APROXIMAÇÃO PLANAR POR PARTES PARA RECONSTRUÇÃO 3D DENSA

Leonardo de Assis Silva*, Junia Gabriela Lube Pimentel*, Raquel Frizera Vassallo*

*Departamento de Engenharia Elétrica, Universidade Federal do Espírito Santo
Vitória, Espírito Santo, Brasil

Emails: leonardo.as.ufes@gmail.com, pimentel.junia@gmail.com, raquel@ele.ufes.br

Abstract— In this paper we propose a method to estimate prior knowledge of a scene before a 3D dense reconstruction is performed, aproximating homogeneous regions by planes. This is because usually it is not possible to estimate with precision the depth of homogeneous regions, due to the lack of good matches between pixels. The proposed approach was compared to the DTAM method, considering the recall (the amount of reconstructed pixels) and the absolute error in relation to the ground truth, by using synthetic images. In our best result, even before performing any 3D dense reconstruction, we achieved a recall of 92.3% against 100% from DTAM, but with an average error seven times smaller.

Keywords— 3D Reconstruction, Homogeneous regions.

Resumo— Neste artigo é proposta uma metodologia de se estimar conhecimentos prévios da cena antes de se realizar a reconstrução 3D densa propriamente dita, aproximando-se regiões homogêneas por planos. Isso porque, normalmente não é possível se estimar com precisão a profundidade de regiões homogêneas de uma imagem, devido à dificuldade de se encontrar boas correspondências entre *pixels*. A abordagem proposta foi comparada com o método DTAM, em termos de *recall* (porcentagem de *pixels* reconstruídos) e erro absoluto em relação ao *ground truth*, utilizando-se imagens sintéticas. Como melhor resultado, mesmo antes de se realizar a etapa final de reconstrução densa, o método apresentou um *recall* de 92, 3% contra 100% do DTAM, mas com erro médio sete vezes menor.

Palavras-chave— Reconstrução 3D, Regiões homogêneas.

1 Introdução

Câmeras digitais estão entre os sensores mais utilizados, principalmente devido à riqueza de informações contidas em uma imagem e ao avanço tecnológico, que permitiu o seu barateamento. Assim, câmeras se tornaram mais comuns em sistemas embarcados, celulares, computadores, etc, podendo ser usadas, em muitos casos, para substituir sensores laser, que normalmente são caros.

Além disso, com o aumento da capacidade de processamento dos dispositivos, imagens podem ser usadas para analisar cenas e extrair informações tridimensionais dos *pixels* em um espaço de tempo bem menor do que se conseguia fazer há alguns anos atrás. Esse processo de recuperação da informação tridimensional do meio recebe o nome de reconstrução 3D.

Mesmo assim, obter uma boa reconstrução 3D do ambiente ainda é um processo computacionalmente custoso. Isso é um problema que a área de sistemas embarcados enfrenta, já que, geralmente, o tempo de processamento das informações deve ser baixo, para que os sistemas consigam responder rapidamente. Por isso, muitos trabalhos tentam equilibrar os ganhos em velocidade de processamento e qualidade da reconstrução 3D.

Newcombe et al. (2011) propõe um sistema chamado DTAM que, em tempo real, faz a aquisição de imagens com uma câmera em movimento, estima a sua posição corrente e realiza a reconstrução 3D da cena filmada. Por outro lado, Wu et al. (2010) realiza a reconstrução de imagens salvas em seu repositório, com as posições já estimadas (câmera calibrada), por meio de uma técnica com alto custo computacional chamada *Tensor-based Multiview Stereo* (TMVS).

A diferença entre essas duas técnicas está no equilíbrio entre velocidade e qualidade. Enquanto Newcombe et al. (2011) deu mais importância ao tempo de processamento de seu sistema, Wu et al. (2010) escolheu ter uma maior precisão na reconstrução 3D em seus testes. A mesma escolha por qualidade pode ser vista em (Pagani et al., 2011), que utiliza uma técnica chamada *Patch-based Multiview Stereo* (PMVS) em imagens esféricas para a estimativa da profundidade de ambientes.

Neste trabalho também será dada prioridade à velocidade de reconstrução. Desse modo, o objetivo é que posteriormente, o algoritmo implementado seja aprimorado e embarcado em um robô.

O processo de reconstrução do DTAM pode ser dividido em duas etapas: estimativa densa discreta da profundidade dos *pixels* (estimativa grosseira) e estimativa densa contínua por meio de um processo iterativo de minimização (mais precisa). Para isso é necessário encontrar uma boa correspondência entre os *pixels* das diferentes imagens para estimar corretamente a profundidade discreta. Isso fica facilitado quando as imagens possuem regiões com muita textura.

Para regiões homogêneas a ambiguidade é elevada, dificultando a correspondência entre *pixels* cuja vizinhança seja semelhante. Consequentemente, mesmo após o processo completo de reconstrução, a precisão do DTAM acaba sendo muito prejudicada por causa da profundidade estimada das áreas sem textura.

Esse problema é discutido em (Concha and Civera, 2015) e (Concha et al., 2015), que apresentam formas de se aumentar a precisão do DTAM em regiões homogêneas, utilizando técnicas que não prejudiquem muito a velocidade do processo de reconstrução $3D$. Essas técnicas adicionam informações de entendimento da cena à etapa de estimativa densa do DTAM como, por exemplo, aproximação planar de áreas homogêneas, e classificação de regiões das imagens como sendo parte do teto, das paredes, do chão dentre outras.

Neste trabalho é proposta uma outra metodologia de se estimar conhecimentos de entendimento da cena antes de se realizar a reconstrução $3D$ densa propriamente dita. Vale ressaltar que tal reconstrução densa não é objeto direto de estudo deste trabalho, o qual está focado na obtenção de um mapa de profundidades mais rico e preciso que favoreça a reconstrução densa final.

Na metodologia proposta, segmentam-se regiões homogêneas assumindo-se planaridade. Inicialmente são reconstruídos apenas os *pixels* que delimitam essas regiões e *pixels* de alto gradiente, para, depois, estimar os planos que as compõem. Os casos em que as regiões não podem ser estimadas com planos são tratados separadamente.

O trabalho aqui proposto será apresentado em cinco seções. A Seção 2 traz a representação do modelo *pin-hole* de câmera e alguns termos adotados. A implementação do algoritmo proposto está explicado detalhadamente na Seção 3. Os resultados são apresentados e discutidos na Seção 4. A conclusão, propondo alguns trabalhos futuros, é abordada na Seção 5.

2 Modelo de câmera

Para desenvolver e verificar o bom funcionamento do método proposto, o trabalho apresentado neste artigo utiliza imagens sintéticas fotorealistas geradas por um *software* chamado POVRay, disponibilizadas por (Handa et al., 2012). Esse banco de dados também fornece a matriz de parâmetros intrínsecos K, e extrínsecos da câmera T_{wc} (Equação 1).

$$T_{wc} = \begin{pmatrix} R_{wc} & t_{wc} \\ 0^T & 1 \end{pmatrix} \quad K = \begin{pmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{1}$$

A matriz T_{wc} descreve a transformação de um ponto do referencial da câmera c para o mundo w. R_{wc} é a matriz de rotação e t_{wc} a de translação. f é a distância focal da câmera expressa em *pixels*, e c_x e c_y são as coordenadas x e y do ponto principal da imagem.

A representação de um *pixel* no sistema de coordenadas da câmera c é $u_c = (u, v)^T$, sendo $\dot{u}_c = (u, v, 1)^T$ em coordenadas homogêneas. Assim, a relação entre um *pixel* no referencial de c e seu correspondente em m é dada pela Equação 2, na qual z_m e z_c são os valores de profundidade do ponto nas coordenadas das câmeras m e c, respectivamente.

$$z_m \dot{u}_m = K(R_{mc} K^{-1} z_c \dot{u}_c + t_{mc}), \tag{2}$$

No processo de reconstrução deste trabalho, são utilizadas M imagens I_m que possuem certa sobreposição com uma imagem de referência I_r, apresentada na Figura 1. Todas as imagens utilizadas pertencem ao espaço de cor RGB.

Figura 1: Imagem de referência I_r.

3 Metodologia

3.1 Visão Geral

A Figura 2 apresenta resumidamente o esquemático do algoritmo, dividido em 4 etapas.

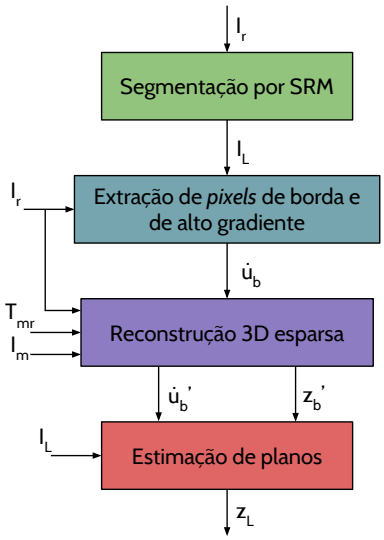


Figura 2: Esquemático resumido do algoritmo de reconstrução.

Na primeira etapa (Seção 3.2), a imagem I_r é segmentada em regiões homogêneas de cor similar utilizando *Statistical Region Merging* (SRM), que retorna uma matriz I_L contendo os rótulos (*labels*) de cada região. I_L é uma matriz $2D$ de mesmo tamanho que I_r. Esse processo de segmentação servirá para a reconstrução de áreas homogêneas que ocorre na última etapa do algoritmo.

Considera-se neste processo que cada região de I_L corresponde a uma área plana. Portanto, para descobrir a profundidade dos *pixels* contidos

em cada região, só é preciso descobrir a profundidade dos *pixels* de suas bordas para estimar o vetor normal ao plano correspondente ($\vec{n}$) e distância do plano à câmera (d). Dessa forma, na segunda etapa (Seção 3.3), os *pixels* da borda de cada área segmentada são extraídos ($\dot{u}_b$).

A terceira etapa (Seção 3.4) realiza a reconstrução esparsa, estimando a profundidade dos *pixels* escolhidos na segunda etapa ($\dot{u}_b$). Essa etapa também retira do conjunto *pixels* com possíveis erros de correspondência. A estimativa de profundidade z'_b é associada aos *pixels* $\dot{u}'_b$ que não foram descartados. Essa estimativa será utilizada na última etapa (Seção 3.5) para a aproximação de cada plano, usando o algoritmo do RANSAC para retirar possíveis *outliers*.

3.2 Segmentação por SRM

Devido à falta de textura em regiões homogêneas, a reconstrução $3D$ dessas áreas normalmente acumula muito erro, pois não é possível encontrar com precisão a correspondência dos *pixels* dessas regiões em outras imagens.

Trabalhos como (Fraundorfer et al., 2006), (Concha and Civera, 2015), (Concha et al., 2015) e (Concha et al., 2014) tentam não apenas estimar a correspondência de *pixels* entre imagens, mas também estimar a correspondência de regiões para aumentar a precisão de seus resultados. Como estas regiões são homogêneas, fica difícil saber exatamente o seu formato tridimensional. Por isso, estes trabalhos aproximam essas áreas por regiões planares e depois as utilizam como uma hipótese inicial em um processo iterativo de minimização.

Nesse contexto, torna-se importante segmentar a imagem em regiões homogêneas, consideradas possivelmente planas. Li et al. (2015) mostra uma comparação entre os métodos SLIC de (Achanta et al., 2010), SRM e o proposto por eles, que junta características dos dois métodos citados. Além desses métodos, outro algoritmo que segmenta regiões homogêneas é o MSER (*Maximally Stable Extremal Regions*).

O SRM foi a técnica escolhida para ser utilizada no processo de segmentação deste trabalho. Ademais, antes da segmentação, o método Tsmooth de (Xu et al., 2012) foi utilizado na imagem RGB I_r para filtrar ruídos e ainda manter bordas bem definidas. O algoritmo do SRM recebe a imagem filtrada e retorna a matriz I_L contendo os *labels* de cada região. A Figura 3 apresenta as regiões segmentadas pelo método de SRM (I_L).

3.3 Extração de borda

Com a matriz I_L fornecida pela etapa de segmentação, o próximo passo é a extração dos *pixels* das bordas internas de cada região homogênea ($\dot{u}_b$, no referencial de I_r). Para isso, foram utilizadas técnicas de morfologia matemática.

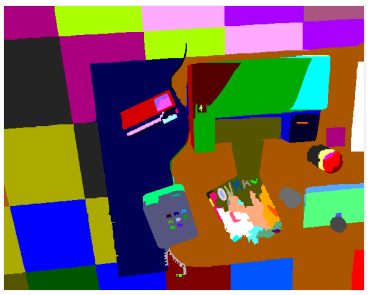


Figura 3: Matriz I_L com as regiões segmentadas.

Como foi dito na Seção 3.1, esses *pixels* terão a profundidade estimada na etapa de reconstrução esparsa. Entretanto, adiciona-se também ao conjunto de $\dot{u}_b$ outros *pixels*, do interior de cada região, com fortes características em relação à sua vizinhança, como os de alto gradiente. Se existem *pixels* com melhor chance de serem bem reconstruídos no interior de cada região, sua utilização pode ajudar na estimação correta dos planos.

Para saber qual método de extração de *pixels* de alto gradiente escolher, foram comparadas algumas técnicas: *pixels* de alto gradiente, com e sem seus vizinhos, e *pixels* de borda encontrados pelo método de Canny, antes e depois da imagem I_r em escala de cinza ter passado por um filtro de média 3×3. Além disso, também foram testadas a utilização de todos os *pixels* da imagem no conjunto de $\dot{u}_b$, e o caso de nenhum *pixel* de alto gradiente do interior das regiões ser adicionado a $\dot{u}_b$ ("Apenas bordas", Tabela 1).

3.4 Reconstrução 3D esparsa

Na etapa de reconstrução esparsa, cada par de imagens I_r e I_m é retificado, resultando em I_{rR} e I_{mR}. Os *pixels* $\dot{u}_b$ também são transformados para o referencial de I_{rR} por meio da transformação T_1, obtida nesse processo de retificação (Equação 3).

$$z_{rR}\dot{u}_{rR} = K(RK^{-1}z_b\dot{u}_b), \qquad T_1 = KRK^{-1} \tag{3}$$

Como o processo de retificação apenas rotaciona os referenciais, a distância entre os centros óticos das duas câmeras se mantém constante. Essa distância, chamada de *baseline*, será representada por b. Assim, os referenciais de I_{rR} e I_{mR} ficam com mesma orientação e deslocados de b na direção x (Figura 4).

$$z_{mR}\begin{pmatrix} u_{mR} \\ v_{mR} \\ 1 \end{pmatrix} = z_{rR}\begin{pmatrix} u_{rR} \\ v_{rR} \\ 1 \end{pmatrix} + K\begin{pmatrix} b \\ 0 \\ 0 \end{pmatrix},$$

$$z_{mR} = z_{rR} = z_R, \quad u_{mR} = u_{rR} + \frac{bf}{z_R} \tag{4}$$

A Figura 4 também apresenta as linhas epipolares do ponto P, projetado nas duas imagens. Pode-se notar que, após a retificação, as linhas epipolares ficam paralelas ao eixo horizontal. Isso acontece por causa da relação entre os *pixels* de I_{rR} e os de I_{mR}, descrita pela Equação 4.

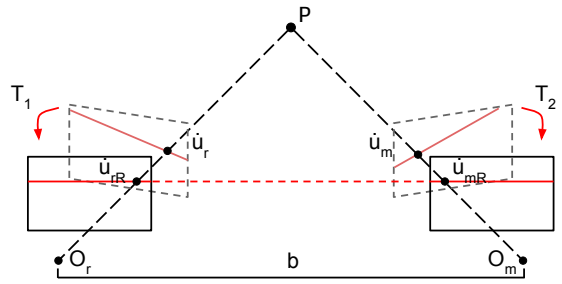


Figura 4: Retificação de I_r e I_m.

Note que, na Equação 4, os *pixels* de I_{rR} são encontrados com mesma coordenada v em I_{mR}, diferindo apenas na coordenada u.

Com as imagens retificadas, a profundidade z_R pode ser estimada buscando a correspondência do *pixel* $\dot{u}_{rR}$ sobre a linha epipolar em I_{mR} que maximiza $ncc(W_{mR}, W_{rR})$ (Equação 5), que é a correlação cruzada normalizada entre W_{mR} e W_{rR} (NCC, *normalized cross correlation*).

$$z_R = \underset{z}{\operatorname{argmax}}\, ncc(W_{mR}, W_{rR}) \tag{5}$$

W_{rR} e W_{mR} são janelas de tamanho 7×7 *pixels* em torno de $\dot{u}_{rR}$ em I_{rR} e $\dot{u}_{mR}$ em I_{mR}, respectivamente. Os valores que $\dot{u}_{mR}$ pode apresentar para cada $\dot{u}_{rR}$ obedecem a Equação 4 variando-se z_R de um valor mínimo de profundidade estipulado, z_{iniR}, até um valor máximo z_{endR}. $\dot{u}_{mR}$ varia de *pixel* em *pixel* dentro dessa faixa. Isso reduz o universo de busca, dispensando a varredura por toda a linha epipolar.

Escolheu-se os valores de z_{iniR} e z_{endR}, no referencial de I_r, iguais a 10 cm e 1000 cm, respectivamente. Desta forma, o fator γ dá a relação entre a profundidade nos referenciais de I_r e I_{rR}, como mostra a Equação 6.

$$\gamma \dot{u}_{rR} = T_1 \dot{u}_b,$$
$$\begin{bmatrix} \gamma u_{rR} \\ \gamma v_{rR} \\ \gamma \end{bmatrix} = T1 \begin{bmatrix} u_b \\ v_b \\ 1 \end{bmatrix}, z_R = \gamma z_b \tag{6}$$

O processo de correspondência é feito com 10 imagens ($M = 10$), obtendo-se assim 10 hipóteses de profundidade $z_{b[1-10]}$ para cada *pixel* $\dot{u}_b$. São realizados ainda dois procedimentos adicionais em sequência para a remoção de possíveis *outliers*, como é proposto em (Concha and Civera, 2015).

- Consistência temporal: uma estimativa de profundidade será um *inlier* se, dentre as 10 hipóteses, pelo menos 5 forem semelhantes. Para estes casos, a profundidade será uma média desses valores semelhantes.
- Consistência espacial: assumindo que a profundidade no mundo varia suavemente, a estimativa da consistência temporal para cada *pixel* será *inlier* se pelo menos um de seus vizinhos apresentar um valor semelhante. Para o conjunto de *inliers* $\dot{u}'_b$, o novo valor estimado de profundidade z'_b será a média entre esses valores semelhantes.

Para evitar o efeito de escada entre camadas estimadas de profundidade, antes do passo da consistência espacial, a profundidade estimada pelo passo de consistência temporal foi filtrada por um filtro de média em cada *pixel* com seus vizinhos de mesmo *label* definido em I_L.

3.5 Estimativa da profundidade em regiões homogêneas

O processo de estimativa da profundidade em áreas homogêneas começa encontrando o plano que melhor se encaixa aos pontos de cada região segmentada. O Algoritmo 1 apresenta este procedimento, retornando o plano ótimo Π_k, em que k se refere ao *label* k.

Algoritmo 1: Algoritmo para estimação da profundidade em áreas homogêneas

Entrada: $\dot{u}'_b$, z'_b, I_L
Saída: Π_k, $inmax$

início
  $\dot{u}_k \in \dot{u}'_b$ para $I_L(u_b, v_b) == k$
  $z_k \in z'_b$ para $I_L(u_b, v_b) == k$
  $inmax = 0$ `/* número de inliers */`
  $distin = \infty$ `/* distância */`
  Π_k `/* Plano ótimo */`
  $\vec{u}_p = [u_k, v_k, z_k]^T$
  para *n de 1 a 500* **faça**
    $\vec{u}^*_p \in \vec{u}_p$ `/* 3 pontos aleatórios */`
    $\vec{v}_1 = \vec{u}^*_p(:,1) - \vec{u}^*_p(:,2)$
    $\vec{v}_2 = \vec{u}^*_p(:,3) - \vec{u}^*_p(:,2)$
    $\vec{n} = \vec{v}_1 \times \vec{v}_2$
    $d = \vec{n}^T \vec{u}^*_p(:,1)$
    $\Pi = [\vec{n}, d]$
    $z_p = \frac{\Pi(4) - \Pi(1)u_k - \Pi(2)v_k}{\Pi(3)}$
    $dist = |z_k - z_p|$
    $inliers = dist <= 1,5\ cm$
    n_in `/* número de inliers */`
    $condição = (n\_in > inmax) \cup ((n\_in == inmax) \cap (dist(inliers) < distin))$
    se $condição == TRUE$ **então**
      $inmax = n\_in$
      $distin = dist(inliers)$
      $\Pi_k = \Pi$
    fim
  fim
  retorna Π_k, $inmax$
fim

Encontrado o plano Π_k, o próximo passo é calcular a profundidade dos *pixels* pertencentes a cada região k por meio da Equação 7.

$$z_L(u_k, v_k) = \frac{\Pi_k(4) - \Pi_k(1)u_k - \Pi_k(2)v_k}{\Pi_k(3)}, \tag{7}$$

em que (u_k, v_k) são os *pixels* pertencentes à região de *label* k. Porém, somente as regiões cujo número de *inliers* ($inmax$) seja de pelo menos 65% do número total de *pixels* reconstruídos destas regiões ($\dot{u}_k$) serão aproximadas por um plano. Isso ajuda a evitar que regiões não planas sejam aproximadas grosseiramente.

4 Resultados e discussões

Para testar o algoritmo, foram utilizadas imagens que apresentam, sobre uma mesa de madeira, uma impressora, um telefone, um calendário, uma revista, um copo, uma xícara, um teclado, uma caixa e uma laranja. Essa mesa se encontra em um escritório com chão quadriculado.

A profundidade real da cena (*ground truth*) é normalmente representada por meio de uma imagem em escala de cinza, denominada mapa de profundidade. Quanto mais próximo da câmera, mais escura é a cor, como pode ser visto na Figura 5.

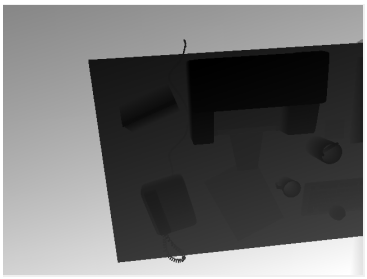


Figura 5: Mapa de profundidade do *ground truth*.

A análise do algoritmo foi feita em termos de *recall* (porcentagem de *pixels* reconstruídos) e erro absoluto em relação ao *ground truth*. Para isso, os resultados obtidos com o método proposto foram comparados com os do DTAM.

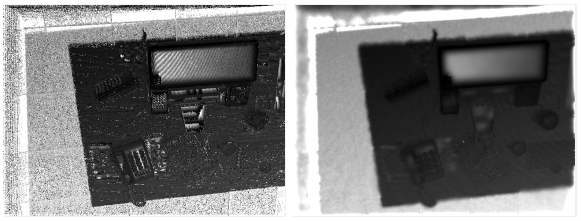
O DTAM estima, primeiramente, a profundidade de todos os *pixels*, e depois tenta encontrar a profundidade correta por meio de um processo iterativo de minimização. Dessa forma, seus resultados foram separados em: a estimativa inicial (Rec. inicial) e o resultado após o processo de minimização (Rec. final). Para testá-lo foram usadas 30 imagens e 32 níveis discretos de profundidade.

Os resultados deste trabalho foram separados de acordo com a técnica utilizada para extração de *pixels* de alto gradiente e subdivididos em: etapa de reconstrução esparsa (Esparso) e de estimativa por aproximação de planos (Pré-denso). Todos esses resultados estão apresentados na Tabela 1. As medidas de erro são dadas em centímetros e o *recall* em porcentagem.

Verifica-se na Tabela 1 que o método do DTAM estima a profundidade de todos os *pixels* da imagem (*recall* de 100%) mas, em média, seu erro médio é 5 vezes maior que os obtidos pelo algoritmo proposto, e sua mediana 9 vezes.

Note que mesmo sem realizar a etapa de reconstrução densa, o método proposto obtém resultados melhores que os apresentados pelo DTAM, que o faz completamente. Um dos motivos é que, mesmo reconstruindo mais *pixels*, a precisão do DTAM acaba sendo muito prejudicada pela profundidade estimada das áreas homogêneas.

A Figura 6 apresenta os mapas de profundidade encontrados utilizando o método do DTAM.

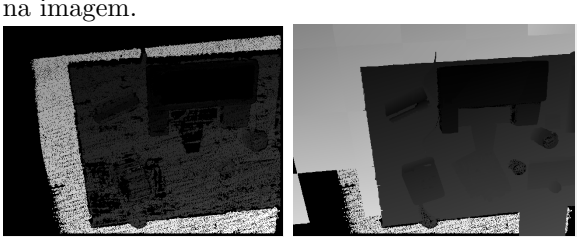


(a) Rec. inicial (b) Rec. final

Figura 6: Mapas de profundidade (DTAM).

É possível ver na Figura 6 erros altos de estimativa sobre diversas partes da impressora e próximo ao telefone (áreas claras que deveriam ser escuras, vide Figura 5). Por não possuírem textura, essas regiões não são bem reconstruídas, e essa estimativa, quando muito grosseira, acaba sendo propagada para a reconstrução final (de Rec. inicial para Rec. final).

A Figura 7 apresenta os mapas de profundidade do resultado do método proposto neste trabalho considerando a reconstrução de "Todos os *pixels*". Pode-se perceber que as regiões cuja reconstrução não é confiável (homogêneas) foram excluídas, sendo representadas por *pixels* pretos na imagem.



(a) Esparso (b) Pré-denso

Figura 7: Mapas de profundidade (método proposto - Todos os *pixels*).

Com a aproximação por planos (Figura 7b), as áreas homogêneas apresentaram uma melhor estimativa de profundidade, resultando, consequentemente, em uma melhor reconstrução $3D$ final.

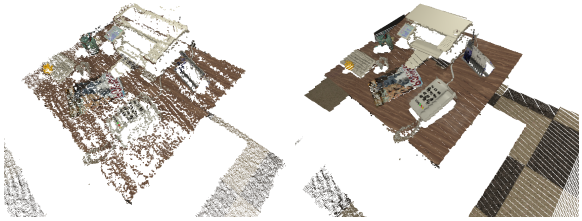
Apesar desses resultados apresentarem erro máximo de 2 vezes menor que o do DTAM, ainda é alto. Isso se deve aos *pixels* de borda da mesa/chão, que pela oclusão, acabam reconstruídos de maneira errada, como parte da região vizinha, levando a erros altos.

Também nota-se que, dentre os 6 resultados do algoritmo proposto, o ideal seria utilizar todos os *pixels* da imagem, afinal, os de reconstrução duvidosa serão descartados pelo algoritmo. Usando-se os *pixels* do interior de regiões com problemas de oclusão, que não foram descartados, foi possível obter uma estimativa menos grosseira, fato facilmente visto ao se comparar os resultados de "Apenas bordas (Pré-denso)"com de "Todos os *pixels* (Pré-denso)"na Tabela 1. A média, a mediana e o desvio padrão do erro apresentaram valores menores no segundo caso. Além da precisão, o *recall*, consequentemente, aumenta.

Tabela 1: Resultados da reconstrução $3D$.

Método	Etapa	Erro (cm)					*Recall*
		Média	Mediana	Desv. Padrão	Mínimo	Máximo	(%)
DTAM	Rec. inicial	24,27	8,28	30,98	0	168,17	100
	Rec. final	23,09	4,06	45,39	0	222,37	**100**
Apenas bordas	Esparso	3,86	0,76	13,43	0	96,07	7,33
	Pré-denso	9,93	0,88	23,95	0	96,6	53,69
Alto gradiente	Esparso	3,45	0,72	12,63	0	96,01	8,53
	Pré-denso	9,73	0,72	24,25	0	96,44	53,85
Alto gradiente + vizinhos	Esparso	2,78	0,69	11,05	0	97,65	13,2
	Pré-denso	5,79	0,57	20,94	0	104,25	72,52
Método de Canny	Esparso	3,36	0,76	12,25	0	96,18	9,16
	Pré-denso	3,87	0,59	15,49	0	96,61	49,97
Método de Canny + filtro de média	Esparso	3,47	0,74	12,61	0	96,19	8,54
	Pré-denso	3,95	0,58	15,56	0	96,26	49,55
Todos os *pixels*	Esparso	1,72	0,67	6,9	0	98,79	48,35
	Pré-denso	3,31	0,46	15,27	0	102,19	**92,29**

A Figura 8 apresenta a reconstrução $3D$ referente aos mapas de profundidade apresentados na Figura 7.



(a) Esparso (b) Pré-denso

Figura 8: Reconstrução $3D$ (método proposto).

Nota-se que uma região do chão foi reconstruída no plano da mesa, Figura 8b. Isso ocorreu pois regiões de planos distintos foram segmentadas juntas. Entretanto, este erro será corrigido na etapa de minimização.

5 Conclusão e trabalhos futuros

Neste artigo foi proposta uma metodologia de se estimar conhecimentos prévios da cena antes de se realizar a reconstrução $3D$ densa propriamente dita. A intenção é aumentar a precisão da estimativa final, mantendo um baixo tempo de processamento.

O método proposto foi testado com várias técnicas de seleção de *pixels* e comparado com o método DTAM, o qual representa um bom compromisso entre tempo de processamento e qualidade de reconstrução, sendo utilizado em aplicações de tempo real.

Mesmo sem realizar a etapa de reconstrução $3D$ densa, o método proposto se destaca por atingir uma taxa de *recall* de $92,3\%$ com uma precisão sete vezes maior que a conseguida pelo DTAM, o qual ainda realiza a etapa de minimização e reconstrução densa. Tal resultado pode ser considerado promissor e sugere que, com a metodologia proposta, resultados ainda melhores poderão ser conseguidos após a etapa final de reconstrução.

Como trabalho futuro, deve-se iniciar a análise de complexidade e tempo de processamento do método.

Referências

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P. and Susstrunk, S. (2010). SLIC Superpixels, *EPFL Technical Report 149300* (June): 15.

Concha, A. and Civera, J. (2015). DPPTAM: Dense Piecewise Planar Tracking and Mapping from a Monocular Sequence, *Intelligent Robots and Systems (IROS 2015), 2015 IEEE/RSJ International Conference on* pp. 5686–5693.

Concha, A., Hussain, W., Montano, L. and Civera, J. (2014). Manhattan and Piecewise-Planar Constraints for Dense Monocular Mapping, *Robotics-proceedings.Org* .

Concha, A., Hussain, W., Montano, L. and Civera, J. (2015). Incorporating scene priors to dense monocular mapping, *Autonomous Robots* **39**(3): 279–292.

Fraundorfer, F., Schindler, K. and Bischof, H. (2006). Piecewise planar scene reconstruction from sparse correspondences, *Image and Vision Computing* **24**(4): 395–406.

Handa, A., Newcombe, R. A., Angeli, A. and Davison, A. J. (2012). Real-time camera tracking: When is high frame-rate best?, *Proc. of the European Conference on Computer Vision(ECCV)*.

Li, L., Sun, L., Guo, J. and Li, S. (2015). Image Segmentation Based on Superpixels and Region Merging, **23**: 8787–8797.

Newcombe, R. A., Lovegrove, S. J. and Davison, A. J. (2011). {DTAM}: Dense Tracking and Mapping in Real-Time, *Int. Conf. on Computer Vision (ICCV)* pp. 2320–2327.

Pagani, A., Gava, C., Cui, Y., Krolla, B., Hengen, J.-M. and Stricker, D. (2011). Dense 3D Point Cloud Generation from Multiple High-resolution Spherical Images, *International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST)* pp. 1–8.

Wu, T.-P., Yeung, S.-K., Jia, J. and Tang, C.-K. (2010). Quasi-dense 3D reconstruction using tensor-based multiview stereo, *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 4, IEEE, pp. 1482–1489.

Xu, L., Yan, Q., Xia, Y. and Jia, J. (2012). Structure extraction from texture via relative total variation, *ACM Transactions on Graphics* **31**(6): 1.