Visual homing using PCA-SIFT

Sandeep.R

Introduction

- This project is about performing visual homing using SIFT.
- For visual homing, we need to compare the image taken from the current location and pose, with reference images in a stored database.
- The image to be compared won't be exactly the same.
- By using SIFT to extract features, we can match images even with large changes in scale and rotation.

Visual homing

- In visual homing the goal is to move towards a home location or goal.

- This is done by obtaining an image of the surroundings at a particular location and pose and using the image to determine what action needs to be taken to move towards the home location.

- The image taken is compared to a set of reference images taken at different locations and poses and actions are associated with each reference image.

- A unique match is found and the action associated with the matched reference image is executed.

SIFT - Scale Invariant Feature Transform

- SIFT is used to extract image features that are invariant to scale, rotation and some changes in illumination.

- It is quite robust and can be used to identify the same features in different images even with large changes in scale and rotation.

How SIFT works

There are 6 steps in this process:

1. Constructing a scale space

2. Difference of Gaussian approximation

3. Finding keypoints

4. Get rid of bad kepypoints

5. Assigning an orientation to the keypoints

6. Generate SIFT features

Entire explanation copied or derived from :
http://www.aishack.in/2010/05/sift-scale-invariant-feature-transform/

Constructing a scale space

- Take the original image and apply Gaussian blur to it many times. Each consecutive image forms a scale.

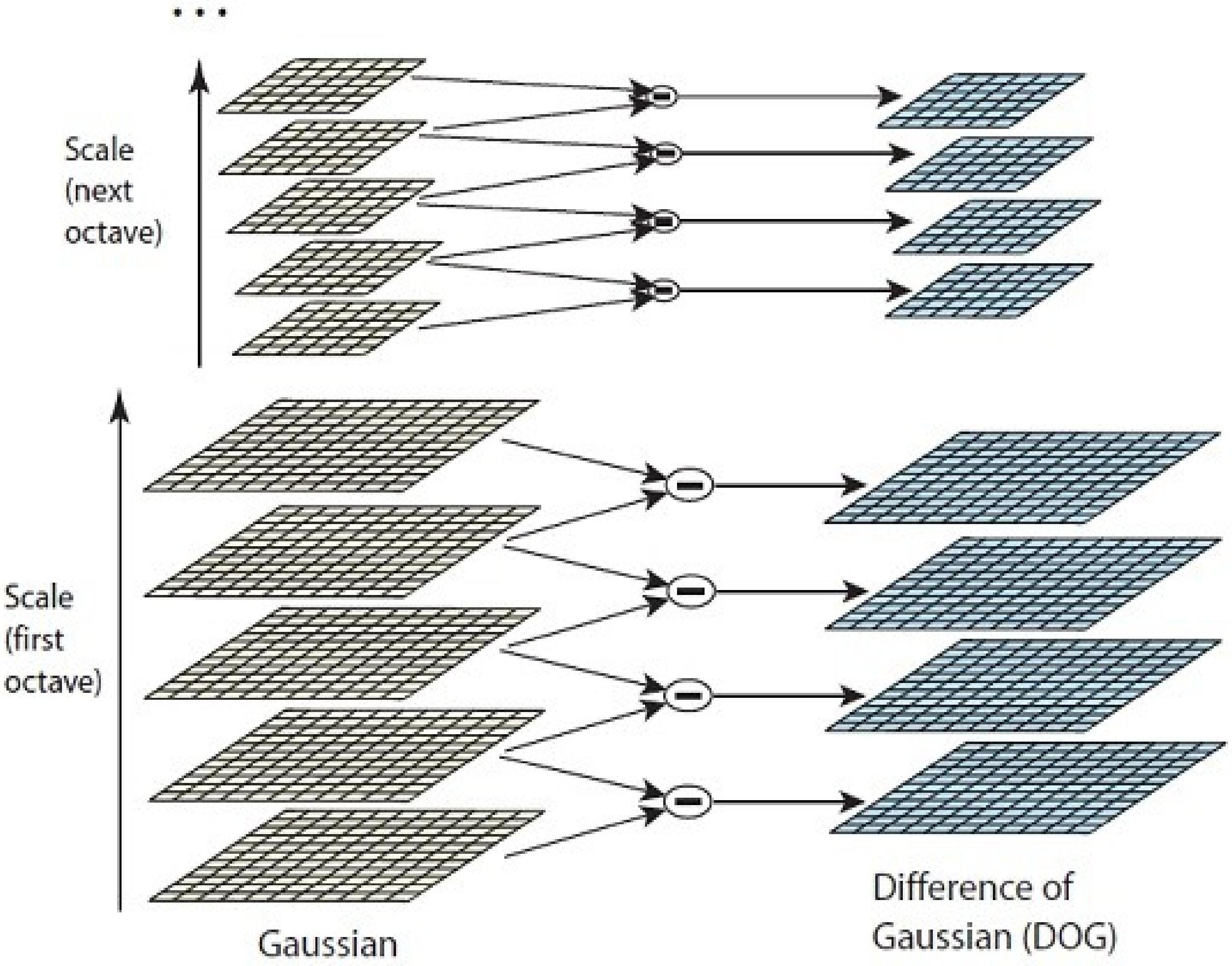
- This set of images forms the 1st octave.

- Next, scale down the image and do the same. This set forms the 2nd octave.

- And again upto a certain extent. The number of octaves can be arbitrary and is upto the person using SIFT.

Difference of Gaussians

We calculate the difference between two consecutive scales.

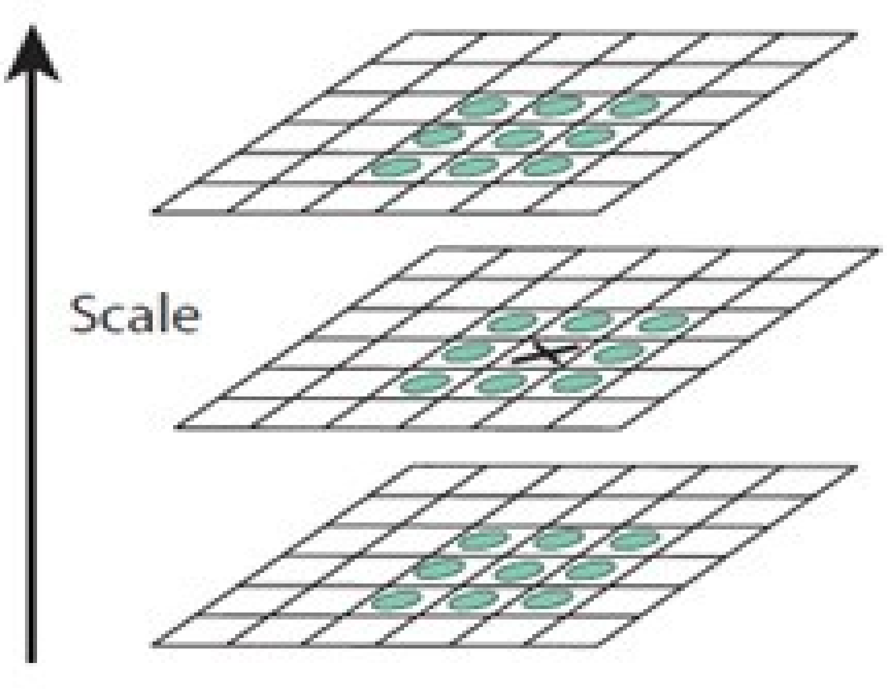


Source for image : http://www.aishack.in/2010/05/sift-step-2-laplacian-of-gaussian-approximation/

Finding keypoints

Find maxima and minima in each scale. Compare the pixel values with all of it's neighbours including the pixels in the scale images above and below it.

Imagine the scale images stacked together into a cuboid with each row being a scale image.



Source for image : http://www.aishack.in/2010/05/sift-step-3-finding-key-points/

Get rid of bad keypoints

Remove low contrast features - Check magnitude of intensity and discard if below a certain threshold.

Remove edges – Calculate gradients in two directions perpendicular to each other.

There are 3 cases :

Flat region : Gradients small in both directions.

Edge : Gradient is large in one direction and small in the other direction.

Corner : Gradient is large in both directions.

Assign orientation to keypoints

Around each keypoint, select an orientation collection region (square in shape) and determine the gradient directions and magnitudes for each pixel in this region.

Create a histogram of the gradient directions by dividing the orientation into bins.
(e.g) 8 bins of 45 degrees each

For each gradient add an amount proportional to the gradient magnitude to the bin to which the gradient direction belongs.

Assign the most prominent orientation to the keypoints.

Generate SIFT features

Take a 16x16 window of pixels around the keypoint. Divide it into sixteen 4x4 windows.

For each 4x4 window, determine gradient magnitude and direction for each pixel and store them in a 8 bin histogram. (e.g) 0-44 deg in the 1st bin, 45-89 deg in the 2nd bin etc.

Amount added to each bin also depends on distance from the keypoint

Thus you end up with 4*4*8 = 128 numbers which form the feature vector for that feature.

Visual homing using SIFT

My work is based on the work in this paper:
“Vision-based robot homing in dynamic environments” by
J. Saez Pons, W. Hübner, H. Dahmen and H. A. Mallot

The authors attempt to use SIFT to create reference images /patterns that are not affected by occlusions and changes in illumination.

At each landmark location, a number of images with occlusions and changes in illumination are taken.

Each image is analysed and SIFT features are extracted.

Visual homing using SIFT

This is done for each image at that reference location.

The same features are matched in different images based on the Euclidean distance between 2 features:

$sim(q,r) = e^{-0.5(q-r)^2}$

A count is kept for each feature and this count is incremented each time a feature match is found.

Thus we can determine which features are robust and invariant to occlusion and changes in illumination.

Visual homing using SIFT

Then, for homing when an image is taken at a particular location, SIFT features are extracted from that image.

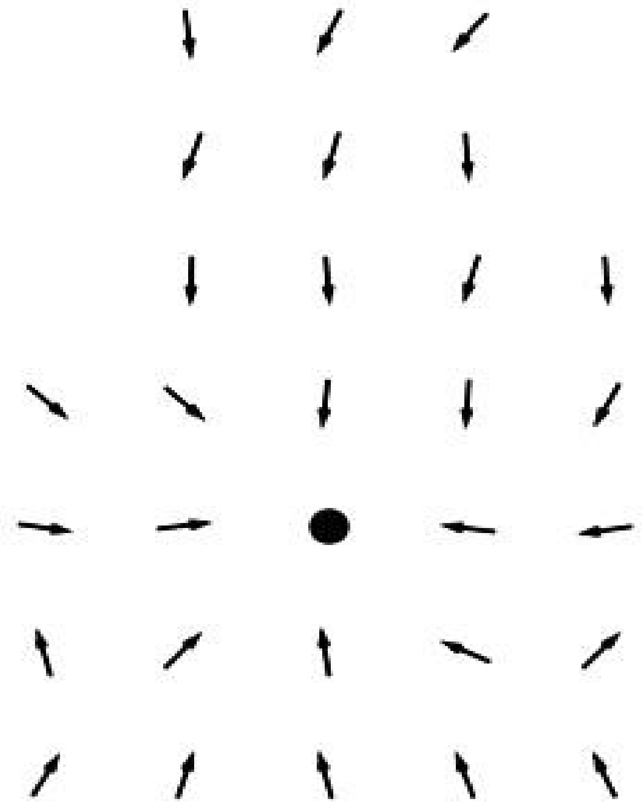
These features are compared with the features visible from the home location and the similarity and angular displacement are determined.

A histogram of angular displacement values is created and used to determine which is the most likely change in orientation

This is used to determine the direction towards the home location

Visual homing using SIFT

Thus, the direction towards the home location can be determined.



Use PCA-SIFT instead of SIFT

I propose using PCA-SIFT to represent features instead of plain SIFT.

“PCA-SIFT is more robust to deformation and produces a more compact representation than SIFT.” *

It also increases accuracy and speed of matching.

I will be using the “arboreal” image dataset from this website:
http://www.ti.uni-bielefeld.de/html/research/avardy/#andrew

* http://www.cs.cmu.edu/~yke/pcasift/

Goals

Minimal:

1) Implement a program to get PCA SIFT features and test them on the images from the dataset

2) Observe accuracy and speed of feature matching and how It affects the performance of visual homing

Additional goals:

1) Determine optimal threshold value for the images in the dataset (threshold for ratio of euclidean distance from current feature to nearest match and second nearest match respectively)

2) Use another metric other than Euclidean to determine similarity between features

References

http://www.aishack.in/2010/05/sift-scale-invariant-feature-transform/

Vision-based robot homing in dynamic environments by J. Saez Pons, W. Hübner, H. Dahmen and H. A. Mallot (August 2007)

http://www.cs.cmu.edu/~yke/pcasift/

The End