

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

1	Introdução	1
2	Modelo Mínimo - Ambiente Bidimensional em Tela Monodimensional	2
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	5
2.3	Encontrando os pontos renderizados, pixel a pixel da tela	5
3	Modelo Ampliado (1) - Ambiente Tridimensional em Tela Monodimensional	6
4	Modelo Ampliado (2) - Ambiente Quadridimensional em Tela bidimensional	6
5	Modelo Ampliado (3) - Ambiente N-Dimensional em Tela M-dimensional	6
6	Múltiplas fontes de Luz e <i>tinting</i>	6
7	Conclusões e Próximos Passos	6

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 radiação, 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 radiação. 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,

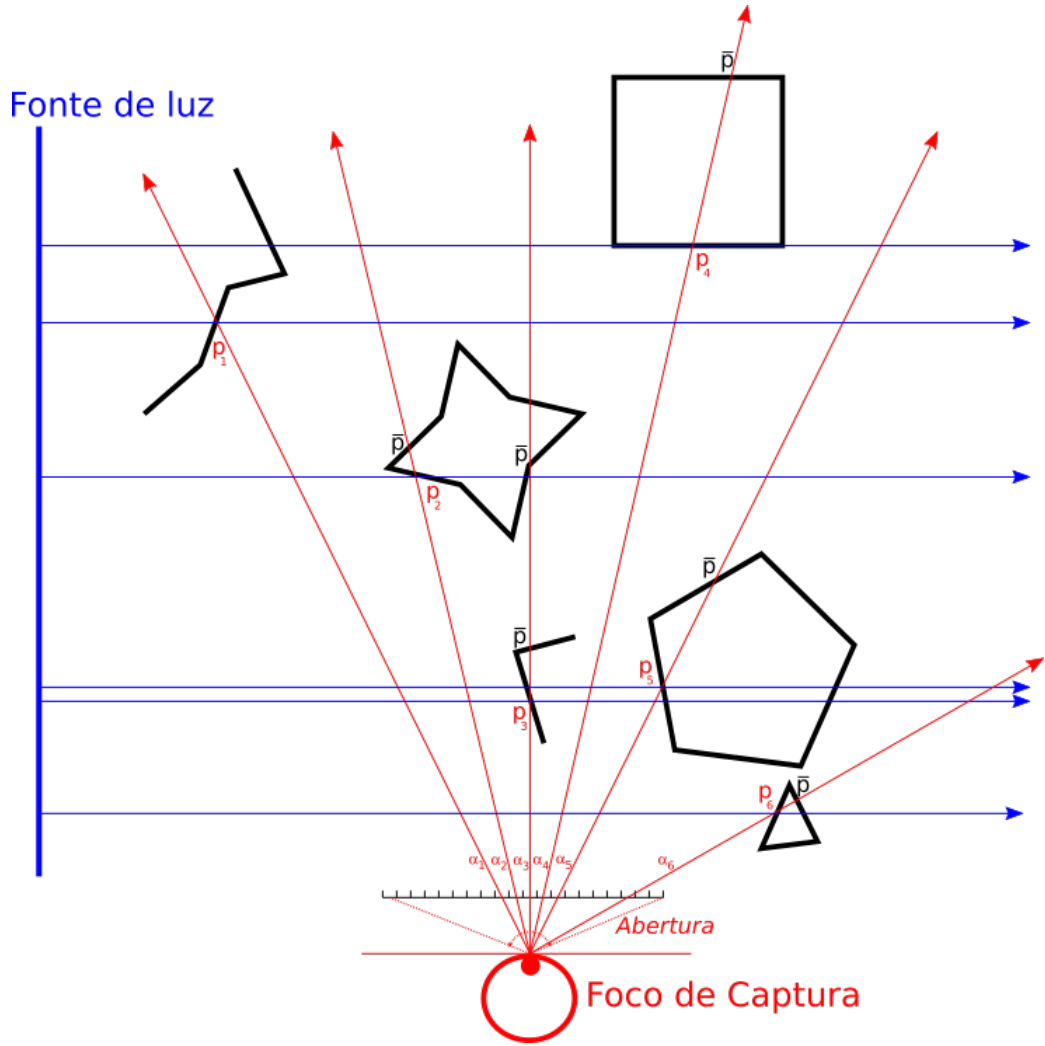


Figura 1: Identificando pontos de renderização. No inferior da figura temos o foco de captura, interceptado por uma tela unidimensional com vinte pixels de tamanho. À esquerda, temos uma fonte de luz paralela. Essa fonte de luz, na figura, está na vertical, mas qualquer direção do plano pode ser utilizada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

estariamos falando de um ângulo representado por uma 3-variedade e a taxa de decaimento seria com o cubo da distância.

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

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

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 3, 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 posição¹. 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.

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 infinitamente² 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.

¹Essa 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.

²Na 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.

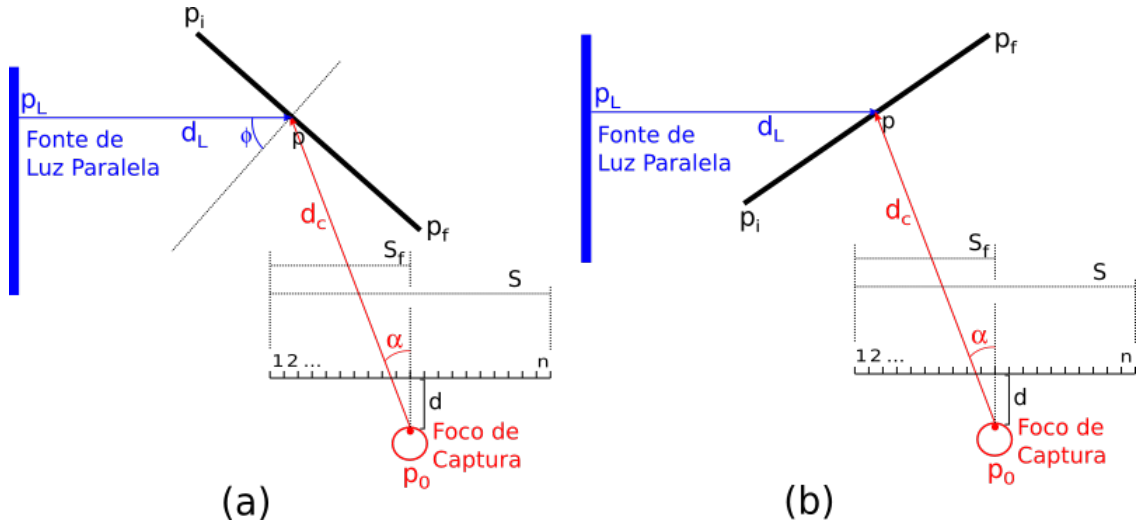


Figura 2: Identificando pontos de renderização.

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_i 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_i 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, 3 e 4.

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 desenhá-la começando pelo pixel de índice $i = 1$, e indo até o pixel de índice $i = n$.

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

alalala alalla

Continua...

2.2 Iluminação Radial

Continua...

2.3 Encontrando os pontos renderizados, pixel a pixel da tela

Continua...

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.
t_l	Largura da tela (unidimensional).
n	Número de Pixels da Tela.
α	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.
ϕ	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.

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

Continua...

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

Continua...

5 Modelo Ampliado (3) - Ambiente N-Dimensional em Tela M-dimensional

Continua...

6 Múltiplas fontes de Luz e *tinting*

Essa seção apresenta a implementação da importante omissão do uso de múltiplas fontes de luz e do *tinting*. O *tinting* é a aplicação de fontes de luz coloridas, que alteram as cores dos objetos iluminados segundo regras específicas diferentes do *alpha-blending* utilizado nas renderizações apresentadas acima. Já o uso de múltiplas fontes de luz provoca um grande aumento de complexidade para a renderização de cada pixel da tela, ainda mais, considerando que cada fonte de luz pode ter uma cor diferente, alterando as regras de *tinting*.

Continua...

7 Conclusões e Próximos Passos

Continua...

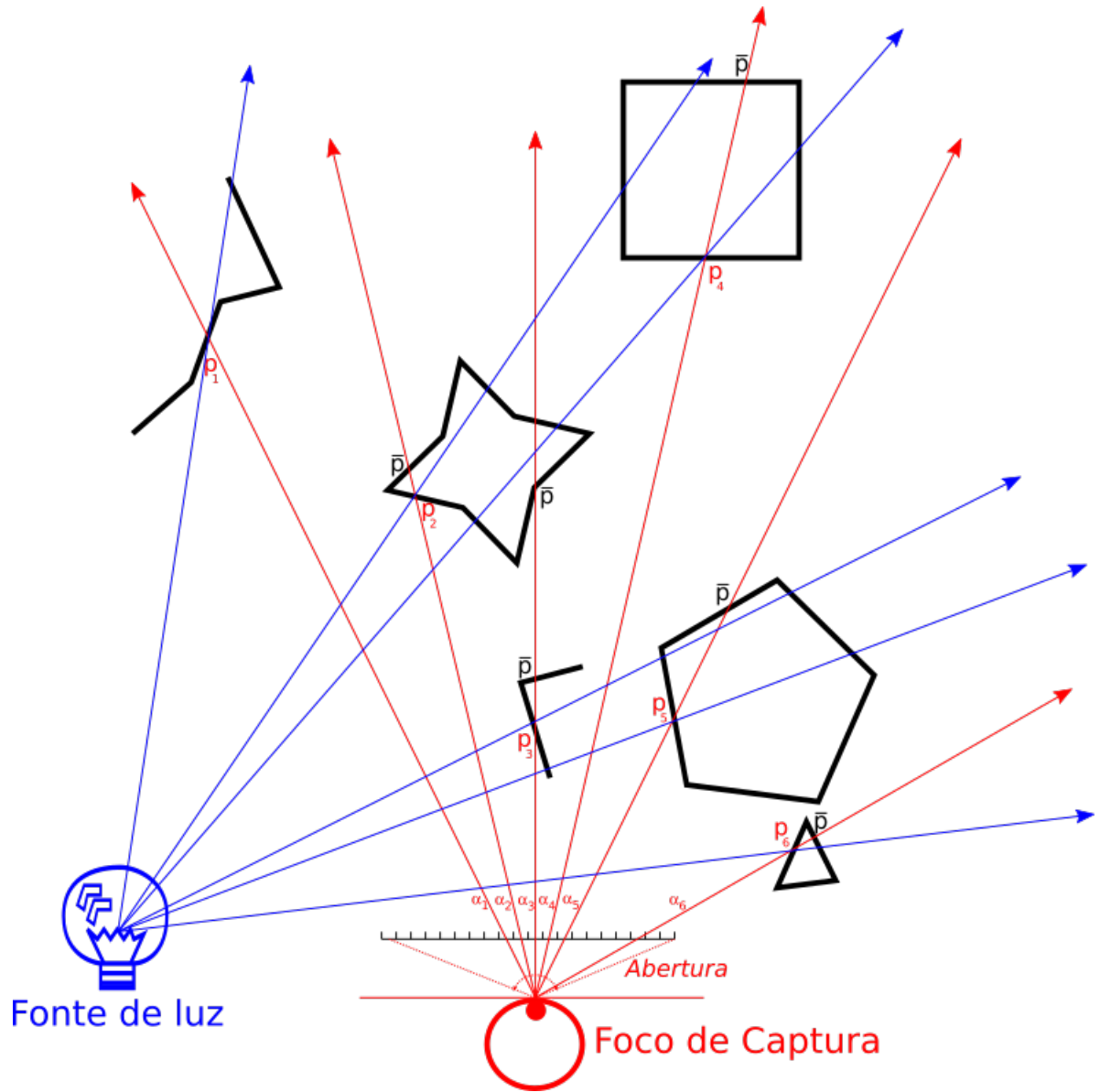


Figura 3: Identificando pontos de renderização.

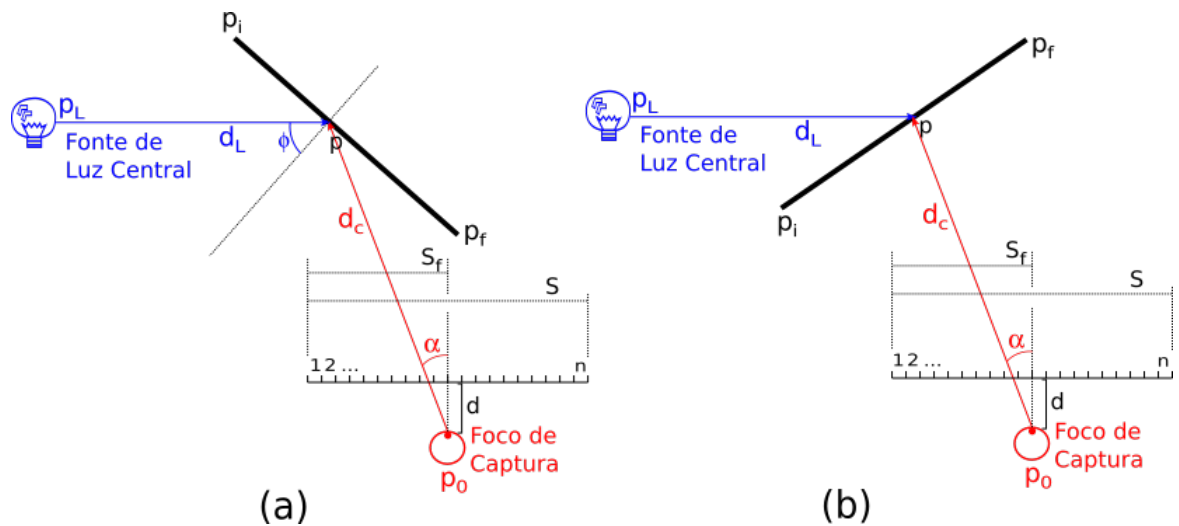


Figura 4: Identificando pontos de renderização.