Fast Panorama Stitching for High-Quality Panoramic Images on Mobile Phones

---

2 main categories of current image stiching approaches

- **transition smoothing**: reduce color differences between source images to make seams invisible and remove stitching artifacts
  - **Alpha blending**: fast transition smoothing approach, but it cannot avoid ghosting problems caused by object motion and small spatial alignment errors
  - **Gradiend Domain Image Blending approaches**: can reduce color differences and smooth color transitions using gradient domain operations, producing high-quality composite images
- **optimal seam finding**: search for seams in overlapping areas along paths where differences between source images are minimal
- **combination**: optimal seams first, if seams and stitching artifacts are visible, transition smoothing to reduce color differences to hide the artifacts then
  - graph cut -> find optimal seams
  - poisson blending -> smoothing color transitions

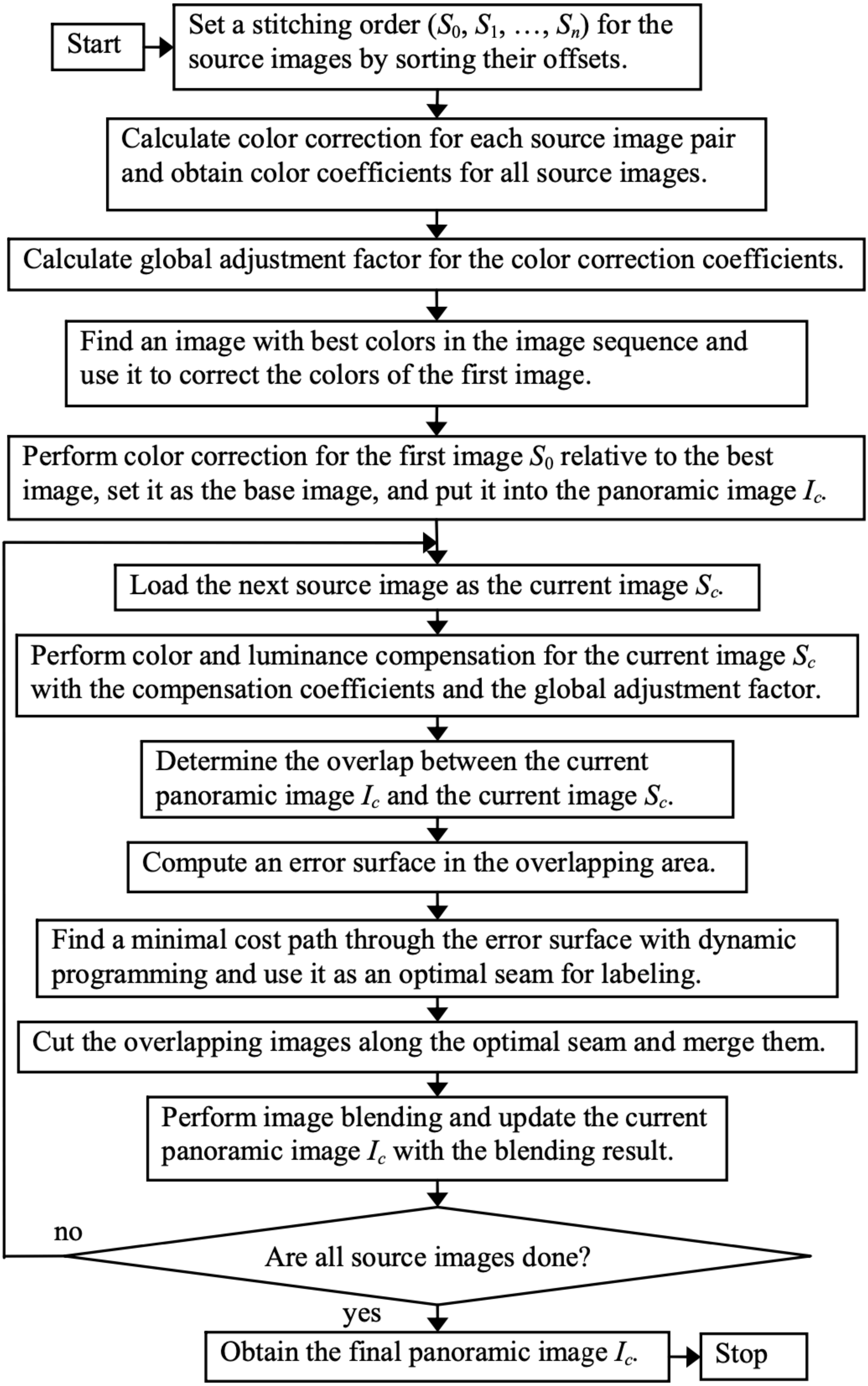
Problem

- computational and memory costs are high
- pixels are easy saturated in color correction
- don't work well for source images in very different colors and luminance
- linear blending, moving objects on the overlapping areas will cause ghosting artifacts

Basic Steps

- **Color Correction**
  - color correction for all source images to reduce color differences
  - smoothen remaining color transitions between adjacent images
- **Image Labeling**
  - error surface is constructed with squared differences between overlapping images
  - low-cost path is found through the error surface by dynamic programming and used as an optimal seam to create labeling
- **Image Blending Operations**
  - **linear blending** -> source images are similar in color and luminance
  - **poisson blending** -> colors remain too different

Summary


Start
Set a stitching order (S_0, S_1, …, S_n) for the source images by sorting their offsets.
Calculate color correction for each source image pair and obtain color coefficients for all source images.
Calculate global adjustment factor for the color correction coefficients.
Find an image with best colors in the image sequence and use it to correct the colors of the first image.
Perform color correction for the first image S_0 relative to the best image, set it as the base image, and put it into the panoramic image I_c.
Load the next source image as the current image S_c.
Perform color and luminance compensation for the current image S_c with the compensation coefficients and the global adjustment factor.
Determine the overlap between the current panoramic image I_c and the current image S_c.
Compute an error surface in the overlapping area.
Find a minimal cost path through the error surface with dynamic programming and use it as an optimal seam for labeling.
Cut the overlapping images along the optimal seam and merge them.
Perform image blending and update the current panoramic image I_c with the blending result.
Are all source images done?
no
yes
Obtain the final panoramic image I_c.
Stop

- don't need to keep all source images in memory due to the sequential stitching
- dynamic programming for optimal seam finding allowing image labeling mush faster than using graph cut
- combination of color correction and image blending allow to construct high-quality panoramic image
- high-quality panoramic images from long image sequences with very different colors and luminance
- work well on both indoor and outdoor scenes

Algorithm

Color and Luminance Compensation

- images captured in paper: automated setting s for focus, exposure, and white balance
- compute light averages in the overlap area by linearizing the gamma-corrected RGB values
- instead of the default gamma-corrected RGB. In an image sequence S_0, S_1, … S_i, … S_n, suppose $S_{i\text{-}1}$ and S_i are adjacent images, and S^o_{i-1} and S^o_i are where image overlap. We compute color correction coefficients for image S_i by linearizing the gamma-corrected RGB values as

  $$\alpha_{c,i} = \frac{\sum_p (P_{c,i-1}(p))^\gamma}{\sum_p (P_{c,i}(p))^\gamma} \qquad c \in \{R, G, B\} \; (i = 1,2,3,\cdots,n), \tag{1}$$

  where $P_{c,i\text{-}1}(p)$ is the color value of pixel p in image S^o_{i-1}; $P_{c,i}(p)$ is the color value of pixel p in image S^o_i; and γ is a gamma coefficient. Usually we set γ to 2.2. For the first image S_0, we set $\alpha_{c,0}$ to 1. To avoid saturating color values, we perform a
- global adjustment in the whole image sequence g_c for each color channel c
-

so that the overall adjustments $g_c\alpha_{c,i}$ approximate 1 by solving the least-squares equation

$$\min_{g_c} \sum_{i=0}^{n} (g_c \alpha_{c,i} - 1)^2 \quad c \in \{R, G, B\}. \tag{2}$$

Equation (2) is a quadratic function in adjustment g_c which can be solved in closed form by setting the derivative to 0,

$$g_c = \frac{\sum_{i=0}^{n} \alpha_{c,i}}{\sum_{i=0}^{n} \alpha_{c,i}^2} \quad c \in \{R, G, B\}\ (i = 0,1,\ldots,n). \tag{3}$$

With the correction coefficients $\alpha_{c,i}$ and the global adjustment factor g_c, we perform color correction for the whole image S_i,

$$P_{c,i}(p) \leftarrow (g_c \alpha_{c,i})^{1/\gamma} P_{c,i}(p),\ c \in \{R, G, B\}\ (i = 0,1,\ldots,n), \tag{4}$$

where $P_{c,i}(p)$ is the color value of pixel p in image S_i in color channel $c \in \{R, G, B\}$. Since the input and output values are gamma-corrected, we also gamma-correct adjustments $g_c\alpha_{c,i}$.

- > 【gamma corrected】
  >
  > https://blog.csdn.net/candycat1992/article/details/46228771
  >
  > https://www.cnblogs.com/qiqibaby/p/5325193.html
- choosing the best image: callesthetic judgment
  - in paper: select the image with most similar means in the R,G, and B chnnels(gray world assumption ofter used in white balancing)
- advantages

correction. Linearizing the light while calculating the correction factors matches the colors better than if the averages were calculated in gamma-corrected RGB, and the global adjustment for color correction coefficients attenuates the corrections, reducing accumulation of corrections that may lead to color saturation.

Optimal Seam Finding and Image Labeling

- ghosting problem caused by the motion
- create labeling for all pixels in the composite image, and merge source images along the optimal seams

> each pixel in the composite image comes from only one source image, the ghosting problems can be avoided

- dynamic programming: differ the least

(a) (b) (c) (d) (e) (f)

1. error surface $e = (I_c^o - S_c^o)^2$ (c)
2. (d)

   path through this surface. We scan the error surface row by row and compute a cumulative minimum squared difference E

   $$E(h,w) = e(h,w) + \min(E(h-1,w-1), E(h-1,w), E(h-1,w+1)), \tag{6}$$

3. (e) all possible paths
4. (f) The optimal path m_c can be obtained by tracking back the paths with a minimal cost from bottom to top.

- color correction improve quality of optimal seam finding and image labeling

Transition Smoothing with Image Blending

- **simple linear blending**: images are similar in color and luminance after color correction
  - on a band that is δ pixels wide on both sides of the seam

seam, as shown in Fig. 6. The new color value of pixel p in the overlapping area can be calculated by a weighted combination of the corresponding pixels

- $$P_{I_{c,new}}(p) = \frac{d_1^n P_{I_c}(p) + d_2^n P_{S_c}(p)}{d_1^n + d_2^n}, \qquad (7)$$

  where d_1 and d_2 are distances from pixel p to boundaries; $P_{I_{c,new}}(p)$ is the new color of pixel p; $P_{I_c}(p)$ is the color of
- n: order

- **Poisson Blending**
  - perform image blending in the gradient domain
  - gradiend vector field (G_x, G_y), with gradients of source images using the labeling obtained using optimal seams

    the calculated seam). A divergence field $div(G)$ is then computed from the gradient vector field,

    $$div(G) = \frac{\partial G_x}{\partial x} + \frac{\partial G_y}{\partial y}. \qquad (8)$$

    We use the divergence field as a guidance to construct a Poisson equation

    $$\nabla^2 I(x, y) = div(G), \qquad (9)$$

  - where ∇^2 is the Laplacian operator

    $$\nabla^2 I(x, y) = \frac{\partial^2 I(x, y)}{\partial x^2} + \frac{\partial^2 I(x, y)}{\partial y^2}. \qquad (10)$$

    In practical implementation, we need to use the discrete form of Equation (9)

    $$I(x+1, y) + I(x-1, y) + I(x, y+1) + I(x, y-1) - 4f(x, y) = G_x(x, y) - G_x(x-1, y) + G_y(x, y) - G_y(x, y-1) \qquad (11)$$

  - solve linear prtical differential equation by fixing the colors at the seam and solving new colors $I(x, y)$ over the gradient field -> iterative conjugate gradients solver