# Implementation of Object Recognition Algorithm on ARM Using Embedded Linux

K. Sravanthi,

MTECH(Embedded Systems)

Aurora's technological and research institute,uppal,hyd.

Abdul Rahim,

Assoc.Professor,ECE

Aurora's technological and research institute,uppal,hyd.

## ABSTRACT

**Optical object detection and pose estimation are very challenging tasks in automobiles since they have to deal with problems such as different views of an object, various light conditions, surface reflections, and noise caused by image sensors. Presently available algorithms such as SIFT can to some extent solve these problems as they compute so-called point features, which are invariant towards scaling and rotation. However, these algorithms are computationally complex and require powerful hardware in order to operate in real time. In automotive applications and generally in the field of mobile devices, limited processing power and the demand for low battery power consumption play an important role. Hence, adopting those sophisticated point feature algorithms to mobile hardware is an ambitious, but also necessary computer engineering task. I am going to port this sift on hardware.**

**The SIFT algorithm (Scale Invariant Feature Transform)is an approach for extracting distinctive invariant features from images. It has been successfully applied to a variety of computer vision problems based on feature matching including object recognition, pose estimation, image retrieval and many others. However, in real-world applications there is still a need for improvement of the algorithm's robustness with respect to the correct matching of SIFT features. In this work, I propose to use original SIFT algorithm to provide more reliable feature matching for the purpose of object recognition.**

**This algorithm will be implemented on ARM processor for portable device applications.**

**Keywords: SIFT algorithm, Improved SIFT, Images matching, Object recognition**

IJERT

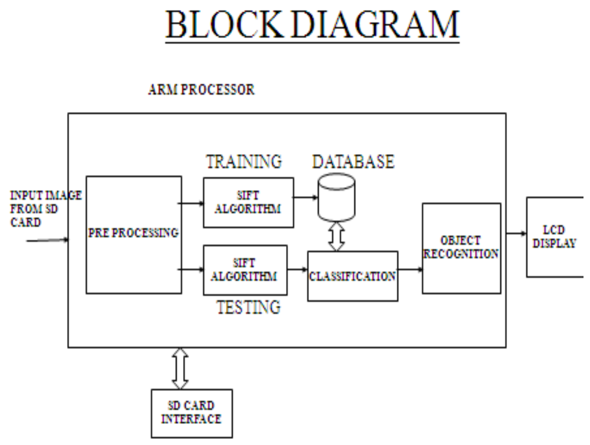
## I. INTRODUCTION

The existing object recognition algorithms can be classified into two categories: global and local features algorithms. Global features based algorithms aim to achieve this, after the acquisition, the test object is sequentially pre-processed and segmented. Then, the global features are extracted and finally statistical features classification techniques are used. This class of algorithm is particularly suitable for recognition of homogeneous (textureless) objects, which can be easily segmented from the image background. In contrast to this, local features based algorithms are more suitable for textured objects and are more robust with respect to variations in pose and illumination.

Local features based algorithms focus mainly on the so-called keypoints. In this context, the general scheme for object recognition usually involves three important stages: The first one is the extraction of salient feature points (for example corners) from both test and model object images. The second stage is the construction of regions around the salient points using mechanisms that aim to keep the regions characteristics insensitive to viewpoint and illumination changes. The final stage is the matching between test and model images based on extracted features.

The development of image matching by using a set of local keypoints can be traced back to the work of Moravec [8]. He defined the concept of "points of interest" as being distinct regions in images that can be used to find matching regions in consecutive image frames. The Moravec operator was further developed by C. Harris and M. Stephens who made it more repeatable under small image variations and near edges. Schmid and Mohr used Harris corners to show that invariant local features matching could be extended to the general image recognition problem. They used a rotationally invariant descriptor for the local image regions in order to allow feature matching under arbitrary orientation variations. Although it is rotational invariant, the Harris corner detector is however very sensitive to changes in image scale so it does not provide a good basis for matching images of different sizes. Lowe overcome such problems by detecting the points of interest over the image and its scales through the location of the local extrema in a pyramidal Difference of Gaussians (DOG). The Lowe's descriptor, which is based on selecting stable features in the scale space, is named the Scale Invariant Feature Transform (SIFT).

.



IJERT

## II. SIFT ALGORITHM:

The scale invariant feature transform (SIFT) algorithm, developed by David G.Lowe is an algorithm for image features generation which are invariant to image translation, scaling,rotation and partially invariant to illumination changes and affine projection. Calculation of SIFT image features is performed through the four consecutive steps which are briefly described in
the following:

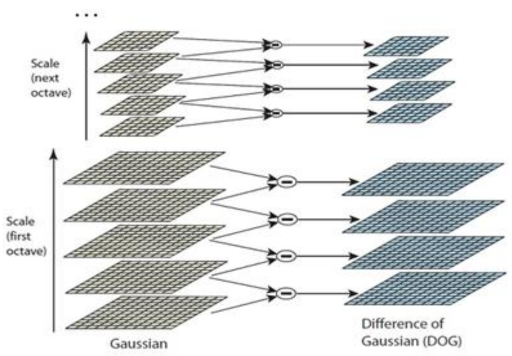
- **scale-space local extrema detection -**

Generate several octaves of the original image. Each octave's image size is half the previous one.The number of octaves and scale depends on the size of the original image. Each collection of images of the same size is called an octave .We have to decide how many octaves and scales are required.The creator of SIFT suggests that 4 octaves and 5 blur levels are ideal for the algorithm.

**First octave:** The original image is doubled in size and antialiased a bit (by blurring it) then the algorithm produces more four times more keypoints. The more the keypoints, the better!All octaves build together the so-called Gaussian pyramid

**BLURRING:** Mathematically, "blurring" is referred to as the convolution of the gaussian operator and the image.Gaussian blur has a particular expression that is applied to each pixel.

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

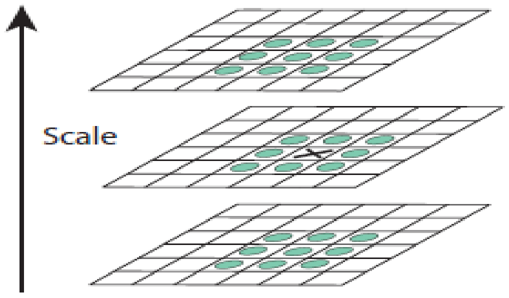


$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$$

- **keypoint localization** –

These keypoints are maxima and minima in the Difference of Gaussian image we calculate in LoG approximation. Finding key points is a two part process a) Locate maxima/minima in DoG images, b) Find subpixel maxima/minima.

**Locate maxima/minima in DoG images**: The first step is to coarsely locate the maxima and minima. We iterate through each pixel and check all its neighbour:



X marks the current pixel. The green circles mark the neighbours. This way, a total of 26 checks are made. X is marked as a "key point" if it is the greatest or least of all 26 neighbours. Usually, a non-maxima or non-minima position won‟t have to go through all 26 checks. A few initial checks will usually sufficient to discard it. Note that keypoints are not detected in the lowermost and topmost scales. There simply aren‟t enough neighbours to do the comparison. 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.

**Find subpixel maxima/minima:**

Using the available pixel data, subpixel values are generated. This is done by the Taylor expansion of the image around the approximate key point. Mathematically, it‟s like this:

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x}$$

We can easily find the extreme points of this equation (differentiate and equate to zero). On solving, we‟ll get sub pixel key point locations. These sub pixel values increase chances of matching and stability of the algorithm.

**Get rid of Bad Key points:**

Edges and low contrast regions are bad key points. Eliminating these makes the algorithm efficient and robust. Key points generated in the previous step produce a lot of key points. Some of them lie along an edge, or they don‟t have enough contrast. In both cases, they are not useful as features. So we get rid of them. For

IJERT

low contrast features, we simply check their intensities. 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 key points (we used the Taylor expansion to refine key points), 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 key point. Removing edges is applying by calculate two gradients at the key point. Both perpendicular to each other. Based on the image around the key point, three possibilities exist.

- **orientation assignment –**
  The idea is to collect gradient directions and magnitudes around each keypoint. The magnitude and orientation is calculated for all pixels around the keypoint. Then, A histogram is created for this.Gradient magnitudes and orientations are calculated using these formulae:

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

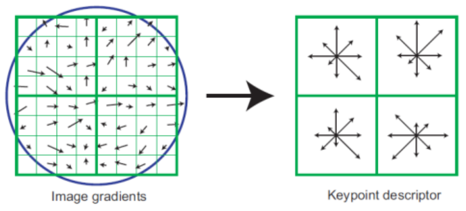
$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,$$

IJERT

  The magnitude and orientation is calculated for all pixels around the keypoint. Then, A histogram is created for this.In this histogram, the 360 degrees of orientation are broken into 36 bins (each 10 degrees). And the "amount" that is added to the bin is proportional to the magnitude of gradient at that point.Once we done this for all pixels around the keypoint, the histogram will have a peak at some point.Also, any peaks above 80% of the highest peak are converted into a new keypoint. This new keypoint has the same location and scale as the original. But it's orientation is equal to the other peak.So, orientation can split up one keypoint into multiple keypoint

- **keypoint descriptor –**
  The region around a keypoint is divided into 4X4 boxes. The gradient magnitudes and orientations within each box are computed and weighted by appropriate Gaussian window, and the coordinate of each pixel and its gradient orientation are rotated relative to the keypoints orientation. Then, for each box an 8 bins orientation histogram is established. From the 16 obtained orientation histograms, a 128 dimensional vector (SIFT-descriptor) is built. This descriptor is orientation invariant, because it is calculated relative to the main orientation. Finally, to achieve the invariance against change in illumination, the descriptor is normalized to unit length.

Feature descriptor



**Training phase:**

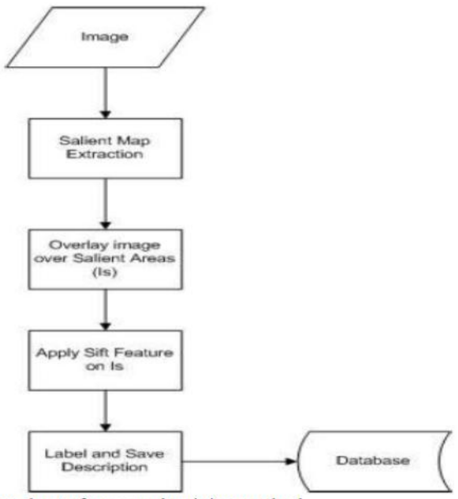
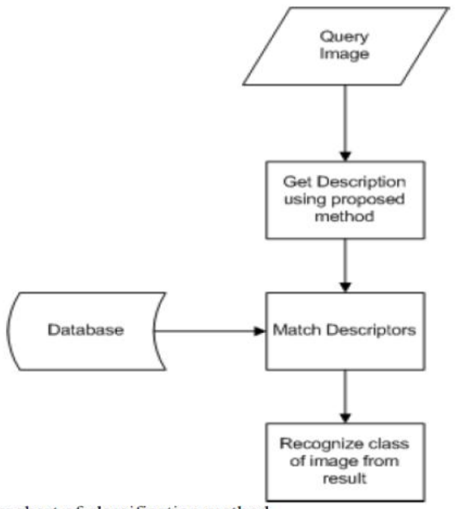


Image
Salient Map Extraction
Overlay image over Salient Areas (Is)
Apply Sift Feature on Is
Label and Save Description
Database

Fig. 1. Flow chart of proposed training method.

**Recognition phase:**


Query Image
Get Description using proposed method
Database
Match Descriptors
Recognize class of image from result

. 2. Flow chart of classification method.

## III. Result:

## IV. CONCLUSION:

In this paper an improvement of the original SIFT-algorithm developed by Lowe was proposed.This improvement corresponds to enhancement of feature matching robustness, so the number of correct SIFT features matches is significantly increased while nearly all outliers are discarded. Also the matching time cost for the case of extracted features into subsets corresponding to different octaves. The new proposed approach was tested using real images acquired with the stereo camera system. The presented experimental results show the effectiveness of the proposed approach.

The SIFT features improve on previous approaches by being largely invariant to changes in scale, illumination. The large number of features in a typical image allow for robust recognition under partial occlusion in cluttered images. A final stage that solves for affine model parameters allows for more accurate verification and pose determination than in approaches that rely only on indexing. The indexing and verification framework allows for all types of scale and rotation invariant features to be incorporated into a single model representation. Maximum robustness would be achieved by detecting many different feature types and relying on the indexing and clustering to select those that are most useful in a particular image.

## V. REFERENCES:

[1] David G.lowe “object recognition from local scale invariant features”B.C.,V6T IZ4,canada lowe@cs.ubc.ca
[2] David G.lowe “distinctive image features from scale-invariant keypoints”B.C., V6T IZ4,canada lowe@cs.ubc.ca,january 5, 2004.
[3] SIFTfeatures “http://www.aishack.in/2010/05/sift-scale-invariant-feature-transform/
[4] C. Harris and M. Stephens. A Combined Corner and Edge Detector. Proc.
[5] David G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints”, International Journal of Computer Vision, Volume 60, Number 2, November 2004.

IJERT