# VFX project2:
# Image Stitching

**B01902037 楊孟遠 / B01902039 施楷文**

## Feature Detection

For the feature detection, we use harris corner detector. Following the slides and the paper, we can get the R value of each pixels by some computation on x, y derivatives of image and applying Gaussian filter.

After getting the R value of each pixels, we need to specify which of them are key-points. At first, we tried choose all points above some threshold as key-points. This works for the parrington dataset, whose photo sizes are relatively small. However, as the photo size becomes bigger, the number of key-points grows and the time of feature matching grows incredibly. In order to speed up the process, we choose one quarter of key-points with biggest R values in each image.

## Feature Matching

In order to establish the matchings between key-points in two images, we need to find robust descriptor of each key-point. Our first trial is the simplest method, the intensities of its spatial neighbors. However, the result of feature matching seems not very good. We then use the local image descriptor presented in SIFT. With the 128 dimension feature, the matchings are good enough for image alignment.

## Cylindrical Warping

We use the provided executable autostitch to get the focal length of images. By applying the following formula and set s = f, we can easily get the cylindrical warped images.

$$x' = s\theta = s\tan^{-1}\frac{x}{f}$$

$$y' = sh = s\frac{y}{\sqrt{x^2+f^2}}$$

## Outliers Detection

Now, we have the matching feature-pairs. First we are going to do is, filter out the outliers. To do this, we use the **RANSAC** algorithm with fitting model of the **distance of two features** in pair. After filtering out the outliers, we use an OPENCV built-in function to solve the linear equation.

Figure 1(a). Original feature matching

Figure 1(b).Feature matching after RANSAC

## Image Alignment & Blending

After the previous steps, we can start to blend the images. We stitch each warped image on to the panorama one after one. In each loop, we transform the new adding image on to the panorama's coordinate by using the transformation matrix we get in the alignment step . To avoid holes from happening, we use **inverse warping**.

In each loop, we have to find the minX, minY, maxX, maxY to set a bounding box to generate the new panorama, and then we put the new adding image on to the old panorama. On the overlaying part of two image, we use **linear blending** (the pixel that are nearer to the left image will have more weight of the left image)

Figure 2(a). Original panorama

Figure 2(b). panorama after blending

## Crop and Refinement

We didn't really implement the bundle adjustment, but we do fix the **drift** and **crop** the black side-edge of panorama. To fix the drift, we use the difference of the panorama's height and original image's height and the panorama's width to get the **drift slope**. And use **inverse warping** again to refine the panorama.

Figure 3(a). Original Panorama

Figure 3(b). Fix Drift

Figure 3(a). Fix Drift & Crop

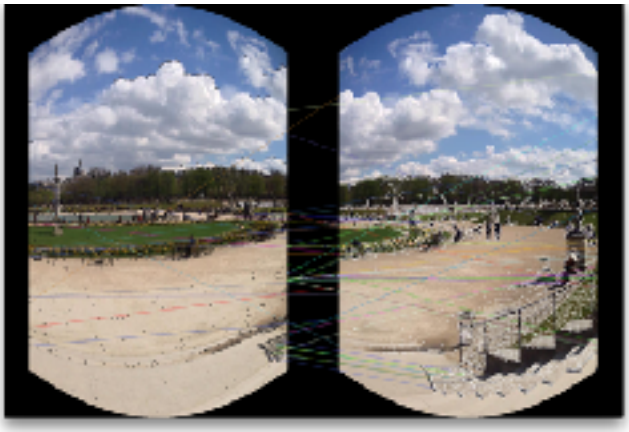
## Our Result

# Reviews and Finding

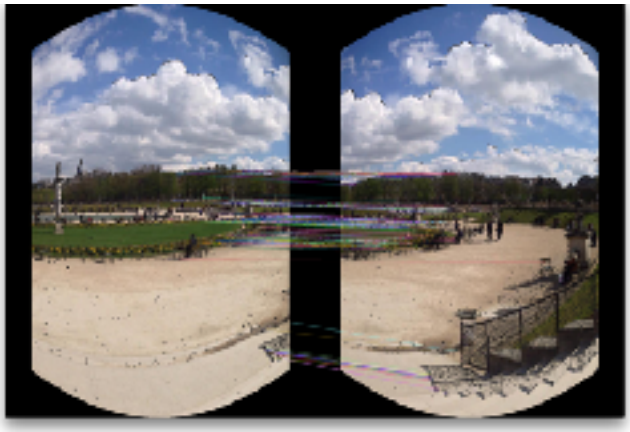
1. Feature matching before/after cylindrical warping :

We found that our result matches well in the origin images, while the matching is terrible in the warped image. The reason may because warping change the local information in different image. In order to get better matching, we match the key-points in the origin images, and then warped the key-points and matching with together with the image.

matchings in origin image



matchings after warping



matchings before warping

2. Use Inverse warping instead of forward warping:

Although we can use splattering method in forward warping to avoid holes, determining the kernel size could be difficult. Bigger kernel might blur the panorama badly. Using inverse warping with interpolate method could these situations become trivial.

3. Autostitch focal length problem

When we use the focal length generated from the Autostitch application, we found that there is a multiplication between the **f** value from the applicaiotn and the real **f** value, after we do an adjustment on the focal length value manually, the stitching result blame much better.