p, (p_i-p_f)^\perp \rangle}{|\langle p_0-p, (p_i-p_f)^\perp \rangle|} = \frac{\langle p_L-p, (p_i-p_f)^\perp \rangle}{|\langle p_L-p, (p_i-p_f)^\perp \rangle|}$$

Se os vetores  $p_0-p$  e  $p_L-p$  estiverem do mesmo lado, ou seja, se a igualdade acima valer, então o ponto é iluminado. Caso contrário, é ponto de sombra.

Tabela 1: Conjunto de variáveis necessárias para renderizar um único pixel na tela

Variável	Descrição
$p$	Ponto Visualizado.
$p_0$	Ponto do Foco de Captura. Por definição $p_0$ é a origem do sistema de coordenadas, porque assume-se que o foco de captura é o ponto central de representação.
$p_L$	Ponto da Fonte de Luz.
$p_i-p_f$	Segmento Visualizado.
$n$	Número de Pixels da Tela.
$\alpha$	Ângulo do Pixel renderizado.
$d$	Distância da tela ao Foco de Captura.
$d_c$	Distância do ponto visualizado ao foco de captura.
$d_L$	Distância do ponto visualizado à fonte de luz.
$\phi$	Ângulo de incidência da luz na normal do Segmento Visualizado em $p$ .
$(r_i, g_i, b_i)$	Vetor de cor original de $p$ , valor intrínseco ao objeto representado pelo segmento visualizado.
$(r_f, g_f, b_f)$	Vetor de cor renderizado (final) de $p$ .
$I_L(D)$	Taxa de iluminação, dependente da distância $D$ do ponto iluminado em relação à fonte de luz.
$I_d$	Taxa de iluminação difusa, constante para todos os pontos do ambiente.
$P_i$	Pixel, de índice $i$ , a ser renderizado
$S$	Largura total da tela
$S_f$	Distância da borda da tela até a posição do foco de captura perante ela.
$i_i$	Índice do Pixel, apenas para diferenciar da unidade imaginária ( $i$ ).

Entretanto, antes de se fazer essa verificação de sinal dos vetores e da normal, deve-se verificar se o ponto  $p$  em questão é o primeiro a ser interceptado, a partir da Fonte de Luz, na reta  $p_0-p_L$ . Se esse não for o caso, como, por exemplo, os dos pontos  $p_4$  e  $p_5$ , da Figura 1, o ponto é, automaticamente, um ponto de sombra, e nenhum cálculo adicional é necessário.

**3 - Cálculo da cor do pixel:** Se foi identificado se o pixel está representando um ponto iluminado ou um ponto de sombra, é necessário, então, definir qual cor o pixel irá assumir, e após essa definição, renderizar o pixel é apenas colori-lo integralmente com essa cor.

As duas primeiras variáveis a serem calculadas são:

A taxa de iluminação total do ponto renderizado:

$$I_t = \begin{cases} \min(I_d + I_L(D_L), 1), & D_L = \text{dist}(p, F_L) = \text{dist}(p, p_L) \quad \text{se o ponto está iluminado} \\ I_d & \text{se o ponto é de sombra} \end{cases} \quad (2)$$

Sendo  $p_L$  o ponto da fonte de luz paralela  $F_L$ , que pode ser modelada como uma reta passando por  $p_L$  (reta infinita, não apenas segmento), e  $p_L$  pertencendo à reta  $p-p_L$  perpendicular a  $F_L$ , lembrando que é  $F_L$  que está fixa no espaço, não  $p-p_L$ , e, da equação 1, assumindo  $n = 2$  e  $k = 2$ :

$$I_L(D_L) = I(\text{dist}(p, p_L)) = \frac{2}{(1 + \text{dist}(p, p_L))^2} \quad (3)$$

A distância  $d(p, p_L)$  deve ter uma unidade de distância qualquer, mas, aqui, vamos considerar que existe uma unidade padrão e que as fórmulas todas apresentadas estão nessa unidade padrão.

E a intensidade de renderização na tela, que segue a mesma fórmula de intensidade da iluminação:

$$I_R \equiv I_D, I_R(D_0) = I(\text{dist}(p, p_0)) = \frac{2}{(1 + \text{dist}(p, p_0))^2} \quad (4)$$

Uma vez que temos a Intensidade de Iluminação e a Intensidade de Renderização, podemos definir a cor final do pixel:

$$(r_f, g_f, b_f) = I_t \cdot I_R(D_0) \cdot (r_i, g_i, b_i), \text{ para cada ponto } p \text{ na tela.} \quad (5)$$

Lembrando que  $0 \leq r_i < 256$ ,  $0 \leq g_i < 256$  e  $0 \leq b_i < 256$ .

## 2.2 Iluminação Radial

Para a Fonte de Luz pontual, que permite uma iluminação radial, segue exatamente os mesmos princípios e cálculos de identificação dos pixels a serem renderizados, da sua intensidade luminosa e decaimento de intensidade de representação.

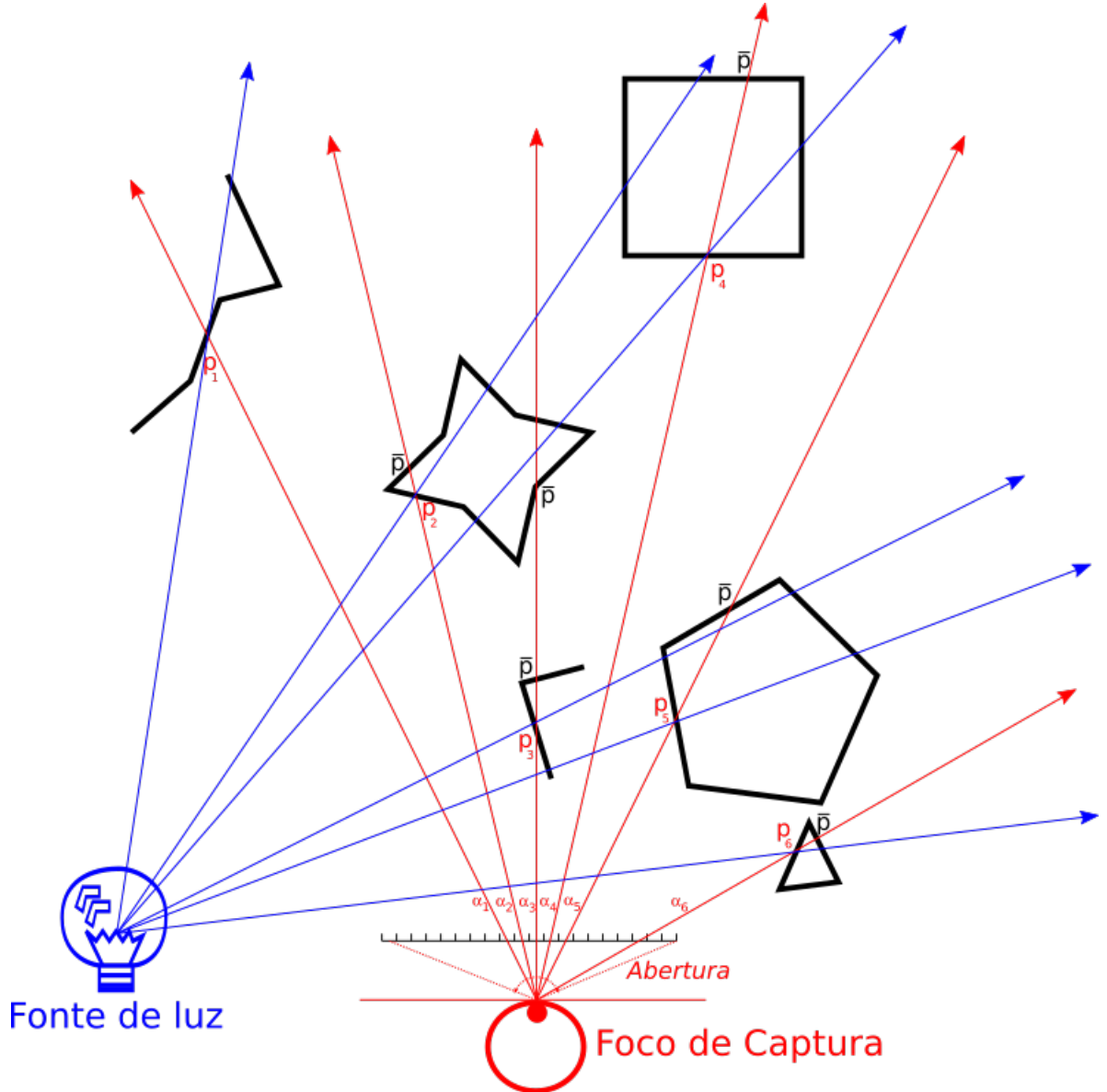


Figura 4: Identificando pontos de renderização.

A única diferença é que  $p_L$ , ao invés de estar em uma reta infinita representando a fonte de luz, está em uma posição específica no espaço, e a distância  $p-p_L$  é a distância do ponto  $p$  ao próprio ponto  $p_L$ , já previamente definido, enquanto no caso da iluminação paralela,  $p_L$  precisava ser tal que a reta  $p-p_L$  fosse perpendicular à reta da fonte de luz.



Sendo assim, nenhuma informação adicional a respeito dos procedimentos para renderização nesse caso serão apresentados, pois seria pura redundância.

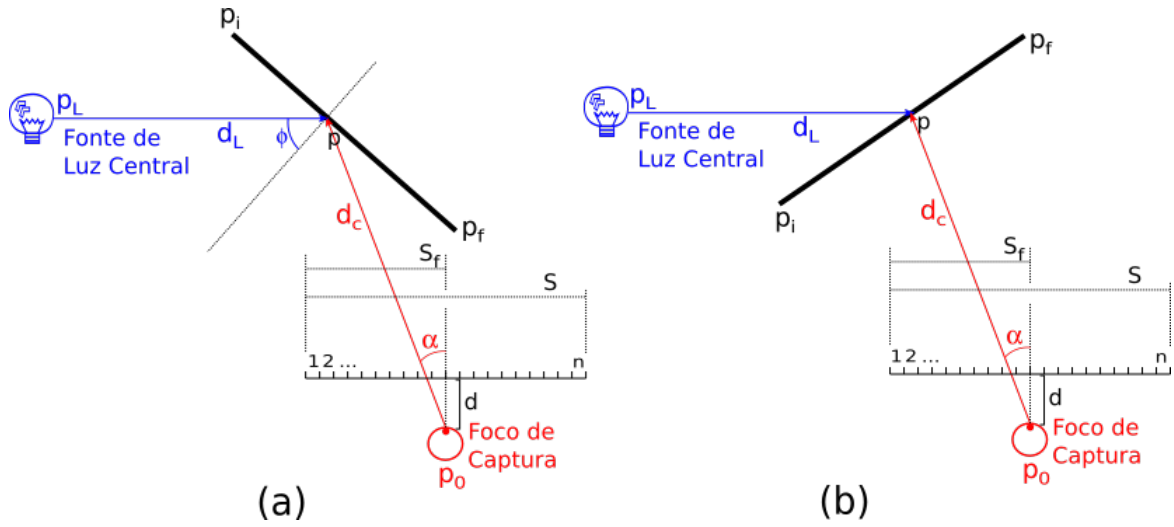


Figura 5: Identificando pontos de renderização.

### 3 Modelo Ampliado (1) - Ambiente Tridimensional em Tela Monodimensional

A apresentação do Ambiente Bidimensional em Tela Monodimensional teve, como propósito, apenas lançar as bases para a representação de ambientes de dimensão mais alta em telas de dimensão inferiores, a uma diferença superior a um, como o caso desse modelo aqui apresentado.

No caso da representação de  $\mathbb{R}^3$  em projeção em  $\mathbb{R}$ , a perda de informação é ainda mais severa do que a da projeção de  $\mathbb{R}^2$  em  $\mathbb{R}$  ou de  $\mathbb{R}^3$  em  $\mathbb{R}^2$ . Além disso, pode haver sobreposição de representações, o que leva a algumas escolhas bastante severas a respeito de que informação iremos renunciar, para a imagem renderizada ainda fazer sentido. Através da **navegação**, entretanto, podemos definir mecanismos de recuperação da informação perdida.

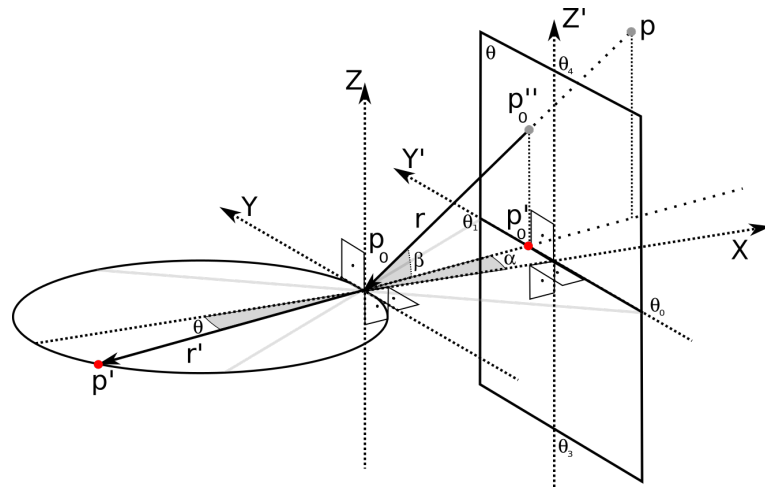


Figura 6: Captura de vetores de luz pelo olho bidimensional em ambiente tridimensional

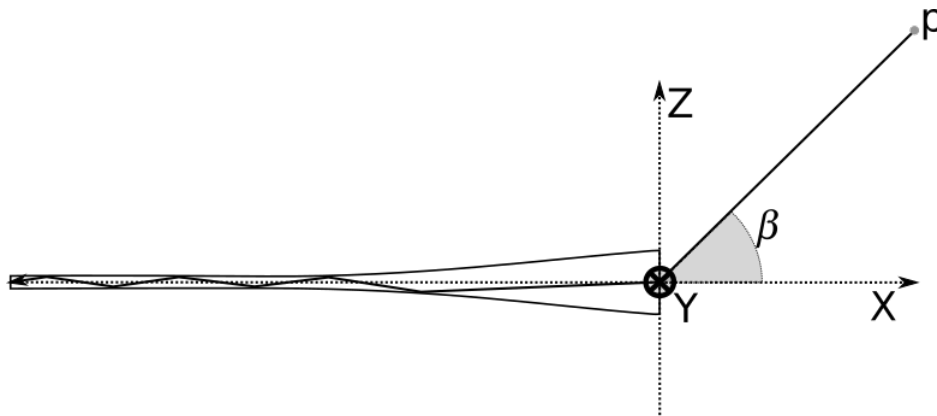


Figura 7: Exemplo de captura de luz de um ambiente tridimensional, para um bidimensional, em um material de altíssima refração.

Mas, faz sentido pensar em um ser bidimensional, com visão monodimensional, imerso em um espaço bidimensional? Podemos afirmar que sim, porque a visão dos animais é baseada na sensibilização de pontos (os cones e os bastonetes), organizados, dentro do olho, na forma de um reticulado. Se esse reticulado é bidimensional ou tridimensional, depende-se da conformação fisiológica do ser em particular.

O modelamento da renderização de um pixel em uma tela monodimensional em um ambiente tridimensional e, então, estender esse modelo para dimensões superiores, segue o que é apresentado na Figura 6 abaixo, que representa o olho bidimensional imerso no ambiente tridimensional.

A pergunta que não quer calar é se é possível para um ser bidimensional, imerso em um espaço tridimensional, enxergar figuras tridimensionais. Essa pergunta é bastante inadequada, porque em um ambiente tridimensional não é possível ter seres bidimensionais: o que é possível é ter seres achatados, cujos órgãos de visão são estruturados bidimensionalmente. É fisicamente factível a existência de uma retina bidimensional em um ser tridimensional achatado, conforme pode ser visto na Figura 7, que apresenta o corte de um olho bidimensional, feito com material transparente de alta refração, que pode capturar todos os raios de luz que atingem o seu ponto de captura luminosa e os direciona para um objeto matemático com uma dimensão a menos.

Na Figura 7, o ponto de entrada de luz é o representado pelo encontro dos eixos  $\mathbf{X}$ , que aponta para a direita,  $\mathbf{Y}$ , que aponta para dentro do papel, e  $\mathbf{Z}$ , que, para respeitar a orientação da Regra da Mão Direita, aponta para cima. O raio de luz incide no corpo do olho com um ângulo de incidência  $\beta$  e, como o olho é feito com material de alta refração, uma parte da luz é refletida, e uma parte, com a intensidade igual à intensidade original multiplicada pelo cosseno do ângulo de incidência  $\beta$  é capturado e retido dentro da estrutura do olho pelas propriedades da reflexão total.

Sabendo que há justificativa física para a construção de olhos que reduzem a dimensão do objeto de projeção - que no caso de um ambiente 3D seria uma linha, no caso de um ambiente 4D um plano, ou uma superfície bidimensional, e assim por diante - devemos, agora, avaliar como representar os pontos capturados por essa *retina* unidimensional em um ambiente tridimensional.

Observando a Figura 6, vamos dar atenção ao ponto  $p$ , que é o ponto capturado. Para esse ponto ser capturado e impresso na tela, são utilizados os mesmos critérios apresentados para a Tela Unidimensional em um Ambiente Bidimensional:

**1 - Identificação do ponto de renderização do pixel:** Aqui, tanto quanto no caso do ambiente bidimensional, deve-se respeitar o **Critério da Direção**. Aqui temos a representação da tela como uma linha no espaço, entre  $\theta_0$ - $\theta_1$ , conforme a Figura 6. Entretanto, os raios que chegam ao ponto de captura do olho,  $p_0$ , não são apenas os que residem no plano X-Y. Se fosse assim, a representação do espaço por esse olho bidimensional não seria possível, e cairíamos no mesmo caso apresentado para o olho bidimensional

Tabela 2: Conjunto de variáveis necessárias para renderizar um único pixel na tela

Variável	Descrição
$n_Y$	Número de Pixels da Tela, número de pontos do reticulado de representação da abertura de visão, em Y.
$n_Z$	Número de Pixels da Tela, número de pontos do reticulado de representação da abertura de visão, em Z.
$(r_{ij}, g_{ij}, b_{ij})$	Vetor de cor original de $p$ , valor intrínseco ao objeto representado pelo segmento visualizado.
$(r_{fj}, g_{fj}, b_{fj})$	Vetor de cor renderizado (final) do subpixel $p_j$ .
$(r_f, g_f, b_f)$	Vetor de cor renderizado (final) de $p$ .
$I_L(D)$	Taxa de iluminação, dependente da distância $D$ do ponto iluminado em relação à fonte de luz.
$I_d$	Taxa de iluminação difusa, constante para todos os pontos do ambiente.
$P_i$	Pixel, de índice $i$ , a ser renderizado
$P_{ij}$	Componente do Pixel de índice $i$ , com subíndice $j$ , a ser renderizado
$S_Y$	Largura total da tela, que representa a largura total de visão em Y
$S_Z$	Largura total da abertura de visão em Z
$S_{fY}$	Distância da borda da abertura de visão, em Y, até a posição do foco de captura perante ela.
$S_{fZ}$	Distância da borda da abertura de visão, em Z, até a posição do foco de captura perante ela.
$i_i$	Índice do Pixel, apenas para diferenciar da unidade imaginária ( $i$ ).
$i_{ij}$	Índice do Componente do Pixel, subindexado em $j$ .

em ambiente bidimensional. Para cada pixel da tela  $\theta_0$ - $\theta_1$ , há um reticulado de pontos renderizáveis que variam em  $\mathbf{Z}$ , entre as coordenadas  $\theta_3$  e  $\theta_4$ , também representados na figura 6, no plano de projeção  $\theta$ .

No Modelo Mínimo, apresentado acima, há a construção de um reticulado da tela, um ponto para cada pixel, e as retas de representação variando um ângulo  $\alpha$ , até cobrir toda a abertura de visão. Agora, temos o mesmo ângulo  $\alpha$ , cobrindo a abertura em  $\mathbf{Y}$  e um ângulo  $\phi$ , cobrindo a abertura em  $\mathbf{Z}$

Então, devemos considerar as novas variáveis apresentadas na Tabela 2, para calcular:

- Reta de todos os pontos  $r_{\text{subpixel}_{ij}} : r_{\text{subpixel}_{ij}} = (P_{ij} - P_0) \cdot t$ ,  $P_{ij} = d \cdot \mathbf{e}_x + S_{fY} \cdot \mathbf{e}_y + S_{fZ} \cdot \mathbf{e}_z$ ,  $S_{fY} = \frac{S_Y}{2} - \frac{S_Y}{n_Y} \cdot i_{ij} + \frac{S_Y}{2n_Y}$ ,  $S_{fZ}$  calculado de forma correspondente.
- Triângulos de todas as superfícies renderizáveis: Objetos tridimensionais não são representados por seus segmentos delimitadores - mas sim, pelos suas superfícies delimitadoras, que, por sua vez, são simplificadas para triângulos, com três pontos:  $p_1$ ,  $p_2$  e  $p_3$ , no lugar de  $p_i$  e  $p_f$ . Então, para todo  $p_1$ ,  $p_2$ ,  $p_3$  representado no espaço a ser renderizado;  $p_1$ ,  $p_2$ ,  $p_3$  representando segmentos renderizáveis, calcular os seus planos correspondentes  $r_{\text{plano}} : r_{\text{plano}} = (p_1 + p_2 \cdot t + p_3 \cdot s)$ ,  $t, s \in \mathbb{R}$
- Pontos de interseção de  $r_{\text{subpixel}_{ij}}$  e de  $r_{\text{plano}}$ . Para calcular tal interseção, basta resolver os sistemas de equação de  $r_{\text{subpixel}_{ij}}$  com cada  $r_{\text{plano}}$  e verificar se o valor no eixo  $(x, y, z)$  de cada ponto encontrado está entre os pontos  $p_1$ ,  $p_2$ ,  $p_3$ . Para fazer essa verificação, diversas técnicas existem, sendo uma das mais eficientes computacionalmente, definir vetores para dois lados e para para o ponto em questão, iniciando em um vértice particular, e verificar se a orientação desses vetores, em relação a um vetor inicial é sempre o mesmo. Por exemplo, podemos fixar o vértice  $p_1$ , gerar os vetores  $\mathbf{a} = p_1 - p_2$ ,  $\mathbf{b} = p_1 - p_3$  e  $\mathbf{c} = p_1 - p$ . Se  $\mathbf{a} \wedge \mathbf{b}$  tiver a mesma orientação de  $\mathbf{a} \wedge \mathbf{c}$ , então o ponto  $p$  está no ângulo interno de  $p_2 - p_1 - p_3$ . Se isso for verdade se fixarmos os vértices também em  $p_2$  e  $p_3$ , então  $p$  está dentro do triângulo e o ponto pode ser renderizado. Esse método pode ser generalizado para verificar pontos internos a tetraedros em  $\mathbb{R}^4$ , ou em figuras correspondentes em dimensões superiores.
- Distância de cada ponto de interseção encontrado em relação à origem.

**2 - Verificação se o ponto está iluminado ou em sombra:** A verificação se um ponto é iluminado ou não depende de se verificar se o vetor de iluminação está do mesmo lado da superfície renderizada ou não. A fórmula para tal verificação é estritamente a mesma do caso bidimensional, o que muda é a definição do vetor perpendicular, agora em relação à superfície  $p_1-p_2-p_3$ , não ao segmento  $p_i-p_f$ . Assim:

$$\frac{\langle p_0-p, (p_1-p_2-p_3)^\perp \rangle}{|\langle p_0-p, (p_1-p_2-p_3)^\perp \rangle|} = \frac{\langle p_L-p, (p_1-p_2-p_3)^\perp \rangle}{|\langle p_L-p, (p_1-p_2-p_3)^\perp \rangle|} \quad (6)$$

Onde:

$$(p_1-p_2-p_3)^\perp = p_1-p_2 \wedge p_1-p_3 \quad (7)$$

**3 - Cálculo da cor do pixel:** Não se calcula diretamente a cor do píxel, mas a do subpixel, para cada ponto do reticulado de representação - a superfície  $\theta$  - com um determinado valor de  $\mathbf{Y}$ , para todos os valores de  $\mathbf{Z}$ , e para cada subpixel, o valor é dado por:

$$(r_{fj}, g_{fj}, b_{fj}) = I_t \cdot I_R(D_0) \cdot (r_i, g_i, b_i) \cdot \cos(\beta), \text{ para cada subixel } p_j \text{ na abertura de visão.} \quad (8)$$

Conforme as mesmas regras do cálculo de cor para o Modelo Mínimo, apresentado acima. O valor  $\cos(\beta)$ , acima, representa o enfraquecimento do vetor de luz provocado pela inclinação  $\beta$  desse vetor em relação ao plano  $\mathbf{X-Y}$  onde reside a retina bidimensional - ou seja, a inclinação do vetor de luz no eixo suprimido ( $\mathbf{Z}$ , nesse caso), provoca o enfraquecimento do sinal do pixel, dando maior ênfase de representação para os pontos mais alinhados ao plano do olho.

Entretanto, pode-se verificar, aqui, que cada pixel da tela irá renderizar todos pontos que possuem uma coordenada  $\mathbf{Y}$  em comum - ou seja, para todos os pontos que se projetarem em  $\alpha$  em uma determinada coordenada ( $Y', Z'$ ), todos os pontos com o mesmo valor de  $\mathbf{Y}$  vão ser renderizados no mesmo ponto. Para isso ser possível, todos esses pontos serão combinados utilizando-se a técnica de *alpha-blending*. A combinação de todos esses pontos representa o que vai ser visualizado naquele ponto da tela (unidimensional no caso apresentado nessa seção).

O valor da cor do ponto  $P_i$  é dado por:

$$(r_f, g_f, b_f) = \text{Alpha-Blending}((r_{f0}, g_{f0}, b_{f0}), \dots, (r_{fnz}, g_{fnz}, b_{fnz})) \quad (9)$$

A forma para o cálculo do *alpha blending* não é dada aqui, e pode ser vista nas referências [adicionar referência].

Esse modelo de representação do espaço tridimensional em uma linha pode trazer dificuldades, porque, nesse contexto, o conceito de *altura* não existe, então um objeto representado acima do horizonte de visão não é distinguível de um objeto representado abaixo do horizonte de visão. Uma forma de resolver esse problema é restringir a abertura de visão apenas acima ou abaixo do horizonte. Outras formas de correção desse problema são possíveis, mas a sua apresentação está fora do escopo desse documento.

## 4 Modelo Ampliado (2) - Ambiente Quadridimensional em Tela bidimensional

O modelo ampliado representando um ambiente quadridimensional em tela bidimensional segue exatamente o modelo de representação de ambiente tridimensional em tela unidimensional, no qual todos os pontos que possuem um mesmo componente  $\mathbf{Y}$ , colapsando nesse ponto, todos os pontos do reticulado de representação  $\theta$ , para todos os valores de  $\mathbf{Z}$ .

Para a representação do Ambiente Quadridimensional em Tela bidimensional, a tela possui uma altura e largura, e os pixels indexados para  $\mathbf{Y}$  e  $\mathbf{Z}$ , conforme a estrutura padrão de uma retina bidimensional [adicionar referências]. Entretanto, o espaço de representação possui um componente  $\mathbf{W}$ , representando a quarta dimensão.

Se, para o caso do olho bidimensional em ambiente tridimensional, estamos interessados no ângulo do vetor de luz do pixel, com o plano  $\mathbf{X-Y}$ , para se calcular a cor do pixel e o enfraquecimento do sinal (ângulo  $\beta$ ), para o caso do olho tridimensional em ambiente quadridimensional, estamos interessados no ângulo do vetor de luz do pixel com o espaço  $\mathbf{X-Y-Z}$ , para se calcular o enfraquecimento do sinal.

Dadas essas observações, o cálculo da cor do pixel na coordenada  $\mathbf{y-z}$  é dada pela combinação em *alpha blending* de todos os pontos do reticulado de representação (tridimensional) que possuem os mesmos valores de  $\mathbf{X}$  e  $\mathbf{Y}$ , para todo valor de  $\mathbf{W}$ .

Todos os valores intermediários podem ser deduzidos por generalização do primeiro modelo ampliado.

## 5 Modelo Ampliado (3) - Ambiente N-Dimensional em Tela bidimensional

É possível ampliar a representação em tela bidimensional para ambientes com mais do que quatro dimensões, mas é completamente sem sentido aumentar o número de dimensões da tela, porque o olho humano só pode enxergar representações bidimensionais. Para ambientes com dimensões maiores do que 4, a representação pode ser deduzida como generalização ainda mais extensa do primeiro modelo ampliado.

## 6 Conclusões e Próximos Passos

Esse documento apresenta a teoria necessária para se fazer a representação de ambientes bidimensionais em tela unidimensional, ambientes tridimensionais em telas unidimensionais e ambientes tetradimensionais em telas bidimensionais - o objetivo final do projeto. A partir daí, é possível fazer representações ainda mais altas, mas saímos do objetivo do projeto.

Podemos, ainda, detalhar modelos de múltipla iluminação e iluminação a cores (*tinting*), mas essas técnicas são bem dominadas, não sendo necessário repeti-las aqui [adicionar referências].

A partir da teoria aqui apresentada, pode-se iniciar a implementação de software para fazer representação **3D** em telas **1D** e representação **4D** em telas **2D**. Essas representações provocam, necessariamente, perda de informação do ambiente observado. Essa perda de informação também ocorre na visão humana usual, sendo mitigada pela possibilidade de movimentação dentro do ambiente tridimensional que estamos imersos. Essa mesma movimentação, que é a realização do **Critério da Navegação** permite os mesmo efeito de mitigação da perda de dados da representação 3D em tela 1D e do ambiente 4D em tela 2D. Essa navegação é dada fixando-se um **Referencial de Frenet** no ponto  $\mathbf{p_0}$ , que é o ponto de captura no olho, tanto em 3D quanto em 4D. Fixando esse referencial, é possível navegar dentro dos ambientes tridimensionais e quadridimensionais, tentando recuperar a informação perdida pelas projeções em telas 1D e 2D, respectivamente.