

Graph Matching em Redes de Pontos em Imagens

Gustavo Pfeiffer¹

¹Escola Politécnica / COPPE

Universidade Federal do Rio de Janeiro, RJ – Brazil

gustavo_tp@poli.ufrj.br

Abstract. This report describes an adaptation of a graph matching method for the problem of feature matching of image processing.

Resumo. Este relatório descreve uma adaptação de um método de graph matching para o problema de feature matching de processamento de imagens.

1. Introdução

Neste projeto, implementamos uma adaptação do algoritmo de Pedarsani et al.[1] de *graph matching* aplicado ao problema de *feature matching* de processamento de imagens.

1.1. Feature Matching

Feature matching é um problema de processamento de imagens que deve ser resolvido para diversas aplicações como panografia e reconstrução estéreo.

Em panografia (Figura 1(a)), são tiradas várias fotografias com a câmera na mesma posição, mas rotações diferentes. As fotografias são então sobrepostas após uma transformação projetiva gerando uma imagem mais ampla.

Em reconstrução estéreo (Figura 1(b)), são tiradas duas fotografias após uma translação da câmera, simulando a separação entre os dois olhos humanos, de forma que é possível calcular a profundidade dos objetos que aparecem nas duas fotografias.

Em ambos os problemas, é necessário saber qual ponto em uma imagem corresponde a qual ponto na outra. Não são necessários muitos pontos, então buscam-se pontos chamados de *features* em cada imagem para fazer o *matching*. Uma *feature* é um ponto da imagem que é fácil de se encontrar na outra imagem, ou seja, inambíguo e que não se deforma muito após uma translação da câmera (tipicamente quinas e pontas de objetos). Para isto utilizamos o método de Harris[2].



Figura 1. (a) Exemplo de panografia. *Features* marcadas em magenta. (Adaptado de [2]). (b) Exemplo de reconstrução estéreo. À esquerda, imagens utilizadas na reconstrução, à direita, modelo de pontos 3D extraído, visualizado de vários ângulos.

1.1.1. Comparando *features*

Para comparar duas *features*, podemos comparar uma vizinhança $L \times L$ (11×11 no caso da panografia e 41×41 no caso da reconstrução estéreo), com 3 canais de cor, normalizada em torno de cada *feature*, utilizando NCC (*Normalized Cross-Correlation*). Isto é, normaliza-se um vetor $u \in \mathbb{R}^{3L^2}$ utilizando:

$$N(u) = \frac{u - \frac{\vec{1}\vec{1}^T u}{3L^2}}{\|u - \frac{\vec{1}\vec{1}^T u}{3L^2}\|_2}$$

E a NCC é calculada seguindo:

$$\text{NCC}_{u,v} = \langle N(u), N(v) \rangle$$

De forma que sabemos que:

$$\|N(u) - N(v)\|_2^2 = 2 - 2 \cdot \text{NCC}_{u,v}$$

Pode-se então utilizar o algoritmo húngaro de emparelhamento[3], que procura um conjunto de pares (sem repetição de uma mesma *feature* em dois pares) tal que a soma das NCCs é maximizada. Neste relatório, referenciaremos esta técnica pelo nome de *técnica ingênua*.

1.1.2. Seleção de pares

Na prática muitos pares, normalmente a maioria, são mapeados incorretamente, de forma que é necessário fazer uma pós-seleção. Para isto utiliza-se o algoritmo RANSAC[2]. O RANSAC é uma técnica que repetidamente aplica um algoritmo de mínimos quadrados (o algoritmo que resolve o problema em questão, como panografia ou reconstrução estéreo) sobre um subconjunto aleatório dos pares de entrada contendo o mínimo de pares necessários para se resolver o problema de mínimos quadrados, e valida o resultado com todos os pares de entrada. Se o conjunto de treinamento só continha *inliers*, ou seja, pares corretos, então um grande número de pares passará na etapa de validação. Caso contrário, passam praticamente apenas os pares utilizados no treinamento. Repetindo-se várias vezes, utiliza-se o maior conjunto de pares que tiver passado na validação, que é considerado de fato o conjunto de *inliers* do emparelhamento. Ao final a panografia ou reconstrução estéreo é calculada utilizando apenas estes *inliers*.

1.2. Proposta

O que propomos é substituir a combinação algoritmo húngaro - NCC, que compara apenas *features* umas com as outras, sem levar em conta a posição das mesmas na imagem (espera-se que *features* próximas numa imagem também sejam próximas na outra, e vice-versa). Resolveremos isto utilizando *graph matching*, onde cada imagem apresenta um grafo completo cujos vértices (as *features*) são caracterizados pela vizinhança normalizada, e cujas arestas, pela distância euclidiana entre as *features*.

2. Nosso algoritmo

Nosso algoritmo faz diversas adaptações ao algoritmo de Pedarsani et al.. Mostramos nesta seção as principais modificações que fizemos e como instanciamos este arcabouço ao problema de *feature matching*. Utilizaremos a mesma notação do artigo.

Primeiramente, supomos que o grafo gerador G é determinístico, de forma que $P[X_{1i}, X_{2i}] = P[X_{1i}] \cdot P[X_{2i}]$, então

$$P[U_1 = U_2 | X_i] = \frac{\frac{1}{n_{\max}} \int q_i(x_{1i}, y_i) q_i(x_{2i}, y_i) dP(y_i)}{\left(\int q_i(x_{1i}, y_i) dP(y_i) \right) \cdot \left(\int q_i(x_{2i}, y_i) dP(y_i) \right)} \quad (1)$$

onde $n_{\max} = \max\{n_1, n_2\}$, e n_1 e n_2 são o número de vértices em cada grafo.

Isto nos permite eliminar a utilização do algoritmo de Monte-Carlo para iterar sobre os possíveis valores de $P[M_i]$, pois a equação final se simplifica:

$$\begin{aligned} P[U_1 = U_2 | F_{u_1}, F_{u_2}] &= \sum_{b \in \{0,1\}^m} \frac{1}{n_{\max}} \left(\prod_{i=1}^m P[M_i = b_i] \right) \dots \\ &\dots \left(\prod_{i=1}^l \frac{P[X_i | U_1 = U_2]}{P[X_i]} \right) \left(\prod_{\substack{i=l+1 \\ \text{s. a } b_i=1}}^{l+m} \frac{P[X_i | U_1 = U_2, M_i = 1]}{P[X_i | M_i = 1]} \right) \\ &= \frac{1}{n_{\max}} \left(\prod_{i=1}^l \frac{P[X_i | U_1 = U_2]}{P[X_i]} \right) \left(\prod_{i=l+1}^{l+m} P[M_i = 0] + P[M_i = 1] \frac{P[X_i | U_1 = U_2, M_i = 1]}{P[X_i | M_i = 1]} \right) \end{aligned}$$

Além disto, a forma de computar as integrais da Equação 1 é diferente: Pedarsani et al. fazem um somatório nos valores y (variáveis são discretas), enquanto nós fazemos um somatório nas ocorrências de y , ou seja, em cada vértice (porque as variáveis são contínuas).

Pedarsani et al. consideram os componentes das *fingerprints* independentes, nós tivemos que fazer algumas adaptações neste ponto. Primeiramente, a vizinhança normalizada ($3L^2 = 363$ variáveis no caso da panografia e 5043 no caso da reconstrução estéreo) é tratada como uma única variável. Segundo, as distâncias são altamente correlacionadas, pois o grafo é euclidiano: Teoricamente, 3 vértices-âncora já são suficientes para desambiguar *features* com vizinhanças iguais, então optamos por limitar o número de iterações a um número muito pequeno (5 iterações, resultando em 8 vértices-âncora na última iteração), e sempre utilizar $V_1^\tau = V_1$ e $V_2^\tau = V_2$.

Para os q_i , utilizamos as seguintes funções. Para a vizinhança normalizada, utilizamos uma função gaussiana multivariável:

$$q(x, y) = \exp \left(-\frac{1}{2} \frac{\|N(x) - N(y)\|^2}{\sigma_C^2} \right) = \exp \left(\frac{NCC_{x,y} - 1}{\sigma_C^2} \right)$$

Observe que não é necessário normalizar q porque na Equação 1 as constantes se anulam. Para a distância, utilizamos uma gaussiana no logaritmo da distância:

$$q(x, y) = \exp \left(-\frac{1}{2} \frac{(\ln(x^2) - \ln(y^2))^2}{\sigma_E^2} \right)$$

Além disto, utilizamos uma distribuição $p(y)$ diferente para as distâncias: ao invés de utilizar todas as arestas de cada grafo para a distribuição empírica, utilizamos apenas as arestas ligadas ao vértice-âncora em questão. No entanto, calcular este somatório pode ser muito custoso para um número de iterações grande (uma iteração teria custo $O(n_1 n_2 n_{\text{âncoras}}(n_1 + n_2))$). Optamos então por calcular o somatório utilizando Monte-Carlo com 10 amostras.

3. Escolha dos parâmetros σ_E e σ_C

Para estimar os parâmetros σ_E e σ_C do algoritmo, utilizamos o seguinte método: Executamos a técnica ingênua e aplicamos RANSAC, para alguns pares de imagens. Calculamos então $E[||X_1 - X_2||^2]$ para os *inliers* encontrados pelo RANSAC, o que nos dá os valores de σ_E^2 e $3L^2\sigma_C^2$. No entanto, observamos experimentalmente que aumentar σ_C em $6 \sim 20 \times$ nos dá resultados melhores, enquanto alterações em σ_E não causam grandes alterações nos resultados.

Foram utilizados os mesmos valores de σ_E e σ_C para todas as imagens de cada aplicação.

4. Resultados

Nosso algoritmo foi avaliado em duas aplicações: panografia e reconstrução estéreo. Avaliamos o desempenho dos algoritmos observando a média e o desvio padrão do número de *inliers* encontrados pelo algoritmo RANSAC em 10 execuções.

4.1. Panografia

Em panografia, a técnica ingênua gera aproximadamente os mesmos resultados que o nosso método (veja na Tabela 1). Nosso método se sobressai quando há múltiplas *features* com redondezas muito parecidas, como no caso da Figura 2-(c/h), porque nesse caso a informação de distância é importante para desambiguar as *features*. De toda forma, em todos os casos, não houve diferenças perceptíveis na panografia gerada, porque já se tinha um número suficientemente alto de *inliers*.

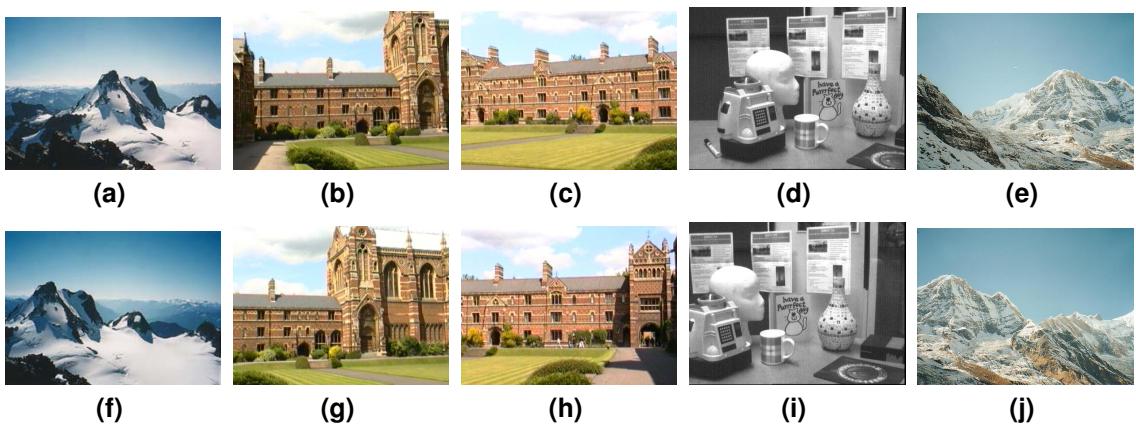
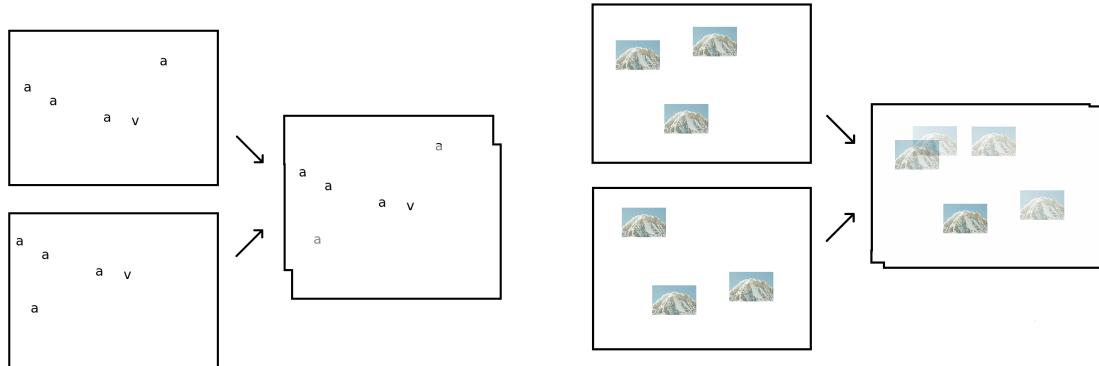


Figura 2. Casos reais para panografia. ((a/f) e (e/j) retirados de [2] e (b/g), (c/h) e (d/i) de [4])

Tabela 1. Comparação do número de *inliers* obtido pelo RANSAC em panografia

Caso	nossa técnica $(\mu \pm \sigma \text{ inliers})$	técnica ingênua $(\mu \pm \sigma \text{ inliers})$	$\min\{n_1, n_2\}$	melhoria relativa $(\frac{\mu_{\text{nossa}} - \mu_{\text{ingênua}}}{\min\{n_1, n_2\}})$
Fig. 2-(a/f)	65.7 ± 3.1	63.8 ± 1.0	112	01.7%
Fig. 2-(b/g)	274.5 ± 3.5	266.9 ± 3.6	483	01.2%
Fig. 2-(c/h)	140.7 ± 4.0	117.4 ± 3.5	490	04.8%
Fig. 2-(d/i)	270.4 ± 3.3	250.8 ± 2.3	425	04.6%
Fig. 2-(e/j)	61.1 ± 6.5	55.6 ± 6.2	477	01.6%
Fig. 3-(a)	20.0 ± 0.0	10.0 ± 0.0	25	40.0%
Fig. 3-(b)	71.0 ± 0.0	71.0 ± 0.0	213	00.0%

Testamos dois casos sintéticos para confirmar esta hipótese. No caso da Figura 3-(a), observe que nosso algoritmo ancora primeiro as *features* da letra “v”, e depois mapeia as dos “a”s, obtendo com sucesso o dobro de *inliers* que o da técnica ingênua. No caso da Figura 3-(b), no entanto, não há um ponto de referência, então o nosso algoritmo também não consegue desambiguar as *features*, gerando o mesmo resultado da técnica ingênua. Ou seja, a técnica erra o primeiro vértice-âncora, sem conseguir voltar atrás, o que a leva a errar todos os vértices seguintes.



(a) Nossa técnica encontra todos os 20 *inliers*, enquanto a técnica ingênua encontra apenas metade (vide Tabela 1). No entanto ambas computam a panografia corretamente.

(b) Ambas as técnicas só encontram metade dos *inliers*, e erram a panografia. O correto seria mapear as duas instâncias do topo da imagem de cima às duas de baixo da imagem de baixo, obtendo 142 *inliers*.

Figura 3. Casos sintéticos para panografia. ((b) adaptado de [2])

4.2. Reconstrução estéreo

O caso da reconstrução estéreo é notavelmente mais delicado que o da panografia. Aqui, a informação de cor (vizinhança normalizada) é menos confiável, e as distâncias também se distorcem mais. Além disto, há muito mais variáveis na etapa de mínimos quadrados, de forma que o RANSAC tem mais dificuldade em filtrar *inliers* (muitos *outliers* são aprovados), e sem intervenção humana é praticamente impossível que o

objeto reconstruído não fique distorcido, ou sequer se assemelhe de alguma forma ao original.

Apesar de não eliminar estas dificuldades, notamos que a nossa técnica gera mais *inliers* que a ingênua na etapa do RANSAC. Veja os resultados na Tabela 2.



Figura 4. Casos para reconstrução estéreo.

Tabela 2. Comparação do número de *inliers* obtido pelo RANSAC em reconstrução estéreo

Caso	nossa técnica $(\mu \pm \sigma \text{ inliers})$	técnica ingênua $(\mu \pm \sigma \text{ inliers})$	$\min\{n_1, n_2\}$	melhoria relativa $(\frac{\mu_{\text{nossa}} - \mu_{\text{ingénua}}}{\min\{n_1, n_2\}})$
Fig. 4-(a/b)	36.5 ± 0.7	30.3 ± 3.1	46	13.5%
Fig. 4-(c/d)	60.2 ± 3.0	45.0 ± 2.3	76	20.0%

5. Conclusão

Podemos concluir que, no caso da panografia, a nossa técnica trouxe poucas vantagens, considerando que é um algoritmo mais custoso e mais difícil de implementar e configurar. Mesmo nos casos em que os resultados numéricos são melhores, há pouca diferença qualitativa.

Na reconstrução estéreo, os resultados foram significativamente melhores, apesar de ainda não termos uma reconstrução estéreo confiável, o que se deve no entanto provavelmente ao mau desempenho do método de reconstrução estéreo escolhido.

Referências

- [1] P. Pedarsani, D. R. Figueiredo, and M. Grossglauser, “A bayesian method for matching two similar graphs without seeds,” *IEEE 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2013.
- [2] R. Szeliski, *Computer Vision: Algorithms and Applications*, 1st ed. New York, NY, USA: Springer-Verlag New York, Inc., 2010.
- [3] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1990.
- [4] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.