

4

Navegação Automática e Assistida

O grafo de guia cuja construção foi descrita no capítulo anterior é utilizado em tempo de execução para prover duas funcionalidades: navegação automática e navegação assistida por todo o RNP.

A navegação automática consiste em movimentar a câmera automaticamente a partir de sua localização atual até um ponto de destino informado pelo usuário. Independente da localização atual da câmera o usuário pode, por exemplo, requisitar uma navegação automática até um determinado poço em uma camada específica. A câmera iniciará um movimento automático, navegando pelo reservatório com movimentos suaves e livres de colisão, até chegar ao destino. O sistema necessita apenas da localização (x, y, z) do ponto de destino desejado e da camada a qual ele pertence, oferecendo assim flexibilidade para outras opções de interface para escolhas de destino.

Na navegação assistida, o usuário pode navegar da forma usual utilizando controles simples de movimentação via teclado, alterando a direção de visualização com o *mouse*. O sistema auxilia o usuário impedindo colisões com a superfície do RNP e seus poços, suavizando o movimento.

O sistema realiza, em ambos os modos de navegação, consultas ao grafo de guia do RNP em busca de caminhos livres de colisão. Conforme visto na seção 2.3, qualquer algoritmo de busca em grafos pode ser utilizado. Este trabalho utiliza o algoritmo A^* [19], definindo uma heurística capaz de encontrar rapidamente um caminho mesmo entre grandes distâncias.

As seções a seguir descrevem, respectivamente, as funcionalidades de navegação automática e assistida. Finalmente, é descrita a heurística utilizada para a busca no grafo de guia.

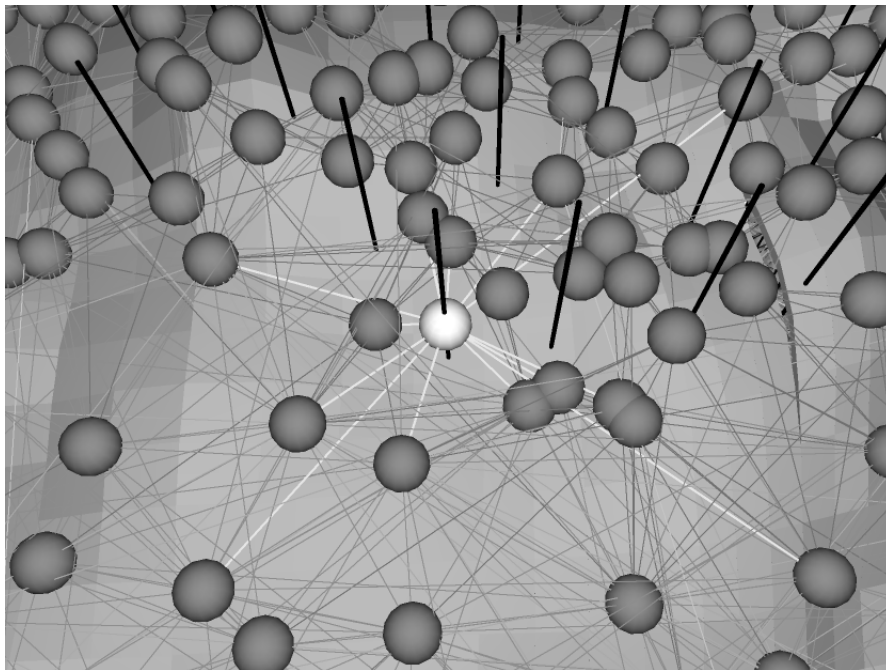


Figura 4.1: O nó de destino é inserido no lugar do poço a ser visitado e conectado como um nó regular do grafo (arestas criadas em destaque).

4.1 Navegação Automática

A navegação automática consiste em mover a câmera a partir de sua posição atual até um destino qualquer, sem intervenção do usuário, com movimentos suaves e livres de colisão com o RNP. Neste trabalho foi implementada uma interface onde o usuário informa um poço do reservatório e uma camada. Em seguida, a câmera se movimenta automaticamente, a partir de sua posição atual, em direção ao poço na camada desejada.

O sistema mantém controle sobre qual *aresta* do grafo de guia a câmera reside. O estado de origem é a posição atual da câmera e é um estado válido por necessariamente residir em uma aresta do grafo de guia. Este estado é promovido temporariamente a nó e então tenta se conectar aos nós da aresta da câmera e também aos vizinhos destes nós. O nó de destino (também temporário) é inferido computando-se a interseção do poço com a camada de destino desejada e recuando o ponto de inserção de acordo com a altura de navegação (seção 3.2). O poço a ser visitado é temporariamente removido do RNP (para fins de teste de colisão) durante a etapa de inserção do nó e sua conexão. O nó de destino conecta-se com os nós da camada de destino da mesma forma que um nó regular de grafo de guia de camada (seção 3.2.1). A Figura 4.1 ilustra um nó de destino inserido para um determinado poço.

Após a conexão dos nós de origem e destino, o sistema dispara uma busca utilizando o algoritmo A^* [19]. Definido um caminho entre os dois nós, a sequência correspondente de arestas é retornada. Em seguida, o caminho é encurtado utilizando um algoritmo incremental, ilustrado na Figura 4.2. A partir do nó de origem n_1 , o sistema tenta encurtar o caminho tentando conectá-lo aos nós subsequentes a seu vizinho n_2 . No exemplo da figura, a primeira tentativa mal sucedida ocorre com n_4 , portanto o sistema cria uma aresta entre n_1 e n_3 , eliminando as arestas $[n_1 - n_2]$ e $[n_2 - n_3]$. O processo continua agora a partir de n_3 e, no caso, é mal sucedido ao tentar conectá-lo a n_6 , portanto criando uma aresta de atalho apenas até n_5 . Ao processar n_5 , o sistema consegue uni-lo a n_7 . Após o encurtamento, o sistema aplica o processo de suavização e controle de velocidade propostos por Nieuwenhuisen e Overmars [23] (seção 2.3.1) no caminho resultante. A Figura 4.3 ilustra um caminho final calculado a partir de um ponto de origem A até um poço localizado no ponto B . Antes de iniciar o movimento da câmera, ela é girada até a direção do poço desejado na camada atual, alinhando seu vetor up com o eixo z negativo. Durante o seu deslocamento, o seu vetor up mantém-se alinhado e a direção de visualização é constantemente atualizada na direção do poço de destino. A câmera se movimenta até uma distância pré-estabelecida do poço desejado, mantendo-se assim em um estado final válido (Figura 4.4).

4.2

Navegação Assistida

O modo assistido permite que o usuário navegue em tempo real pelo RNP utilizando, via teclado, seis controles simples de movimentação: quatro para navegação na camada atual (frente, trás, esquerda e direita) e dois para mudança explícita de camada (acima e abaixo). A direção de visualização é totalmente controlada pelo usuário, sem restrições, através do *mouse*. Formas similares de controle de navegação são comuns em jogos e em outras aplicações de realidade virtual. O sistema auxilia o usuário evitando colisões com o RNP e produzindo movimentos suaves.

O grafo de guia do RNP é consultado constantemente, enquanto há entrada do usuário, em busca de caminhos livres de colisão para a câmera seguir. A câmera sempre se movimenta sobre um caminho do grafo de guia. De acordo com a entrada do usuário, o sistema infere um estado de origem e um de destino (ambos válidos), através de uma heurística proposta, utilizando-os para calcular um caminho suave e livre de colisões. As técnicas

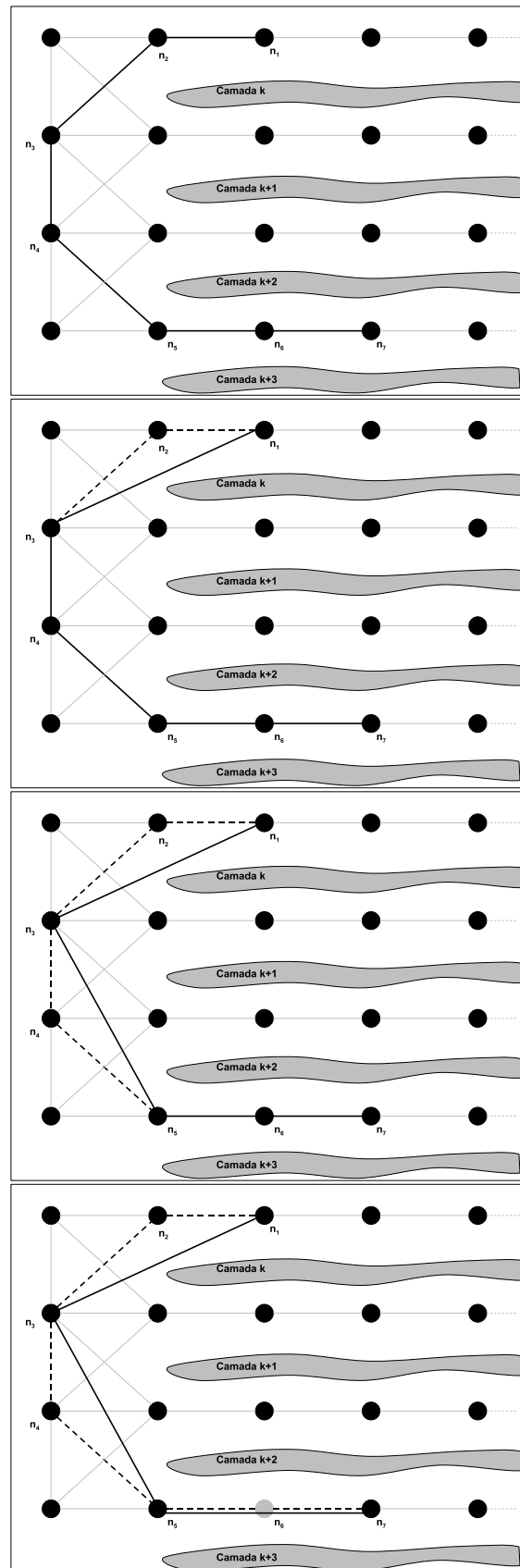


Figura 4.2: O funcionamento do algoritmo de encurtamento em um caminho com extremos em n_1 e n_7 .

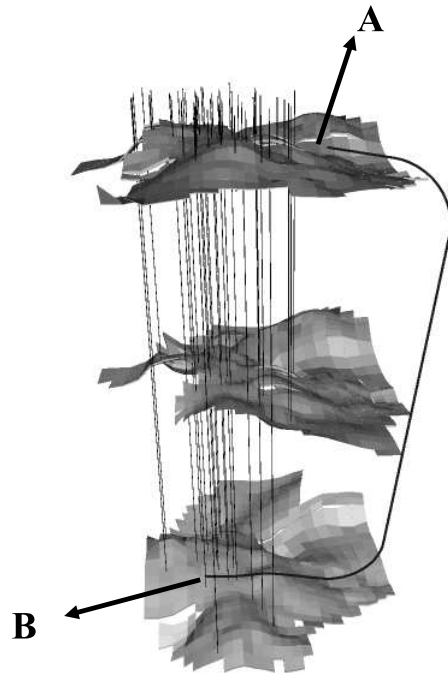


Figura 4.3: O caminho final a ser percorrido pela câmera localizada em A até o destino B , após encurtamento e suavização.

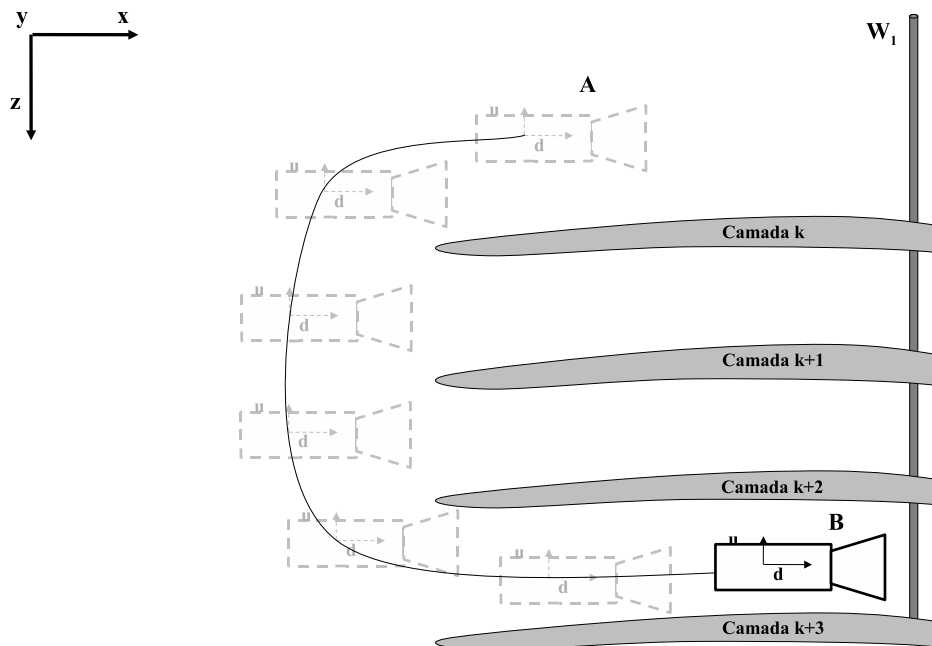


Figura 4.4: A câmera se desloca de seu ponto original (A) até o ponto de destino (B) sempre visualizando o poço desejado (W_1), ao mesmo tempo em que mantém seu vetor up alinhado com o eixo z negativo.

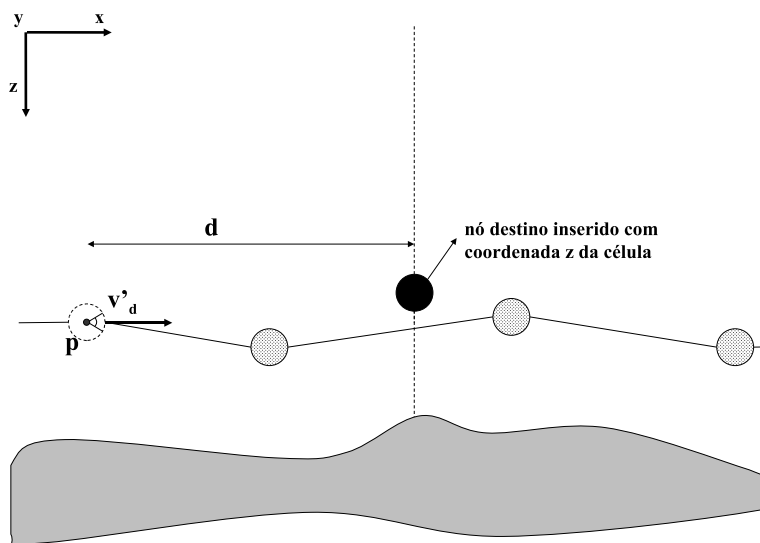


Figura 4.5: A obtenção do nó temporário de destino quando o usuário requisita um movimento para frente.

de conexão dos nós, de suavização do caminho e de controle de velocidade são idênticas às utilizadas no modo automático. Ao longo da navegação, se o usuário modificar a orientação da câmera ou informar um comando de navegação diferente, o sistema descarta o caminho atual (juntamente com os nós temporários inseridos) e repete o processo de acordo com o estado da câmera e comando de navegação atuais, encontrando um novo caminho a seguir. Se o usuário mantiver o comando de navegação e a orientação da câmera, o sistema descarta o caminho atual após a câmera ter percorrido um trecho suficiente (nos experimentos foi utilizado o valor de 70% do caminho), recalculando um novo caminho a partir do estado atual. Se a entrada do usuário cessar ou não houver um caminho possível (câmera voltada para o vazio, navegação em direção a um local cercado de poços, descer estando na última camada ou subir estando na primeira), a câmera pára. Em momento algum o sistema interfere na orientação da câmera.

O estado de origem é sempre inferido a partir do estado atual da câmera, de forma idêntica ao modo automático. De acordo com o comando de navegação informado, a posição da câmera e sua orientação, o sistema tenta inferir um estado de destino.

Ao receber um comando de navegação em camada (frente, trás, esquerda ou direita), o sistema projeta a direção de visualização atual na grade de amostragem da camada onde a câmera está e, de acordo com o comando, computa um novo vetor \mathbf{v}_d' a ser utilizado para prever o destino do movimento. Por exemplo, se o usuário requisitar um movimento para frente, o vetor utilizado é a própria projeção da direção de visualização.

Dada a posição \mathbf{p} da câmera, o sistema computa uma célula alvo determinando em qual célula da grade reside o ponto $\mathbf{p} + d\mathbf{v}'_d$, onde d é um fator de extrapolação pré-estabelecido (nos experimentos foi utilizado um quarto da diagonal da grade). A Figura 4.5 ilustra a obtenção do nó temporário de destino quando o usuário deseja ir para a frente. Se a célula contiver um nó do grafo, este nó é utilizado como objetivo. Caso contrário, o estado de destino é inferido localizando-o no centro da célula, utilizando a coordenada z armazenada (Seção 3.2). Se o estado for válido (ou seja, sua esfera correspondente não colidir com o reservatório), o nó é criado e conectado como um nó regular do grafo (Seção 3.2.1).

Entretanto, nem sempre a primeira tentativa é bem sucedida. A célula escolhida pode ser *inativa* ou o estado encontrado pode ser inválido (sua esfera correspondente colide com o reservatório). Nesse caso, o sistema efetua uma rasterização a partir desta célula inicial até próximo da célula onde reside a câmera, em linha reta, buscando uma célula *ativa* ou *de borda*. A câmera 1 ilustrada na Figura 4.6 se enquadra nesta situação: a célula inicial era *inativa*, portanto foi necessária a rasterização em direção à célula de origem, obtendo uma amostra mais próxima na borda. Se mesmo após este processo não for possível obter uma amostra, o sistema realiza uma nova rasterização a partir da célula inicial, também em linha reta, na direção oposta até conseguir obter um nó de destino ou atingir os limites da grade. Ao processar a câmera 2 da Figura 4.6, por exemplo, não foi possível obter uma amostra na rasterização inicial, pois todas as células visitadas no processo eram *inativas*. Logo, o sistema rasterizou da célula inicial em diante e obteve uma amostra no agrupamento de células a frente. Se o sistema atingiu os limites da grade sem conseguir obter uma amostra, a câmera permanece parada (câmera 3 da Figura 4.6). O objetivo é a análise das células hexaédricas do reservatório, portanto uma navegação em direção ao vazio não é considerada relevante.

Se a câmera estiver em uma aresta que conecta nós de diferentes camadas, os nós de destino correspondentes das camadas intermediárias e dos extremos são obtidos e o nó escolhido será o que possuir menor $|\Delta_z|$ em relação ao nó de origem, conforme ilustrado na Figura 4.7.

O sistema também pode receber um comando explícito de mudança de camada. Neste caso, busca-se um nó de destino na camada mais próxima para cima ou para baixo, dependendo do comando fornecido. O sistema determina em qual célula da grade de amostragem da camada de destino a câmera reside e a partir dela, realiza uma rasterização ao longo da direção de visualização projetada na grade, até encontrar uma célula *ativa*, *de borda* ou

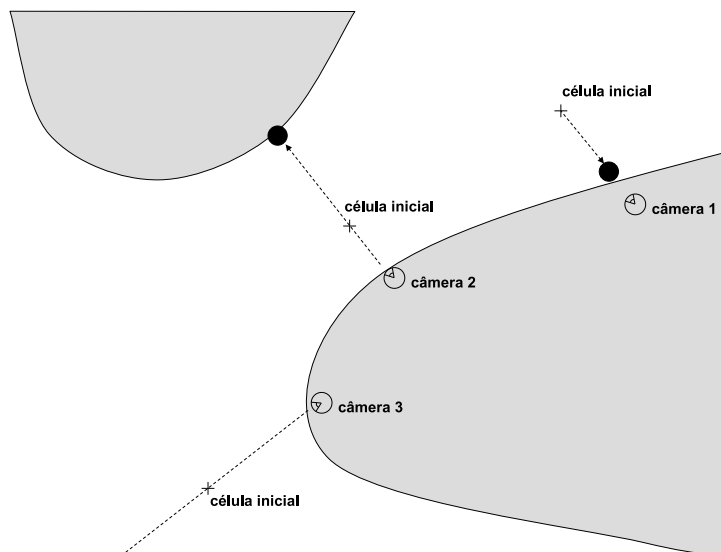


Figura 4.6: A busca por um nó de destino temporário no modo de navegação assistida.

atingir os limites da grade. Se for encontrada uma célula *ativa* ou *de borda*, seu nó correspondente é utilizado como nó de destino ou, na sua ausência, um nó é criado no centro da célula, com a coordenada z armazenada. Se os limites da grade foram atingidos, tenta-se a próxima camada acima ou abaixo, de acordo com o comando fornecido. Se a câmera estiver entre camadas, somente as camadas do intervalo são visitadas. Se o sistema não conseguiu obter um nó após visitar todas as camadas correspondentes, a câmera permanece parada. A Figura 4.8 ilustra o processo quando o usuário requisita uma navegação para baixo. Ao processar primeiramente a camada $k + 1$, um nó foi encontrado e o caminho é computado.

4.3

Heurística para o algoritmo de busca no grafo de guia

O algoritmo A^* utilizado neste trabalho é considerado um algoritmo de busca *informada* [19]. Algoritmos deste tipo obtém uma informação adicional a respeito do ambiente com o intuito de chegar ao destino mais rapidamente. No caso do algoritmo A^* , esta informação é a estimativa do custo do nó sendo processado até o destino. O custo total $f(n)$ de um nó n do grafo é dado por:

$$f(n) = g(n) + h(n)$$

Onde $g(n)$ é o custo real da origem até n e $h(n)$ uma estimativa do

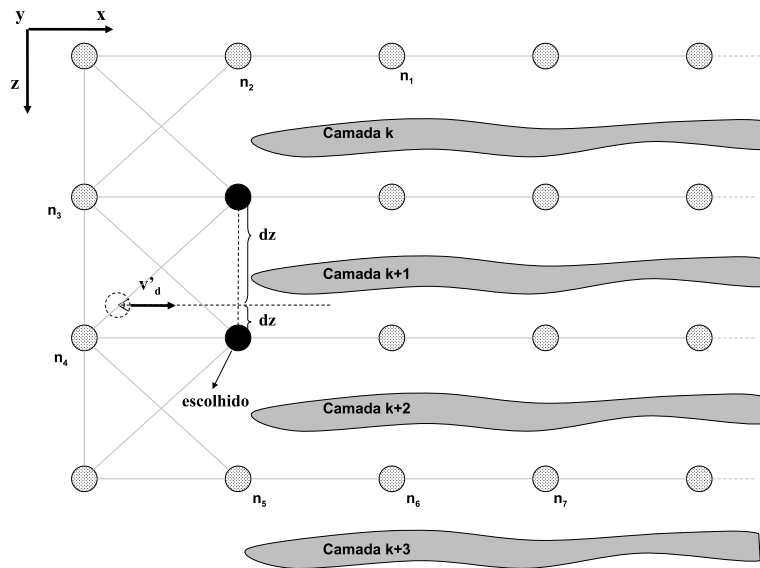


Figura 4.7: Se a câmera estiver entre camadas, o nó de destino escolhido é o de menor $|\Delta_z|$.

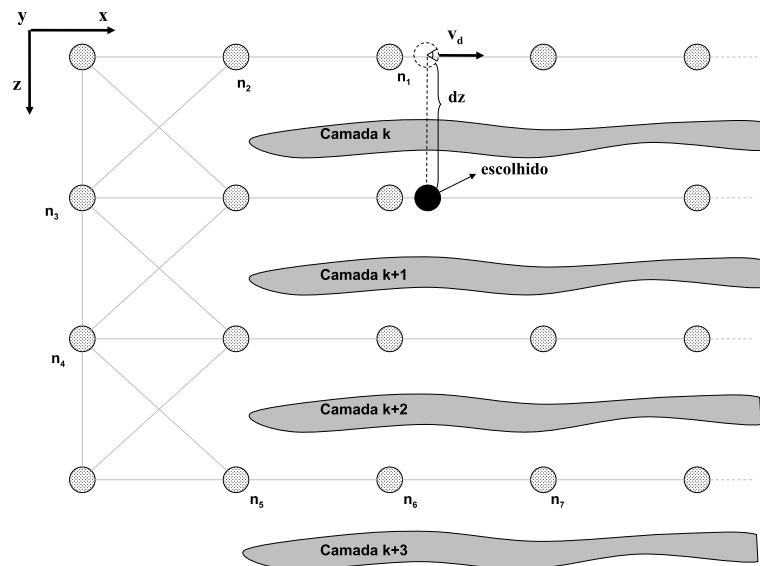


Figura 4.8: Ao receber um comando de navegação para baixo, o sistema processou primeiramente a camada $k + 1$ e conseguiu encontrar um nó de destino.

custo de n até o destino. O algoritmo visita, a cada iteração, o nó mais promissor, ou seja, o que possui menor valor de $f(n)$.

Se a função $h(n)$ for uma *subestimativa* do custo real de n até o destino, garantidamente o algoritmo encontra o caminho ótimo, caso exista [19]. Uma função $h(n)$ deste tipo é dita *admissível*. Por outro lado, se $h(n)$ for uma *sobrestimativa*, embora não haja garantias de encontrar o caminho ótimo, o algoritmo tende a prosseguir com mais avidez em direção ao destino, visto que o valor de $f(n)$ será bem menor em ramos mais próximos do objetivo. Isso faz com que o algoritmo encontre uma solução mais rapidamente na maioria dos casos.

No modo de navegação assistida, mais especificamente no tratamento de comandos para navegação na mesma camada (frente, trás, esquerda, direita), é importante que o caminho encontrado seja ótimo, pois neste caso, deve-se poder navegar ao longo de áreas com alta densidade de poços, para que o usuário possa analisá-los. A utilização de uma função $h(n)$ sobrestimada causaria desvios indesejáveis, impossibilitando a navegação nessas áreas mais problemáticas. Na navegação automática, também quando o nó de destino encontra-se na camada atual, é melhor obter um caminho ótimo, pois levando-se em conta que o poço está sempre à vista, é desconfortável a percepção que a câmera está tomando um caminho mais longo para alcançá-lo.

Por outro lado, tanto na navegação assistida quanto automática, quando o nó de destino não se encontra na camada atual, o fato de sair da camada e prosseguir pelas camadas intermediárias através de um caminho um pouco mais longo já é tolerável. O objetivo preliminar passa a ser encontrar um caminho até a camada de destino rapidamente. Uma vez na camada destino, acha-se o caminho ótimo até o poço alvo. Essa característica é importante na medida em que permite que a busca seja mais rápida nesta situação difícil: a camada de origem e as camadas intermediárias representam obstáculos problemáticos pois oferecem uma enorme quantidade de alternativas para serem transpostas (representadas pelo seu grafo de guia correspondente). O algoritmo não pode ponderar por muito tempo qual alternativa utilizar para transpor a camada. Para acelerar o processo, foi concebida uma função $h(n)$ que guia o algoritmo para sair da camada rapidamente e, também importante, para nunca mais retornar a camadas anteriores para cogitar outras alternativas.

Dado um nó n a ser avaliado pelo algoritmo, se n estiver na mesma camada do nó de destino, o valor de $h(n)$ é dado pela sua distância euclidiana para o objetivo. Este comportamento clássico de $h(n)$ para nós na mesma

camada faz com que a câmera navegue, após alcançar a camada de destino, pelo trajeto ótimo a partir do ponto de chegada até o destino. Por outro lado, se o nó n e o nó de destino estiverem em camadas distintas, $h(n)$ é dado por:

$$h(n) = 2d + l_p |\Delta_k|$$

O fator d é a distância euclidiana para um *nó de borda* próximo de n (armazenado no próprio nó em pré-processamento, conforme descrito na Seção 3.3). Se n for um nó de borda, o valor de d é zero. Este fator d é responsável por guiar o algoritmo para fora da camada o mais rápido possível. Com o intuito de reforçar ainda mais este efeito, o fator é sobrestimado, sendo multiplicado por 2. O fator $l_p \times |\Delta_k|$ tem a função de penalizar nós de acordo com a distância, em número de camadas, de n até o nó de destino. O valor de l_p é fixo e escolhido de forma a fazer com que nenhum nó das camadas anteriores seja mais promissor que os da camada atual, impedindo que o algoritmo retorne às camadas anteriores em busca de outras alternativas. O valor de l_p pode ser ajustado para um valor exagerado porém garantido, como o comprimento da diagonal da caixa envolvente de todo o RNP. O fator Δ_k representa a distância, em número de camadas, de n até o nó de destino. Durante a busca em uma dada camada, o fator $l_p \times |\Delta_k|$ é o mesmo para todos os nós da camada, fazendo com que d seja responsável por guiar o algoritmo para o nó de borda (ponto de saída) mais próximo.

Esta heurística aumentou de forma significativa o desempenho do algoritmo ao processar consultas entre nós de camadas distintas, bastante comuns no modo de navegação automática, atingindo taxas de tempo real (Capítulo 5).