Homework X Writeup

Instructions

- Describe any interesting decisions you made to write your algorithm.
- Show and discuss the results of your algorithm.
- Feel free to include code snippets, images, and equations.
- Use as many pages as you need, but err on the short side If you feel you only need to write a short amount to meet the brief, th
- **Please make this document anonymous.**

In the beginning...

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. See Equation **??**.

$$a = b + c \tag{1}$$

Interesting Implementation Detail

My code snippet highlights an interesting point.

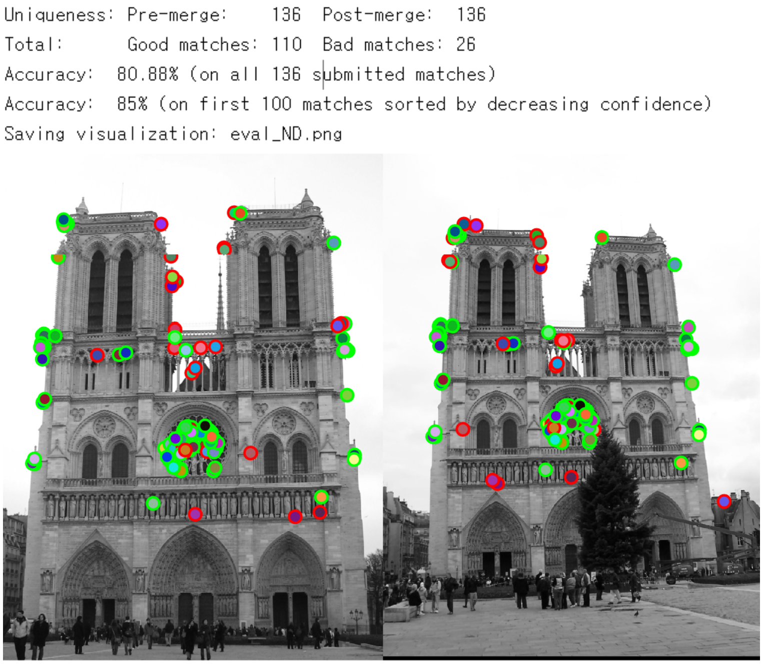
```
[~, ind] = sort(C_arr, 'descend');
x = x(ind,:);
y = y(ind,:);
ind = zeros(1,n);
visit = zeros(r,c);
w = descriptor_window_image_width/4;
for i=1:n
    if visit(y(i), x(i)) == 0
        ind(i) = 1;
        for j=y(i)-w:y(i)+w
            for k=x(i)-w:x(i)+w
                if dot([j, k] - [y(i), x(i)], [j, k] - [y
                    (i), x(i)]) <= w^2
                    visit(j,k) = 1;
                end
            end
        end
    end
end
ind = logical(ind);
x = x(ind,:);
y = y(ind,:);
```

This is about adaptive non maximum suppression part. Carr is array of C value of Harris corner detector. So x,y coord sort with descend order about C value. Then visit coordinate index of 1 to n(C is upper than threshold, point number, lst coordinate have largest C value), If the current point (x,y) I'm visiting is confirmed as feature, then check with visit(x,y) and nearby points (max distance : w) (* visit(x,y) means (x,y) is never be feature point)

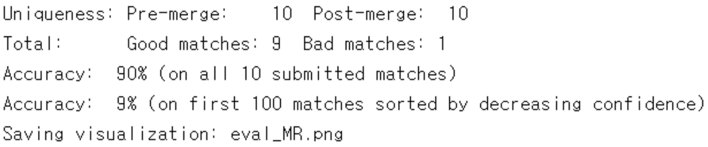
```
gauss_distance = fspecial('gaussian', 16, 4);

    grad_patch = grad(y(i)-w:y(i)+w-1, x(i)-w:x(i)+w-1);
    grad_patch = grad_patch .* gauss_distance;
    ori_patch = ori(y(i)-w:y(i)+w-1, x(i)-w:x(i)+w-1);
    ori_patch = grad_patch .* ori_patch;
    dominant_orient = sum(ori_patch(:))/sum(grad_patch(:));
    if dominant_orient == 360
        shift = -7;
    else
        shift = -floor(dominant_orient / 45);
    end


h = circshift(h,shift);
```

This is about Estimate feature orientation part code, considering magnitude of gradiant and gauss distance about center (y(i), x(i)) so calculate average orientation. And later calculate orientation histogram by 16 part by 1 feature point, standard about dominant orient, rotate using shift.

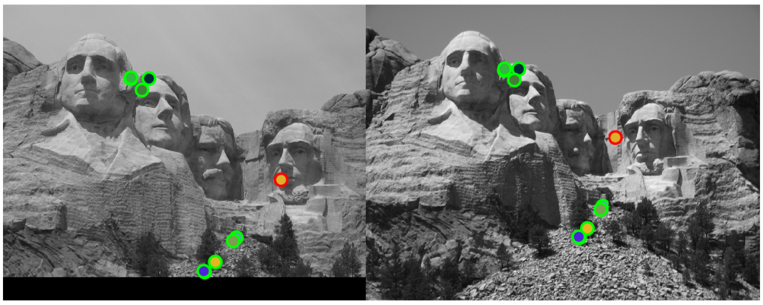
A Result

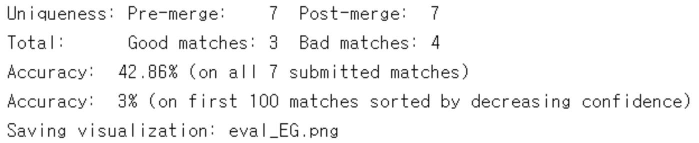


```
Uniqueness: Pre-merge:    136  Post-merge:  136
Total:      Good matches: 110  Bad matches: 26
Accuracy:  80.88% (on all 136 submitted matches)
Accuracy:  85% (on first 100 matches sorted by decreasing confidence)
Saving visualization: eval_ND.png
```



```
Uniqueness: Pre-merge:    10  Post-merge:  10
Total:      Good matches: 9  Bad matches: 1
Accuracy:  90% (on all 10 submitted matches)
Accuracy:  9% (on first 100 matches sorted by decreasing confidence)
Saving visualization: eval_MR.png
```





```
Uniqueness: Pre-merge:    7  Post-merge:  7
Total:      Good matches: 3  Bad matches: 4
Accuracy:  42.86% (on all 7 submitted matches)
Accuracy:  3% (on first 100 matches sorted by decreasing confidence)
Saving visualization: eval_EG.png
```