ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 2, Issue 2, February 2013

Plant Recognition through Leaf Features Mapping

Ms. Pushpinder Kaur, Ms. Janhavi Mane, and Ms Sonali Sawant

Abstract— **There are thousands of plants around us and they are useful in many ways, but in day today life we are so busy that we can't even recognize few important ones e.g. everyone loves mango, lemon is a well known fruits but many of us can't identify a simple mango and lemon tree. Thus this project recognizes for a curious person who wants to know the plant for general purpose or may be study purpose. This system is platform independent and design with open source tools and software. In this system we are using outer boundary shapes and shapes of ventricles for plant recognition. For this we are using SIFT algorithm.**

I. INTRODUCTION

Image processing and pattern matching are the areas in which a wide research work has been done in past few years. This research provides wide scope for application development in image processing. There are two types of queries in this type of system, 1. Give similar images as a result 2. Authenticate the given image from the existing database. In this project we are using the first type of query. This application will give information about the plant in image to user to minimize the error and further a chance to choose from given images and once confirmed it will give the information about the plant.

II. Aim and Objective

There are a lot of plants in our surroundings which may possess some unique properties in terms of general, study related and medicinal qualities. Since our life has become so fast and busy that we lack in getting the information about these trees and plants, hence we unaware of the benefits of these co-existents. Hence plant recognition application can be useful in number of situations and to number of people.

Objective includes

1. *Specifying keypoints*

This will take the given image through keypoint detection algorithm which in turn gives specific keypoints related to the shape of the image.

2. *Matching keypoints*

The keypoints will be matched with the already stored keypoints in the database.

3. *Retrieving information*

When a match is found the information related to the that particular plant can be given as an output.

III. Existing System

Whenever we want to know which plant it is then we need to perform searching it in the huge books thus wasting lot of time. We sometimes takes photo of that plant and then matches that image with the images present in books. This process involves considerable wastage of time as it is not always possible. The process requires human interaction at each state and takes a lot of time. Briefly we can summarise the drawback as:

1. *Memorize the plants*

If we want to know which plant it is then we have to memorize the different plants.

2. *Less Efficient*

The existing system is less efficient as it may not give information about each and every plant as the books are static in nature and having limited data.

3. *Time Consuming*

As each and every time we have to match the images taken by us with the available images in books so this requires more time.

IV. Problem Statement

User gives input in the form of binary image, the image is of the leaf of plant which we want to know, then SIFT algorithm is applied on that binary image, so we get the edges of that leaf. The edges we get by applying SIFT algorithm are matched with our database saved. Finally the client will be getting information saved in database if it matched with existing one.

V. Implementation

Here we are using eclipse for writing C++ source code for designing interface and creating various processes included in application.

A. Specifying keypoints

1. Constructing a scale space-
This is the initial preparation. You create internal

representations of the original image to ensure scale invariance. This is done by generating a "scale space". SIFT takes scale spaces to the next level. You take the original image, and generate progressively blurred out images. Then, you resize the original image to half size. And you generate blurred out images again. And you keep repeating.

2. .Log Approximation-

The Laplacian of Gaussian is great for finding interesting points (or key points) in an image. But it's computationally expensive. So we cheat and approximate it using the representation created earlier. The Laplacian of Gaussian (LoG) operation goes like this. You take an image, and blur it a little. And then, you calculate second order derivatives on it (or, the "laplacian"). This locates edges and corners on the image. These edges and corners are good for finding keypoints.But the second order derivative is extremely sensitive to noise. The blur smoothes it out the noise and stabilizes the second order derivative.

3. Finding keypoints-

With the super-fast approximation, we now try to find key points. These are maxima and minima in the Difference of Gaussian image we calculate in step 2.

The first step is to coarsely locate the maxima and minima. This is simple. You iterate through each pixel and check all it's neighbours. The check is done within the current image, and also the one above and below it.

Once this is done, the marked points are the approximate maxima and minima. They are "approximate" because the maxima/minima almost never lies exactly on a pixel. It lies somewhere between the pixel. But we simply cannot access data "between" pixels. So, we must mathematically locate the subpixel location.

$$m(x,y)=\sqrt{(L(x+1,y,\sigma)-L(x-1,y,\sigma))^2+(L(x,y+1,\sigma)-L(x,y-1,\sigma))^2}$$

$$\theta(x,y)=\arctan\left((L(x,y+1,\sigma)-L(x,y-1,\sigma))\div\left(L(x+1,y,\sigma)-L(x-1,y,\sigma)\right)\right)$$

4.*Get rid of bad keypoints-*

Edges and low contrast regions are bad keypoints. Eliminating these makes the algorithm efficient and robust. A technique similar to the Harris Corner Detector is used here. If the magnitude of the intensity (i.e., without sign) at the current pixel in the DoG image (that is being checked for minima/maxima) is less than a certain value, it is rejected.

Because we have subpixel keypoints (we used the Taylor expansion to refine keypoints), we again need to use the taylor expansion to get the intensity value at subpixel locations. If it's magnitude is less than a certain value, we reject the keypoint.

5.*Assigning an orientation to the keypoints-*

An orientation is calculated for each key point. Any further calculations are done relative to this orientation. This effectively cancels out the effect of orientation, making it rotation invariant. The idea is to collect gradient directions and magnitudes around each keypoint. Then we figure out the most prominent orientation(s) in that region. And we assign this orientation(s) to the keypoint.

6.*Generate SIFT features-*

Finally, with scale and rotation invariance in place, one more representation is generated. This helps uniquely identify features. Let's say you have 50,000 features. With this representation, you can easily identify the feature you're looking for (say, a particular eye, or a sign board). We want to generate a very unique fingerprint for the keypoint. It should be easy to calculate. We also want it to be relatively lenient when it is being compared against other keypoints. Things are never EXACTLY same when comparing two different images.

B. Matching keypoints

It consists of storing SIFT keys and identifying matching keys from the new image. The Best –Bin First algorithm uses a modified search ordering for the k-d tree algorithm so that bins in feature space are searched in the order of their closest distance from the query location. The best candidate match for each keypoint is found by identifying its nearest neighbor in the database of keypoints from training images. The probability that a match is correct can be determined by taking the ratio of distance from the closest neighbor to the distance of the second closest.

C. Retrieving information

After successful matching with the database stored in the system, the output will be given by the system in form of written text that we have provided in the database.

The block diagram of the whole system shows in the fig VI.

VI. FUNCTIONAL FLOW

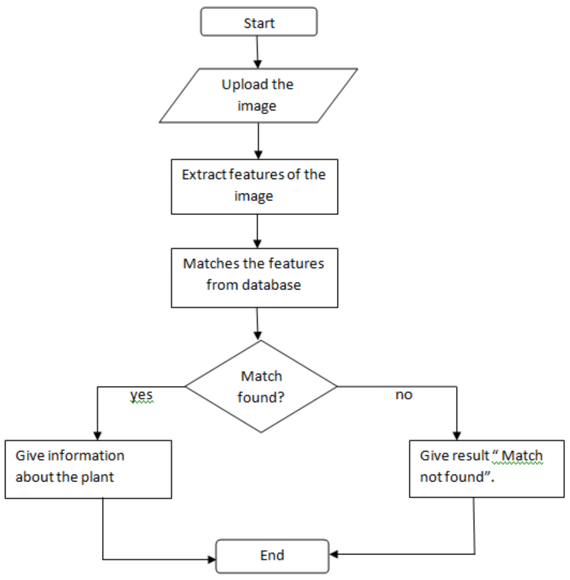
Start
Upload the image
Extract features of the image
Matches the features from database
Match found?
yes
no
Give information about the plant
Give result " Match not found".
End

Fig.1. Block diagram

VII. DETAILS OF SOFTWARE AND HARDWARE.

1. *Software Requirements*

- The software will be platform in dependent
- Frontend : OpenCV2.4.3, Eclipse4.2
- Backend : MySQL
- Operating System:-Windows

ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 2, Issue 2, February 2013

2. *Hardware Requirements*

- Digital camera- 5Mpixels (min)
- Processor- 1GHz(min) (any processor of similar capacity as Intel dual core and onwards)
- RAM – min 1GB DDR3.
- Printer (if needed)

VIII. CONCLUSION

We can identify the plant from its given image as input. The system gives full information about that plant. Also the user can print the information about the given plant if needed.

REFERENCES

[1] Hanife Kebapci, Berrin Yanikoglu and Gozde Unal: Plant Image Retrieval Using Color, Shape and Texture Features

[2] Faraj Alhwarin, Chao Wang, Danijela Risti -Durrant, Axel Gräser :Improved SIFT-Features Matching for Object Recognition

[3]David G. Lowe, International Journal of Computer Vision, :SIFT - The Scale Invariant Feature Transform

[4]Digital Image Processing Part-1: Huiyu Zhou, Jiahua Wu

[5]Digital Image Processing Part-2: Huiyu Zhou, Jiahua Wu

[6]Faraj Alhwarin, Chao Wang, Danijela Risti -Durrant, Axel Gräser: Improved SIFT-Features Matching for Object Recognition

AUTHOR'S BIOGRAPHIES

Ms. Pushpinder Kaur is Under Graduate student of Computer Engineering in RMCET, Ambav (Devrukh), Ratnagiri, University of Mumbai, India. She is a member of ISTE.

Ms Janhavi Mane is Graduate student of Computer Engineering in RMCET, Ambav (Devrukh), Ratnagiri, University of Mumbai, India. She is a member of ISTE.

Ms. Sonali Sawant is Graduate student of Computer Engineering in RMCET, Ambav (Devrukh), Ratnagiri, University of Mumbai, India. She is a member of ISTE.