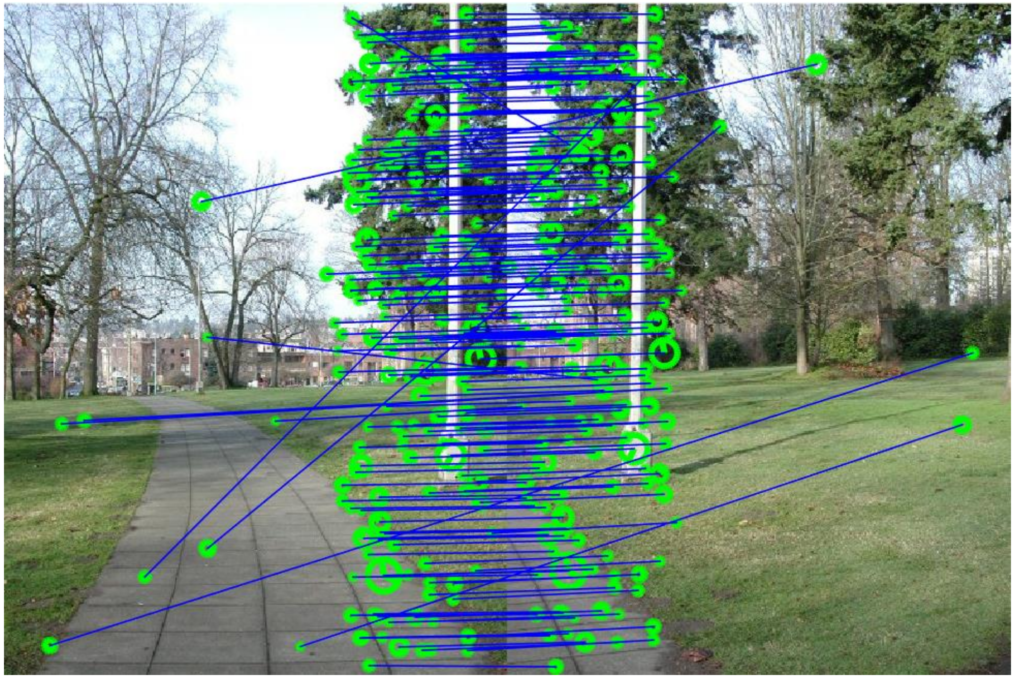
CSE185
Introduction to Computer Vision
Lab 09: Image Stitching

Instructor: Daniel Leung
TA: Mohammadkazem Ebrahimpour
Xueqing Deng

Image Stitching



4 Steps for Image Stitching

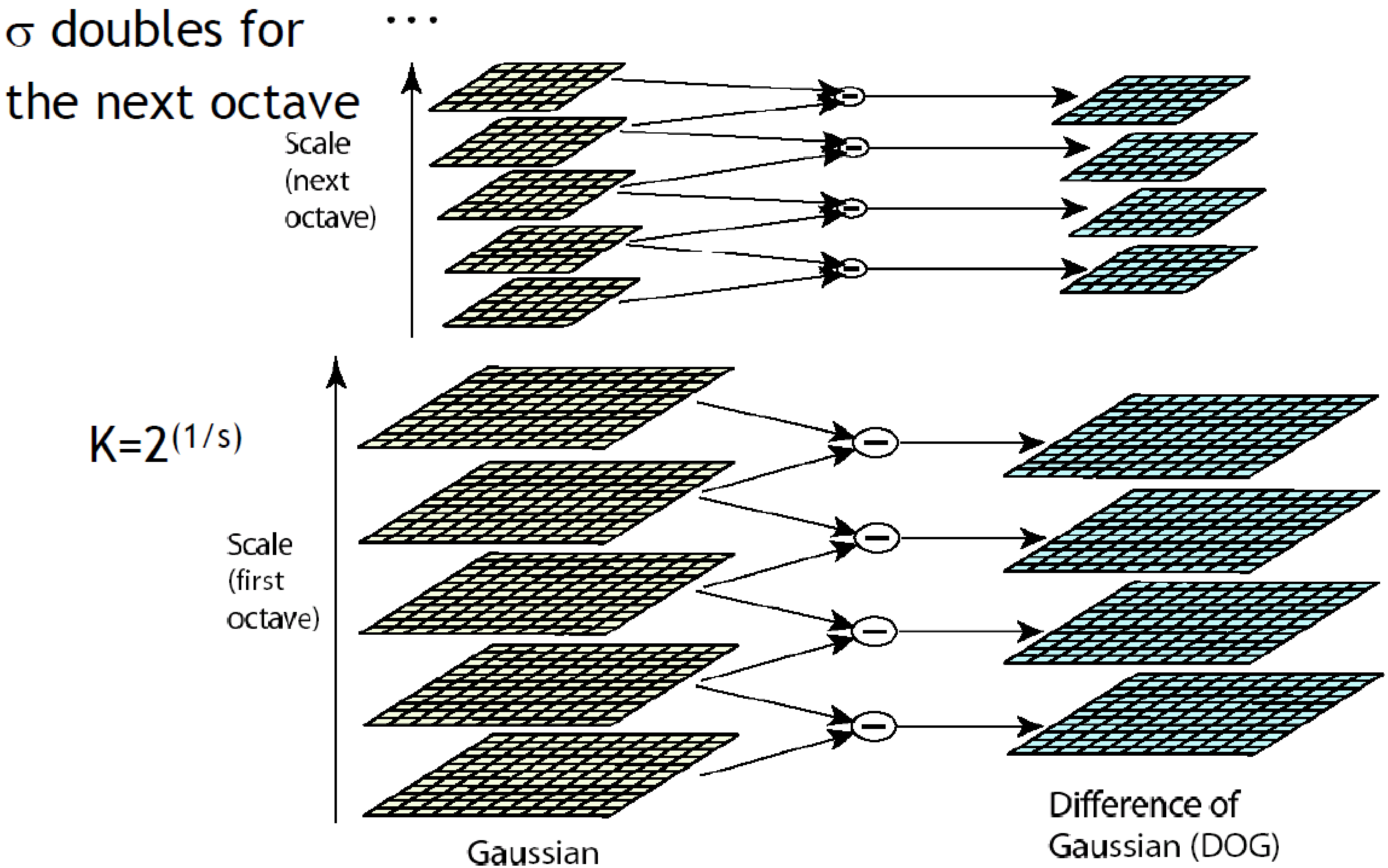
1. Feature Extraction (SIFT or Harris)
2. Feature Matching
3. Image Alignment with RANSAC
4. Image Blending/Stitching

4 Steps for Image Stitching

1. **Feature Extraction (SIFT or Harris)**
2. Feature Matching
3. Image Alignment with RANSAC
4. Image Blending/Stitching

SIFT Features Detector

- Use Difference-of-Gaussian to detect multi-scale features



SIFT Features Descriptor

- Use magnitude and orientation of gradients to describe feature points as 128-D feature vectors

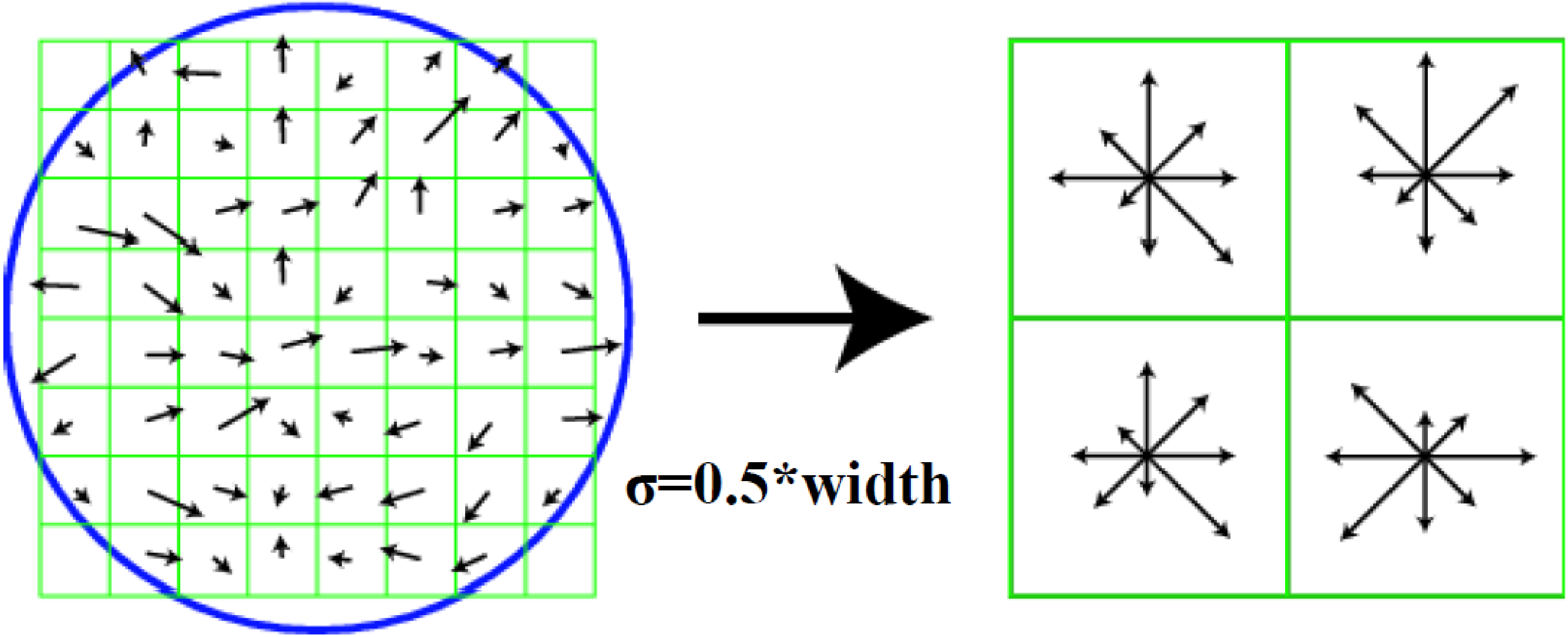


Image gradients

Keypoint descriptor

vlfeat

- Use vlfeat toolbox to extract SIFT features
  - The `vlfeat-0.9.20-bin` folder is already in the archived lab 9 file. After unzipping the lab 9 file, set MATLAB to work within the lab 9 folder.
  - Setup in MATLAB:

```
run('vlfeat-0.9.20-bin/toolbox/vl_setup');
```

- SIFT feature in vlfeat: http://www.vlfeat.org/overview/sift.html

Extract SIFT Features

1. Load image as single format
2. convert to gray scale
3. apply `vl_sift()`

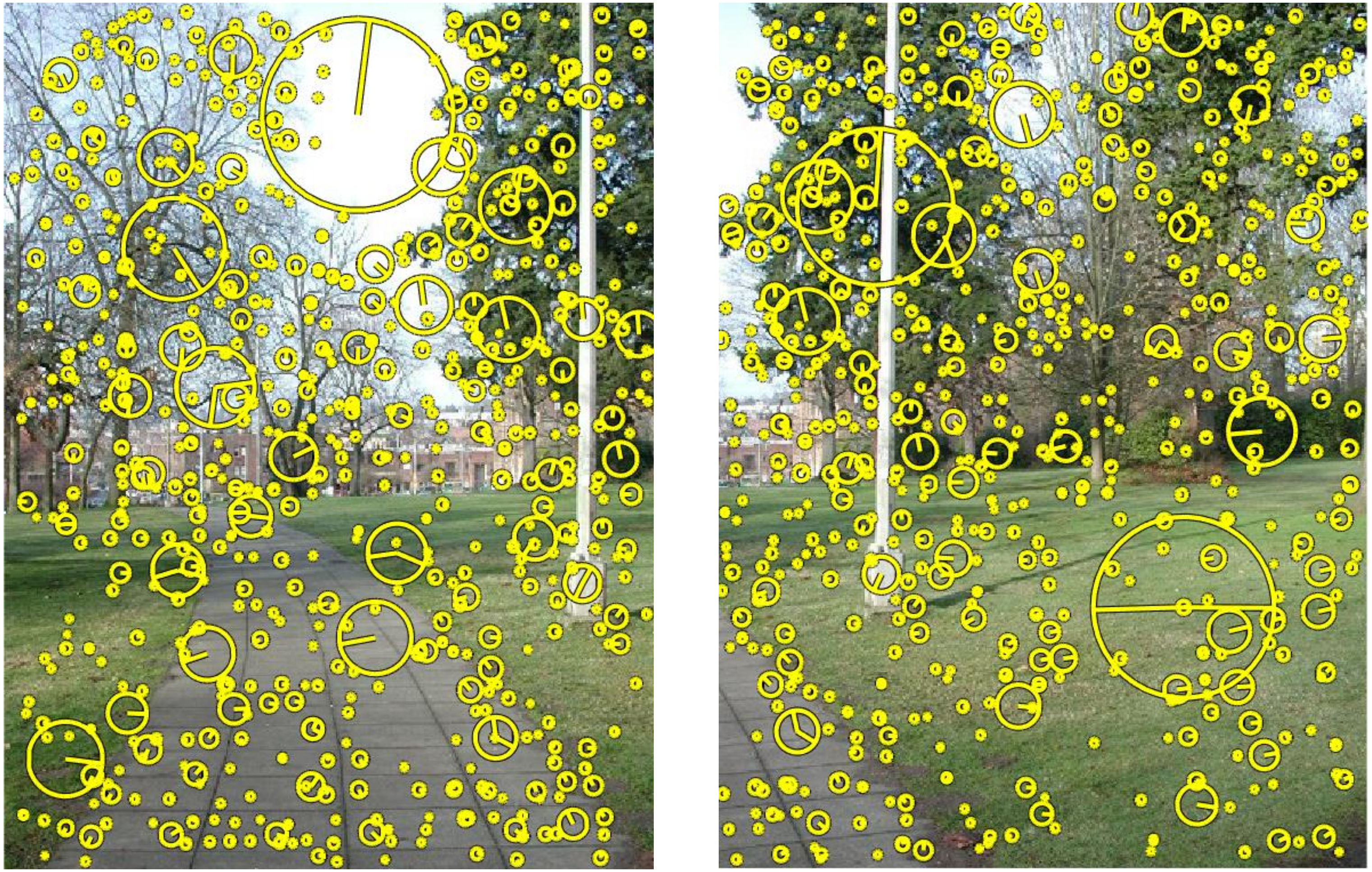
```
img1 = im2single(imread('prtn13.jpg'));
img2 = im2single(imread('prtn12.jpg'));

%% SIFT feature extraction
I1 = rgb2gray(img1);
I2 = rgb2gray(img2);

[f1, d1] = vl_sift(I1);
[f2, d2] = vl_sift(I2);

d1 = double(d1);
d2 = double(d2);
```

`f` is a $4 \times N$ feature frame, each column $= [x, y, scale, orientation]$

`d` is a $128 \times N$ descriptor, each column is an 128-D feature vector

Extract SIFT Features

- Draw feature points with `plot_sift(img, f, d)`

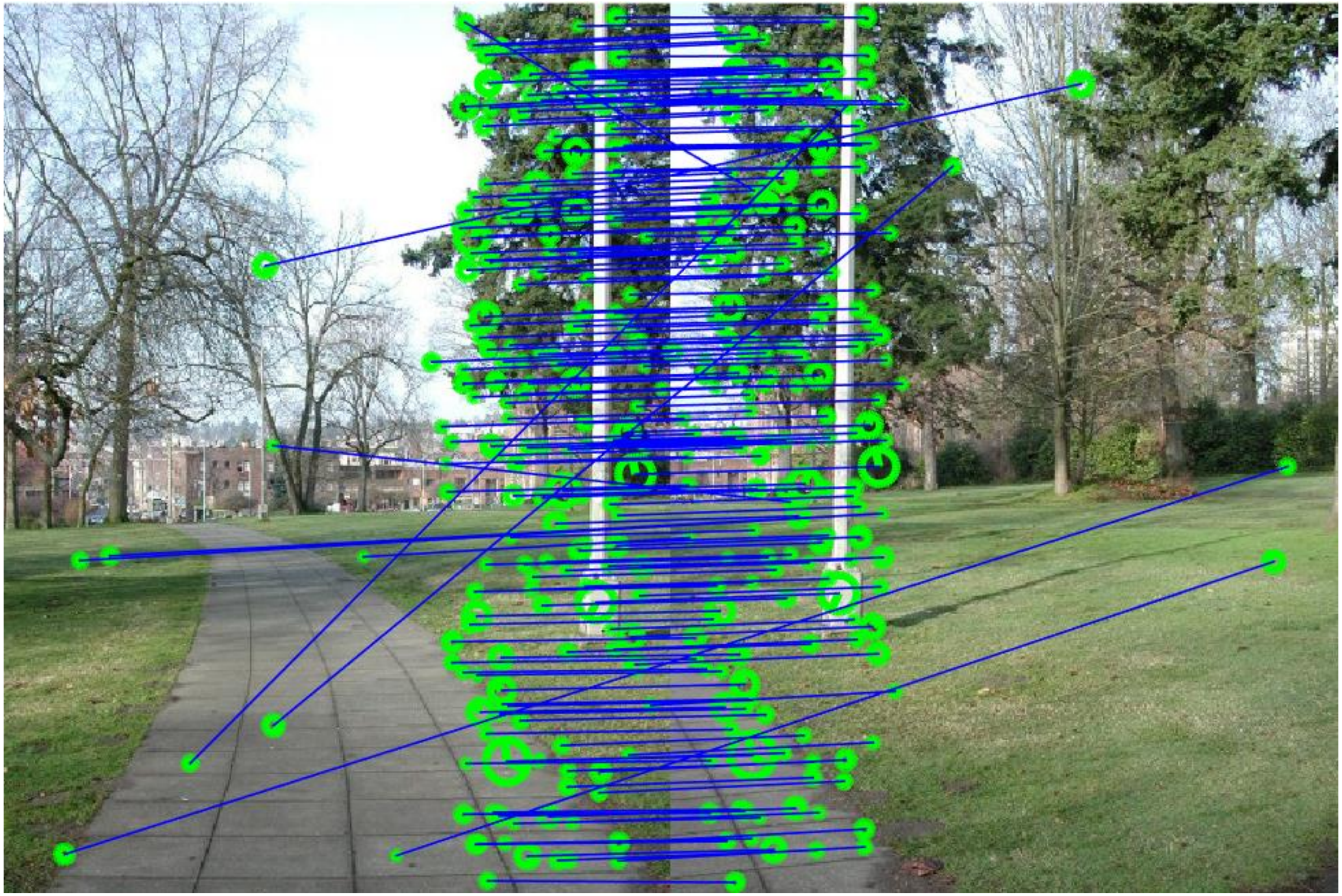


4 Steps for Image Stitching

1. Feature Extraction (SIFT or Harris)
2. **Feature Matching**
3. Image Alignment with RANSAC
4. Image Blending/Stitching

Feature Matching

- Find matched features between two images



Feature Matching

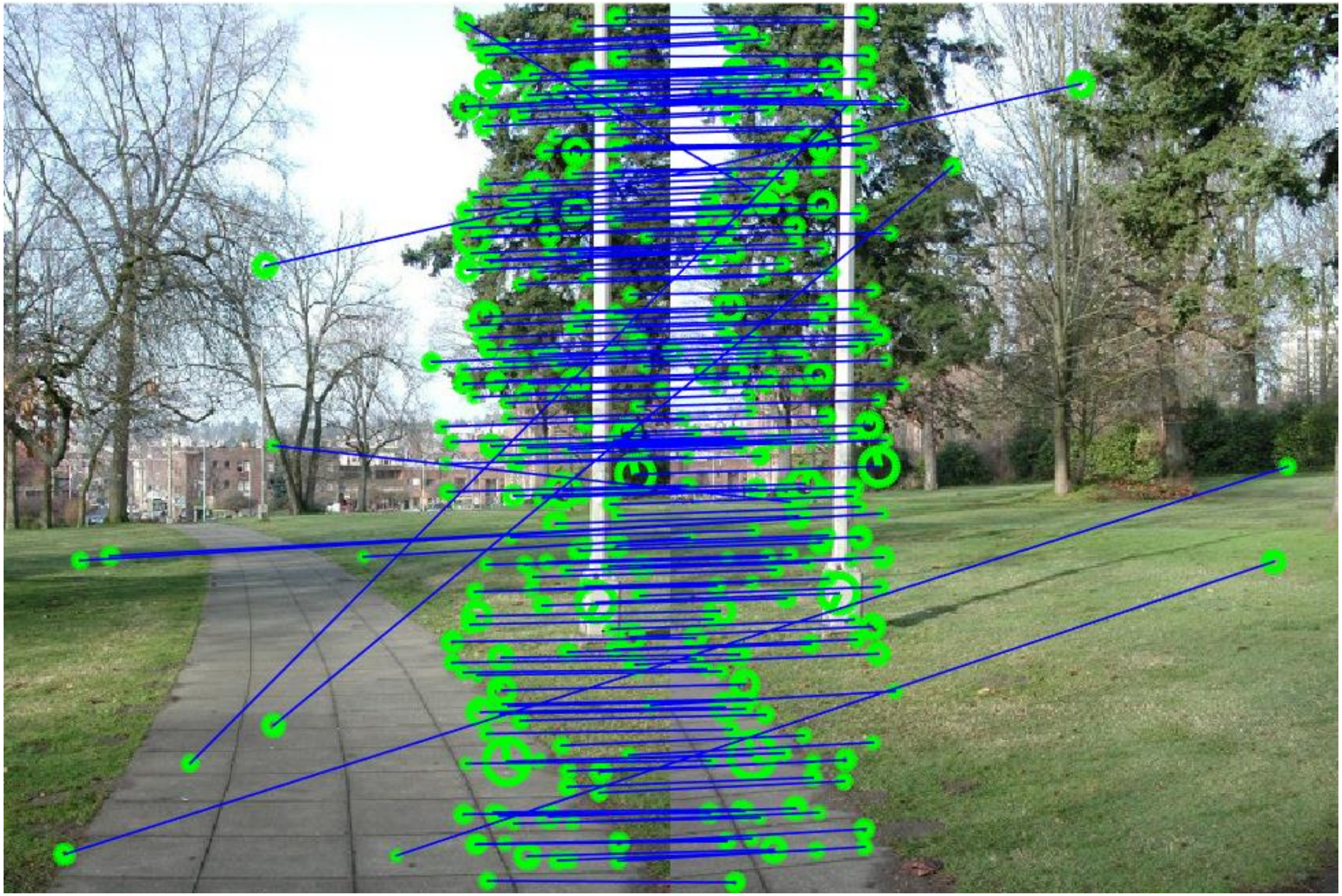
- Use `vl_ubcmatch()`

```
[matches, scores] = vl_ubcmatch(d1, d2) ;
```

- `matches` is a $2 \times N_{match}$ matrix, each column indicates the matched index of `d1` and `d2`
  - d1(:, matches(1, 1)) and d2(:, matches(2, 1)) are the first matched pair
  - d1(:, matches(1, k)) and d2(:, matches(2, k)) are the k-th matched pair
- Plot matched points with `plot_match(img1, img2, f1, f2, matches)`

Feature Matching

- `plot_match(img1, img2, f1, f2, matches)`



4 Steps for Image Stitching

1. Feature Extraction (SIFT or Harris)
2. Feature Matching
3. **Image Alignment with RANSAC**
4. Image Blending/Stitching

Image Alignment

- Assume simple translation model: $p1 = p2 + \begin{pmatrix} tx \\ ty \end{pmatrix}$

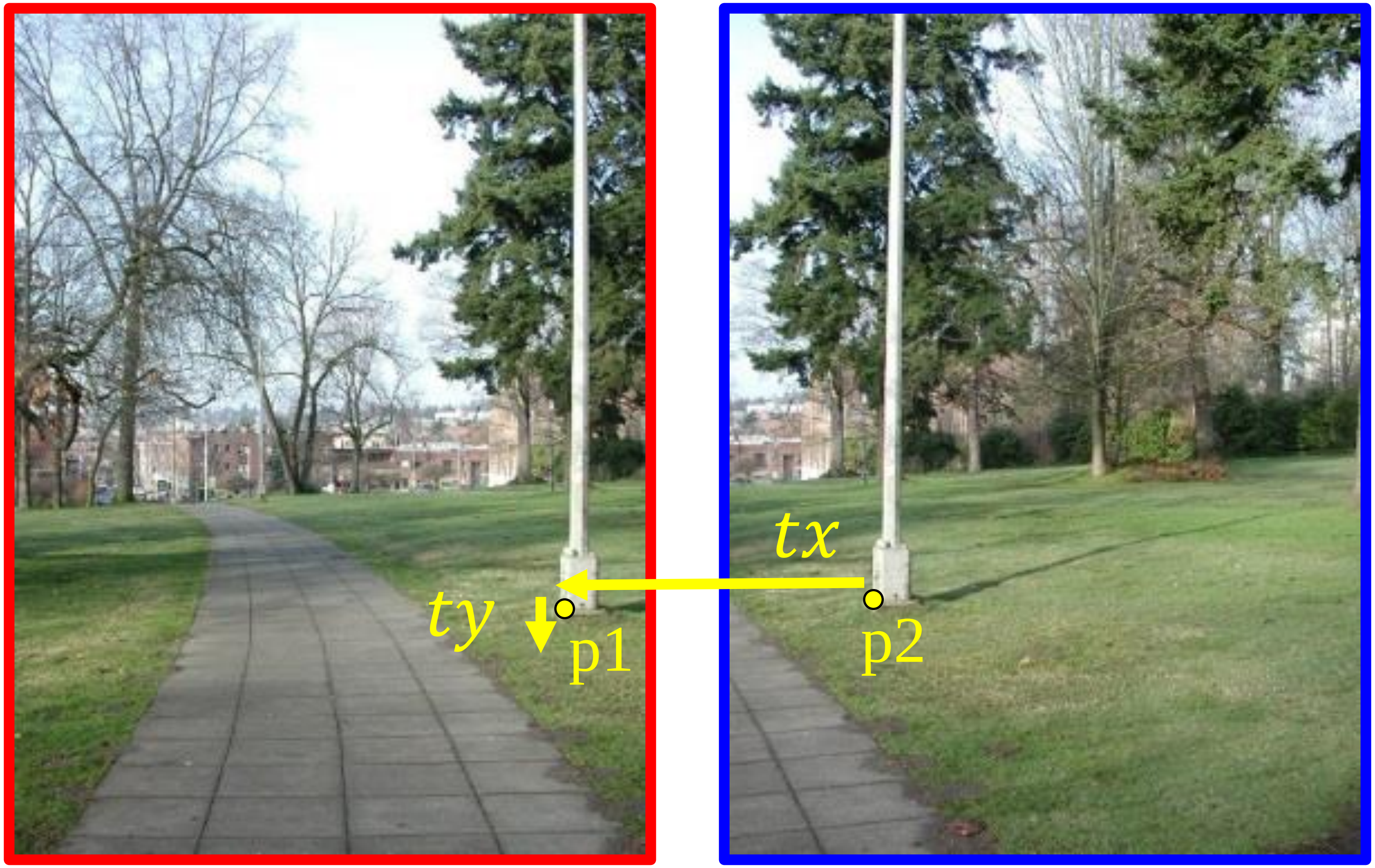


image 1 image 2

Image Alignment

- Assume simple translation model: $p1 = p2 + \begin{pmatrix} tx \\ ty \end{pmatrix}$

image 1 image 2

Image Alignment

- Each matching pair can determine a pair of (tx, ty)

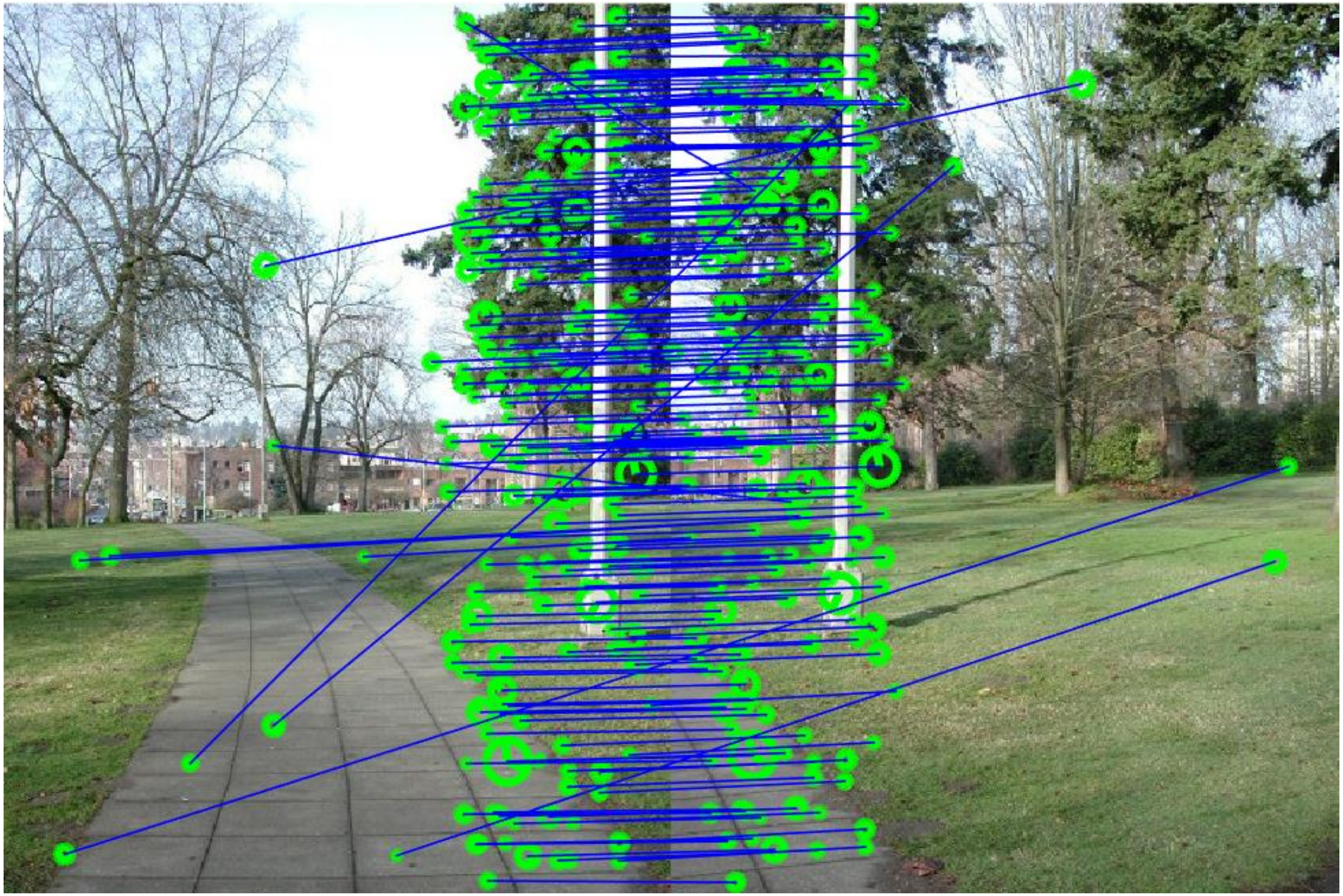


Image Alignment

- Use all matches pair to compute average (tx, ty)
  - outliers cause large error

Image Alignment

- Use RANSAC to select the best (tx, ty)

RANSAC

- Assume total *N* match pairs

```
Run k times:
    1. randomly choose 1 pair (2 feature points)
    2. compute tx_0 and ty_0
    3. for other N-1 pairs:
        (a) compute tx_1 and ty_1
        (b) if (tx_1, ty_1) is close to (tx_0, ty_0),
            count as an inlier
Return the pair (tx_0, ty_0) with the maximal #inliers
```

How to determine k?

- n: the number of selected points (in our case, $n = 2$)
- e: the proportion of outliers
- P: the probability of success rate after k trials

$$P = 1 - (1 - (1 - e)^n)^k$$

$$k = \frac{\log(1 - P)}{\log(1 - (1 - e)^n)}$$

- If $n = 2$, $e = 0.1$, $P = 0.999$, then $k \cong 4$

How to count inliers?

- Compute (tx, ty) from feature frame f

```
p1 = f1(1:2, matches(1, i));
p2 = f2(1:2, matches(2, i));

tx = p1(1) - p2(1);
ty = p1(2) - p2(2);
```

`f` is a $4 \times N$ feature frame, each column $= [x, y, scale, orientation]$

`matches` is $2 \times N_{match}$ matrix, each column $= [f1_{index}, f2_{index}]$

- A pair (tx_1, ty_1) is inlier of (tx_0, ty_0) if

$$(tx_1 - tx_0)^2 + (ty_1 - ty_0)^2 < \delta$$

RANSAC

- Assume total *N* match pairs

use randperm()

```
Run k times:
    1. randomly choose 1 pair (2 feature points)
    2. compute tx_0 and ty_0
    3. #inlier = 0
    4. for other N-1 pairs:
         (a) compute tx_1 and ty_1
         (b) if (tx_1-tx_0)^2 + (ty_1-ty_0)^2 < δ:
              #inlier = #inlier + 1

    if #inlier > max_#inlier:
         best_tx = tx_0
         best_ty = ty_0

Return (best_tx, best_ty)
```

4 Steps for Image Stitching

1. Feature Extraction (SIFT or Harris)
2. Feature Matching
3. Image Alignment with RANSAC
4. **Image Blending/Stitching**

Image Blending/Stitching

- Fix image 1, and shift image 2 by (tx, ty)

image 1 image 2

Image Blending/Stitching

- Simply paste image 2 over image 1

Image Blending/Stitching

- Fix image 1, and shift image 2 by (tx, ty)

```
output = zeros(H + ty, W + tx, 3);
output(1:H, 1:W, :) = img1;

for y2 = 1:size(img2, 1)
    for x2 = 1:size(img2, 2)

        y1 = y2 + ty;
        x1 = x2 + tx;

        if( y1 >= 1 && y1 <= H + ty &&
            x1 >= 1 && x1 <= W + tx )
            output(y1, x1, :) = img2(y2, x2, :);
        end

    end
end
```

TODO

- Implement lab09.m
  - use vlfeat for feature extraction (5pt) and feature matching (5pt)
  - implement RANSAC (5pt)

- Adjust e and P to see the difference on k and stitching results

- Try other image pairs (in denny.zip), upload at least 3 results and lab09.m (5pt)