同济大学
TONGJI UNIVERSITY

IPMV-Experiment-4

Lab 2 Blending with OpenCV

课程名称：图像处理与机器视觉

实验地点：嘉定校区智信馆 131

指导教师：Lei Jiang， Rui FAN

姓　　名：姚天亮

学　　号：2150248

Task
Try to implement and test blending algorithm.

Simple linear blending

```
cv::Mat linearBlending(const cv::Mat& img_1, const cv::Mat& img_2, const cv::Mat& weights)
{
   return weights.mul(img_1) + (cv::Scalar(1.0, 1.0, 1.0) - weights).mul(img_2);
}
```

Laplace blending

Construct a Gaussian pyramid

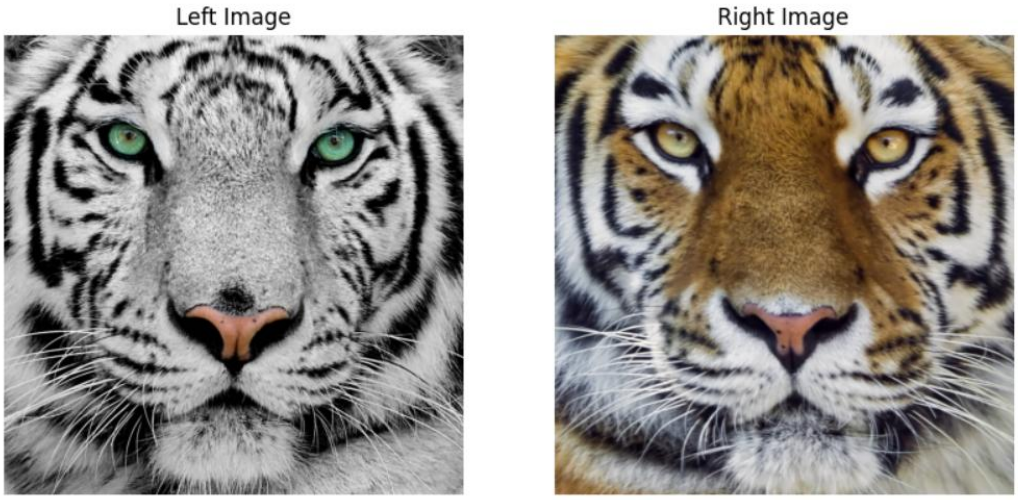
```
std::vector<cv::Mat> constructLaplacianPyramid(const cv::Mat& img)
{
   std::vector<cv::Mat> gaussianPyr = constructGaussianPyramid(img);
   std::vector<cv::Mat> laplacianPyr;

   for (size_t i = 0; i < gaussianPyr.size() - 1; i++)
   {
      cv::Mat expanded;
      cv::pyrUp(gaussianPyr[i + 1], expanded, gaussianPyr[i].size());
      cv::Mat diff = gaussianPyr[i] - expanded;
      laplacianPyr.push_back(diff);
   }
   laplacianPyr.push_back(gaussianPyr.back());
   return laplacianPyr;

}
```

Construct a Laplacian pyramid

```
std::vector<cv::Mat> constructLaplacianPyramid(const cv::Mat& img)
{
   std::vector<cv::Mat> gaussianPyr = constructGaussianPyramid(img);
   std::vector<cv::Mat> laplacianPyr;
   for (size_t i = 0; i < gaussianPyr.size() - 1; i++)
   {
      cv::Mat expanded;
      cv::pyrUp(gaussianPyr[i + 1], expanded, gaussianPyr[i].size());
      cv::Mat diff = gaussianPyr[i] - expanded;
      laplacianPyr.push_back(diff);
   }
   laplacianPyr.push_back(gaussianPyr.back());
   return laplacianPyr;
}
```



Reconstruct an image by collapsing a Laplacian pyramid

```
cv::Mat collapsePyramid(const std::vector<cv::Mat>& pyr)
{
   cv::Mat result = pyr.back();

   for (int i = pyr.size() - 2; i >= 0; i--)
   {
      cv::Mat expanded;
      cv::pyrUp(result, expanded, pyr[i].size());
      result = expanded + pyr[i];
   }

   return result;
}
```

Perform the Laplace blending

```
cv::Mat laplaceBlending(const cv::Mat& img_1, const cv::Mat& img_2, const cv::Mat& weights)
{
   // Construct a gaussian pyramid of the weight image.
   // TODO: Finish constructGaussianPyramid().
   std::vector<cv::Mat> weights_pyr = constructGaussianPyramid(weights);

   // Construct a laplacian pyramid of each of the images.
   // TODO: Finish constructLaplacianPyramid().
   std::vector<cv::Mat> img_1_pyr = constructLaplacianPyramid(img_1);
   std::vector<cv::Mat> img_2_pyr = constructLaplacianPyramid(img_2);

   // Blend the laplacian pyramids according to the corresponding weight pyramid.
   std::vector<cv::Mat> blend_pyr(img_1_pyr.size());
   for (size_t i = 0; i < img_1_pyr.size(); ++i)
   {
      // TODO: Blend the images using linearBlending().
      blend_pyr[i] = linearBlending(img_1_pyr[i], img_2_pyr[i], weights_pyr[i]);
   }

   // Collapse the blended laplacian pyramid.
   // TODO: Finish collapsePyramid().
   return collapsePyramid(blend_pyr);
}
```