Criação Automática de Panoramas

Porque as vezes não dá pra dar um passinho pra trás

Introdução

Uma câmera tradicional consegue capturar muito pouco... Apenas um quadradinho de menos de 65° × 46° (28mm), de um mundo inteiro, 360º

Introdução

Neste projeto, tentamos obter uma fotografia 360º combinando diversas fotos em um mosaico que cobre todas as direções. Etapas:

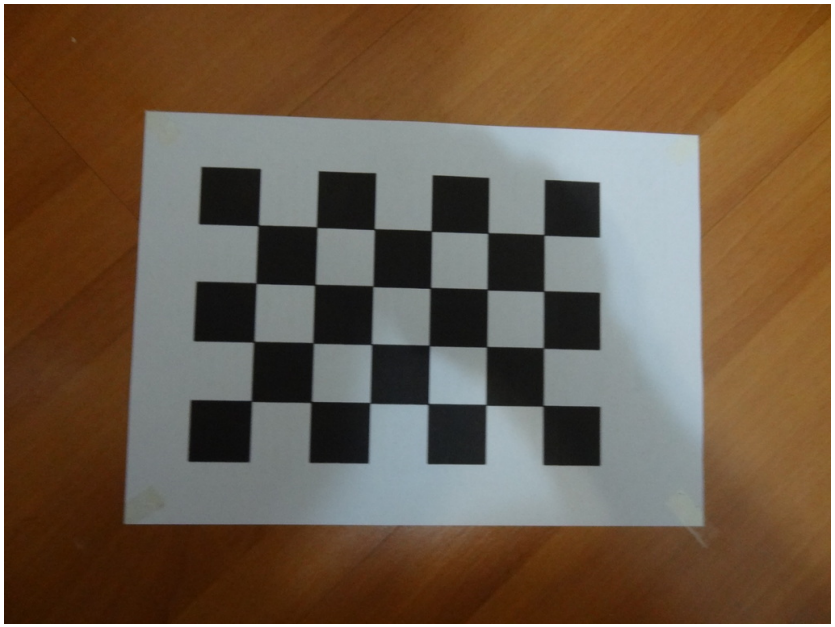
- Calibração da Câmera
- Extração de pontos de interesse
- Casamento dos pontos de interesse entre pares de imagens
- Cálculo de orientação (Relativa) de cada imagem
- Normalização de cores
- Composição da imagem final, com blending e HDR

Calibração da Câmera

O programa deveria ser capaz de analisar as imagens de entrada e calcular automaticamente os parâmetros intrínsecos da câmera, inclusive distorções não-lineares.

Infelizmente, não houve tempo de implementar isso. Alternativamente, fizemos a calibração prévia da câmera utilizando um padrão quadriculado, como descrito por Zhang 99.

Calibração da Camera



Com os parâmetros intrínsecos da câmera, podemos mapear as coordenadas (x,y) de um pixel em um vetor (x,y,z) com a direção do ponto em relação a câmera.

Extração de Pontos de Interesse

É utilizado SURF para detectar e descrever pontos de interesse em cada uma das imagens.

Os parâmetros intrínsecos da câmera são usados para mapear cada ponto de interesse em um vetor (x,y,z) com a direção do ponto em relação a câmera.

Casamento de Imagens

Utiliza-se um casamento guloso para encontrar pares de pontos de interesse candidatos à matching.

A partir de alguns pares de pontos correspondentes, podemos estimar a matriz de rotação usando o método de Registro 3-D de Paul J. Besl and Neil D. McKay, modificado para não considerar translação (Uma vez que a câmera apenas rotaciona em torno do centro da cena)

Casamento de Imagens

Por fim, utilizamos RANSAC para encontrar os pares de pontos que casam, de forma robusta.

O conjunto será considerado válido que houver pelo menos 50 pares de pontos de interesse que casem após a rotação com erro máximo de 2°.

Orientação de Imagens

É fácil encontrar a orientação relativa entre 2 imagens, mas encontrar a orientação relativa de N imagens minimizando os erros é bem mais complicado!

A técnica implementada é rudimentar:

- É criado um grafo onde cada imagem é um vértice e cada match é uma aresta.
- Calculamos a árvore geradora máxima, isto é, descartamos matches redundantes, ficando apenas com os melhores.

Orientação de Imagens

- Se o grafo for desconexo, pegamos a maior componente e descartamos as demais imagens.
- Assumimos umas das imagem como referência, percorremos o grafo calculando a rotação de cada aresta e propagando a rotação relativa para cada imagem encontrada.

Resultados razoáveis para poucas imagens.

Orientação de Imagens

Um algoritmo melhor deveria fazer otimização global, considerando todos os matches, ao invés de descartá-los.

É possível adaptar o algoritmo Shape And Motion From Image Streams (Tomasi, Kanade) para resolver este caso:

Orientação de Imagens

- Podemos ignorar todas as etapas para tratar translação, uma vez que a câmera já está centralizada na origem.
- Os pontos observados são convertidos em vetores (x,y,z), obtidos pelos parâmetros intrínsecos da câmera.
- As matrizes de câmera são 3x3, isto é, são matrizes de rotação, e não de projeção ortográfica.
- Todo o resto permanece inalterado.

Orientação de Imagens

- Infelizmente não chegamos a implementar o método completo, especialmente devido às dificuldades em implementar a tolerância a oclusão. =(

Normalização de Cores

Separamos as diferenças de cor em 2 componentes:

- Intensidade e Balanço de cor:
  F(rgb) -> rgb
- Atenuação pela distância do centro:
  G(raio) -> Escala

Para todos os pontos correspondentes em 2 imagens, devemos otimizar:

F1(rgb1)*G1(raio1) = F2(rgb2)*G2(rgb2)

Normalização de Cores

É dificil resolver F e G simultaneamente, mas podemos otimizá-los separadamente.

Também é difícil otimizar

F1(x1)=F2(x2)*g2(raio2)/g1(raio1)

mas é fácil otimizar F1(x2)=f2(x2)*g2(raio2)/g1(raio1)

Por fim, iteramos até a convergência.

Normalização de Cores

Não chegamos a implementar a otimização de G(raio)

Para a otimização de F, inicialmente assumimos uma transformação linear (Matriz 3x3) e otimizamos por mínimos quadrados.

- Resultados péssimos, havia muito ruído!

Finalmente, implementamos apenas um fator de escala, com resultados aceitáveis

Blending e HDR

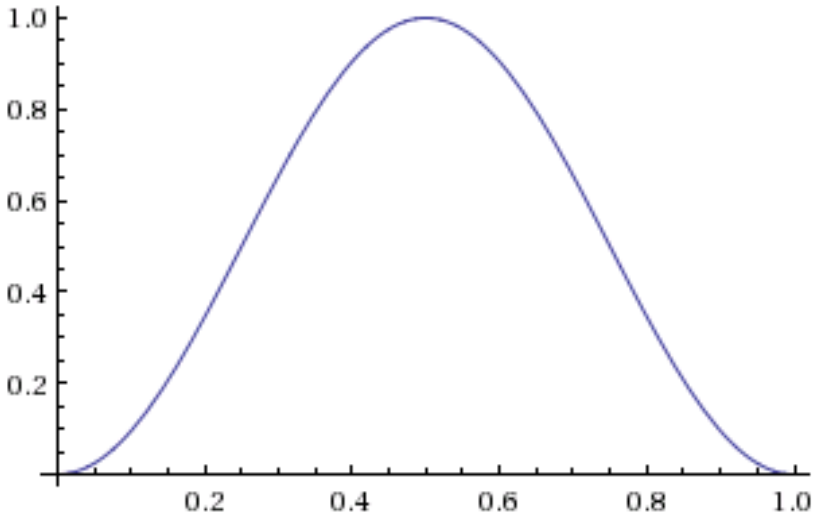
Cada pixel da imagem final é mapeado para uma vetor direção(x,y,z) de acordo com a projeção escolhida (Equiretangular ou cúbica).

Para cada imagem, re-orientamos o vetor em relação a aquela imagem e convertemos para a coordenada do pixel correspondente (se for válida) através dos parâmetros intrínsecos e pegamos a cor correspondente (normalizada).

Blending e HDR

A cor do pixel na imagem final é calculada por uma média ponderada das cores em cada imagem correspondente, onde o peso de cada imagem é dado por:

$W = \sin^2(\pi.x).\sin^2(\pi.y).\sin^2(\pi.brilho/255).$



Blending e HDR

Esta fórmula garante transição suave próximo das bordas (blending), e diminui a influência de áreas sub-expostas ou sobre-expostas (HDR).

Demonstração

Perguntas?!