Image-Based Localization Using the Plenoptic Function

Alireza Ghasemi

AudioVisual Communications Laboratory Laboratory
School of Computer and Communication Sciences
École Polytechnique Fédérale de Lausanne

Abstract. In this report we study the ways to exploit the vast amount of information inherent in the plenoptic space and constraints of the plenoptic function to improve the efficiency of image retrieval, recognition and matching techniques. The specific application we are concerned with is image-based location recognition on mobile devices.
The plenoptic space is formed by extending the notion of traditional two-dimensional by adding more dimensions for viewing direction, time and wavelength. Using current mobile devices' built-in cameras, one can easily capture a large sequence of pictures from a single static scene by moving the camera in one direction, which form a three dimensional plenoptic function.

1 Introduction

1.1 Image-Based Localization

Global Positioning System (GPS) is considered the dominant solution for the localization problem, i.e finding the location of a moving object (person, vehicle,...). However, GPS's localization accuracy is not always acceptable.

One of the main drawbacks of GPS is that its performance degrades dramatically in urban environments (in presence of streets, building,...). It causes problem for people aiming to use location-based services in everyday use. Therefore, the localization accuracy should be improved using extra sources of information.

In this work, we study the use of visual information to improve localization. Visual information are readily available these days since cellphones' camera can easily capture photos of the surrounding scenes [11]. These photos can be used to recognize the location at which camera stands. Figure 1 depicts typical photos taken in urban scenes.

Our goal is to develop methods and algorithms to enable location recognition on a cellphone using visual information (i.e. photos taken by the camera of the cellphone). However, as an even bigger picture, we would desire to fuse GPS and visual information (more about this later).

To achieve this goal, we divide the overall system architecture into an offline and an online part. The offline part will be executed on a centralized server which contains a trained model of the urban areas under consideration. The online part, on the other hand, will be executed on the cellphone itself and therefore has to be computationally optimized. In section 1.3 we will talk more about the overall architecture and workflow of the system.

1.2 The Plenoptic Space

For having accurate image-based localization, usually having only one photo from a single viewpoint of a location is not sufficient. It's intuitive that having multiple photos from different viewpoints of the same scene could be beneficial in improving recognition performance. Moreover, using mobile devices' camera it is quite easy to capture a sequence of images rather than a single one, just by moving the camera in a row. therefore we will study methods to perform location recognition based on image sequences taken by mobile cameras.

The plenoptic function and plenoptic space are formalisms through which we can study the properties and regularities of such image sequences. We will discuss these in this section.

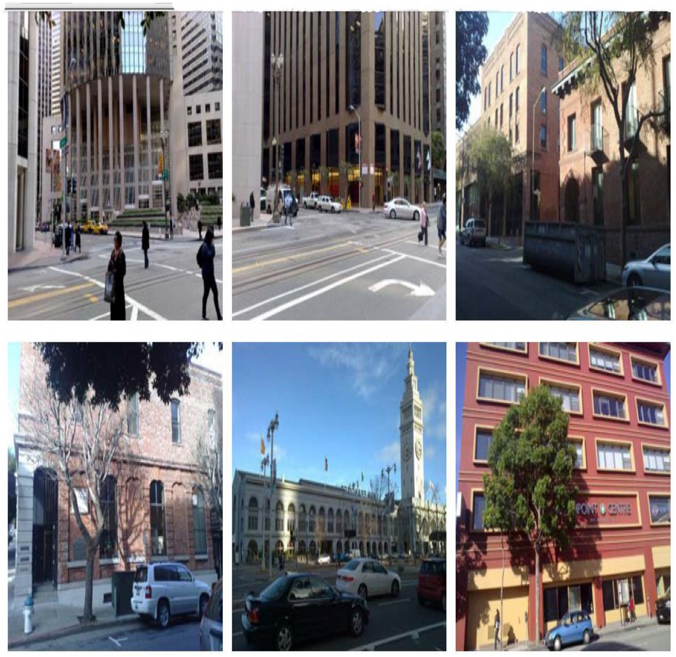


Fig. 1: Examples of Urban Scenes' Photos [11]

The Plenoptic Function in Different Dimensions The traditional notion of an image has two degrees of freedom, one for each of the image coordinates (x and y). In the context of digital signal processing, this translates to a two dimensional matrix.

However, we have the intuition that at least our eyes view in three dimensions. One may also argue that time is the fourth dimensions. Such evidences make us think about adding more degrees of freedom to the traditional notion of an image, which leads to the so-called "Plenoptic Function".

The most general form of plenoptic function contains 7 degrees of freedom (arguments). This is formed by adding 3 degrees of freedom for the spatial camera position, one for time and one more for the wavelength, to the tradition (x, y) arguments of the image plane. Therefore we have:

$$P = P_7(x, y, V_x, V_y, V_z, t, \lambda) \tag{1}$$

In which (V_x, V_y, V_z) is the spatial coordinate of the camera position.

The plenoptic function is highly structured and a significant amount of information about the captured scene can be extracted by exploiting this regularity. However, the general seven dimensional plenoptic function is extremely difficult to capture and operate. Therefore, we usually reduce the number of parameters by introducing constraints on the capture process. For example, we can consider only static scenes, thereby omitting the time index. Moreover we can omit the wavelength by considering single-sensor lenses. We may also enforce restrictions on the camera movements to reduce the number of parameters even more. Figure 2 depicts devices which are used to capture different subsets of dimensions of the plenoptic function.

A special case of the plenoptic function has the special name of "the epipolar volume". Epipolar volumes are easily acquirable using traditional cameras yet contain a significant amount of useful information. We will discuss this special kind of plenoptic function in the next section.

Epipolar Volume An epipolar volume is formed from a plenoptic function by restricting the camera motion to a straight horizontal line (and thereby omitting V_y and V_z from camera parameters), considering only static scenes (thereby omitting t) and also considering only one sensor (omitting λ). This 3 dimensional subspace of the plenoptic function is called the epipolar plane image or simply epipolar volume and is denoted as:

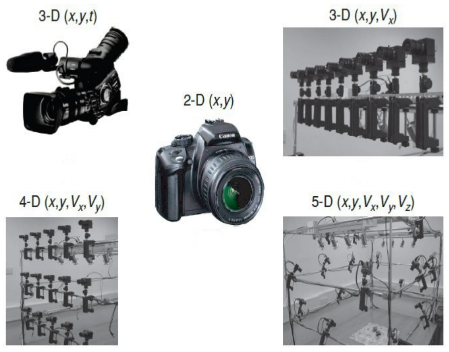


Fig. 2: Capturing the Plenoptic Function in Different Dimensions [6]

$$I = I(x, y, V_x) \tag{2}$$

Therefore, the epipolar volume is a three dimensional function with one dimension for the camera position along the X axis [1] and the other two for the image itself, i.e. horizontal and vertical axis of the image plane.

In the context of digital image processing, an epipolar volume is represented via a three dimensional matrix formed by stacking a sequence of images. Note that it may also be a four dimensional matrix because of the three color channels (RGB) available for each image.

The epipolar volume is highly structured and follows regularity that can be exploited to extract information regarding the scene being captured. In the rest of this section we will study the structure of the epipolar volume.

Figure 4 shows an epipolar volume. Three different slices of the volume are visible in this picture.

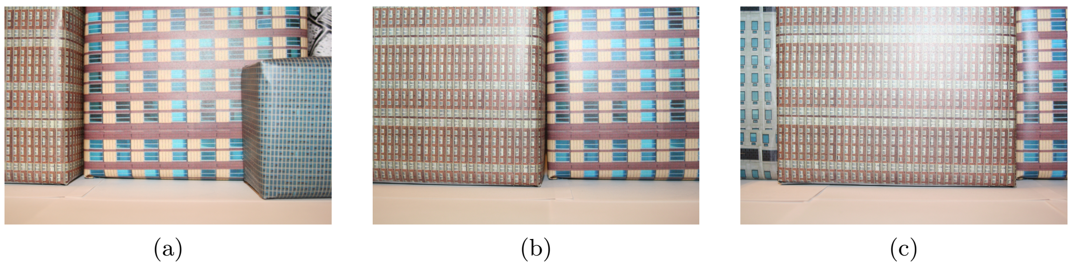


Fig. 3: Three Sample Pictures of a Moving Image Sequence

Characteristics of the Epipolar Volume We start our discussion on regularities of the plenoptic function by studying the properties of the epipolar volume using the pinhole camera model [17].

Consider taking a single picture using the pinhole camera model. Assuming that the focal length of the camera is unity and the optical center is located at the origin, the projection of a scene point (X, Y, Z) to the image plane is calculated as:

[1] or equivalently the time of taking the image, since camera speed is assumed constant. In fact, many literature paper denote the third dimension by t

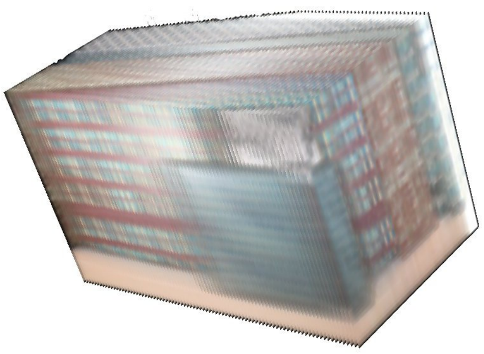


Fig. 4: An Epipolar Volume

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \mapsto \begin{pmatrix} \frac{X}{Z} \\ \frac{Y}{Z} \end{pmatrix} \tag{3}$$

Now consider the case of an image sequence, taken by moving camera V_x units along the horizontal axis. Adding a third dimension for the camera location (set to V_x), now the mapping becomes:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \mapsto \begin{pmatrix} \frac{X}{Z} - \frac{V_x}{Z} \\ \frac{Y}{Z} \\ V_x \end{pmatrix} \tag{4}$$

This is how an epipolar volume is formed. We conclude the following two important facts regarding the equation (4).

1. Each scene points corresponds to a line (a single parameter curve) in one of the $x - V_x$ slices of the epipolar volume (the slice corresponding to $y = \frac{Y}{Z}$).
2. The slope of this line is proportional to the depth (Z value) of the scene point. This means that lines corresponding to closer points are more horizontal.

Please note that the above facts are valid only when all our initial assumptions have been met. For example, if the motion is non-linear or speed varies, then the feature paths are no longer lines. An $x - V_x$ slice of an epipolar volume is called an "epipolar plane".

The two fact described above are the starting points for our analysis of the epipolar volume and its properties and how we use it to extract information about the scene.

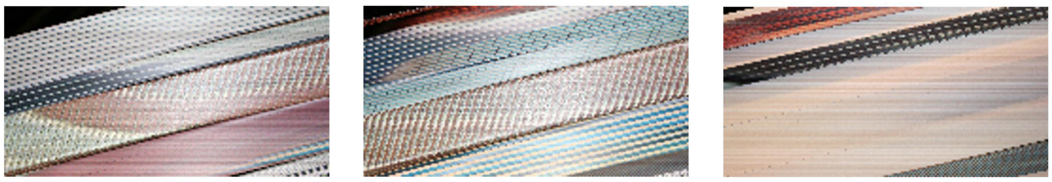


Fig. 5: Three Epipolar Planes of an Epipolar Volume

1.3 The Overall Workflow of the System

The goal of this report is to study the properties of plenoptic function (especially epipolar image volumes) and propose efficient approaches to extract meaningful and discriminative features from such data using their coherent regularity. Our proposed method should as much as possible invariant to small changes in capturing environment and camera parameters and be adaptable to varying number of images in the sequence. the application for which we will use this approach is image-based localization, i.e. finding location of an observer using the photos he captures using his cellphone from the surrounding environment.

Many parts of the system are like a traditional object recognition system [27], but with the custom application of location recognition. There is a training phase in which a model of the urban areas is built and stored in a centralized server.

The other part of the system is the recognition or testing phase, in which the mobile device communicates with the server by sending the information it has about the surrounding environment. The system compares the information with its model and infers the location of the system. Finally the determined location os sent back to the mobile phone.

The overall workflow of the system will be as following:

- **The Offline Training Phase:** In the training phase which will be offline, and needs to be executed only once and on the centralized server we already talked about, a model of the urban areas under consideration is trained using the data (photos) avilabel to the learning algorithm. The following steps need to be done:
  1. *Acquire the plenoptic model of the scene:* We may have to deal with unwanted camera jitters ans non-linear motions. However, in this report we assume that camera motion is perfectly linear.
  2. *Extract meaningful features:* There are some criteria like repeatability, discriminativeness,... which should be met for a feature extraction method to be considered useful. We will talk more about this in section 3.
  3. *Construct a visual vocabulary:* After detecting features, we have to describe them in multi-dimensional vectors. This process is called feature description. Moreover, in order to be able to robustly match features against each other, wee a quantization step (called vector quantization since we are working with vectors). These issues will be discussed in sections 4.3 and 4.3.
- **Online Recognition:** The recognition phase, on the other hand, will be run online, many times, and on the cellphone processor with limited computational resources.
  1. *Capture the plenoptic-space view of the scene:* The camera and capturing settings are usually different from that of training stage. For example the number of captured images may be quite different. We should deal with these difficulties.
  2. *Extract features:* We should do feature extraction in the same way as we did for the training phase.
  3. *Match features of the test images with the visual vocabulary:* It will be partly run on the cellphone and partly on the server. the efficiency issues should be considered.

Finally, the accurate location is estimated based on the meta data available about the images.

1.4 Terminology

In this section we review the definition of the most important terms used throughout this report. These terms are:

- **Plenoptic Function:** Generalization of an image where as well as horizontal and vertical coordinates of the image we can vary the location of the camera in space (3DOF in the most general form) and time and wavelength (The most general form).
- **Epipolar Image Volume:** A special of the general plenoptic function where the scene is static and camera can only move horizontally perpendicular to the image normal. It's a 3d function: $f(x, y, V_x)$ where V_x is the image index or the camera location.
- **Epipolar Plane:** A horizontal slice of the epipolar volume, i.e a function of the form $f(x, y_0, V_x)$ where y_0 is constant.

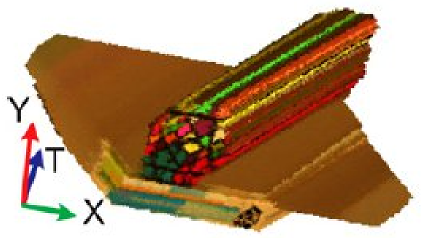


Fig. 6: Layer Extraction from Images Using Plenoptic Information [13]

– **Epipolar Line:** A straight line segment in the epipolar plane. Epipolar lines are also called feature paths since each line corresponds to a unique scene point.

The rest of this report is structured as follows: We first review some related and literature (section 2). After that we propose some approaches based on plenoptic information for feature extraction and study their properties (section 3). Finally we illustrate some initial results of implementing the proposed algorithms (section 4).

2 Related and Similar Works

In this section we will review the important research studies related to our work on plenoptic function.

The concept of plenoptic function and its properties was first introduced and discussed by Adelson and Bergen in [1]. Before them, [8] had proposed the concept of epipolar plane image (EPI) volume which is now known to be a special case of plenoptic function. They used the EPI volume for depth estimation.

In a more recent work, [6] introduced the concept of a plenoptic manifold. Plenoptic manifolds are trajectories traversed by scene points in the plenoptic space. Understanding plenoptic manifolds is crucial in studying the inherent regularity of plenoptic function. This regularity can be exploited in many different ways for processing and extracting information from images, as previously done by Bolles [8] and more recently by [37] for depth estimation.

The algorithm proposed in [5], exploits the regularity of the plenoptic function for more accurate image segmentation. It uses a modified version of the active contour algorithm which takes into account the depth information extracted from the plenoptic volume.

In [13] another work is presented which uses the plenoptic function for extracting information from images. The authors propose to extract layers (regions of image which lie in the same depth) of the image using the information contained in multiple views of the scene.

The work in [3] uses T-junctions present in the epipolar planes as occlusion indicators to improve motion segmentation accuracy in video sequences. Their approach is an intermediate way between global methods (which assume that the camera motion is fully linear) and local approaches which do not have any assumption about camera motion. Indeed, in their work they assume the overall camera motion can be divided into locally linear parts. They detect T-junctions of epipolar planes in each part, and then estimate the edge occlusion energy using these. Finally, a customized graph-cut segmentation method using the T-junctions' information is developed. Figure 7 depicts some typical T-junctions in a movie scene.

In [40], a distributed object recognition system designed specifically for bandlimited camera networks is proposed. The main similarity between this work and our idea is the use of multiple cameras. However, the viewing angles between camera are not necessarily linearly related. The contribution of the paper is reducing the amount of transferred information using compressed sensing.

[16] utilizes the regularity of the plenoptic space and the redundancy inherent in the epipolar lines to obtain a sparse representation of multi-view data. Redundancy of the information in epipolar planes is exploited in order to partition the image into layers (regions of the same depth). The layer extraction is performed using a special segmentation algorithm which takes into account the

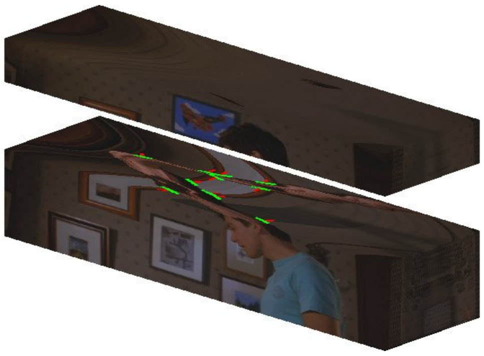


Fig. 7: Using T-Junction for Occlusion Detection and Thereby Segmentation [3]

occlusion information obtained using plenoptic information. To obtain the sparse representation, layers are decomposed using a Discrete Wavelet Transform (DWT) along the view domain and then a two dimensional DWT in the image domain.

[14] uses information of gradients of epipolar lines for better segmentation of a multi-view image. They first extract epipolar lines using an improved Hough Transform algorithm. The improved Hough Transform tries to reduce erroneously-detected lines by exploiting the prior information about the epipolar manifolds. Among these prior information is the fact that all epipolar lines' slopes have the same sign (either positive or negative). Moreover, knowing that the same feature point should be available in many of the images in the sequence, we can adjust the threshold parameter of the Hough Transform. Another fact we can use is that intensity of the points along a line should be the same (assuming consistent lightning conditions). Therefore the variance of intensity of points along a line should vanish for correctly detected lines.

After detecting lines, they are grouped based on their slopes and an initial segmentation is formed this way. The initial segmentation is refined using a boundary growing method. The boundary growing method tries to form a continuous boundary around each segment. Finally a contour linking procedure is used to complete the segmentation. Figure 8 shows how the initial segmentation is formed by grouping points based on the slope of their corresponding epipolar line.

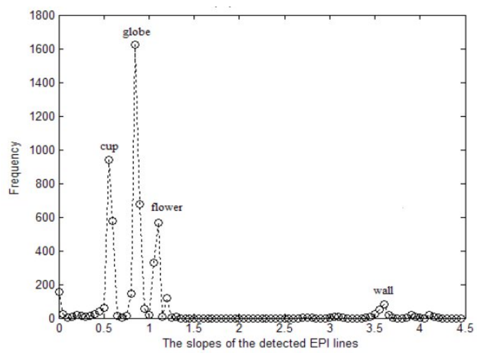


Fig. 8: Initial segmentation of a multi-view image using slopes of epipolar lines [14]

In the work presented in [42], the assumption of exactly linear camera motion has been slightly relaxed. In their work, they use plenoptic function for 3D model reconstruction. However, they use a preprocessing image-stabilization method if the camera has not been moving exactly linearly. They use the information in the Fourier domain of the epipolar plane in order to detect principal orientations in the image in a more robust manner. Moreover, they use Panoramic View Image (the $y - t$ slices of the EPI cube) as well as the epipolar planes to better reconstruct the scene. Figure 9 shows a typical Panoramic View Image.

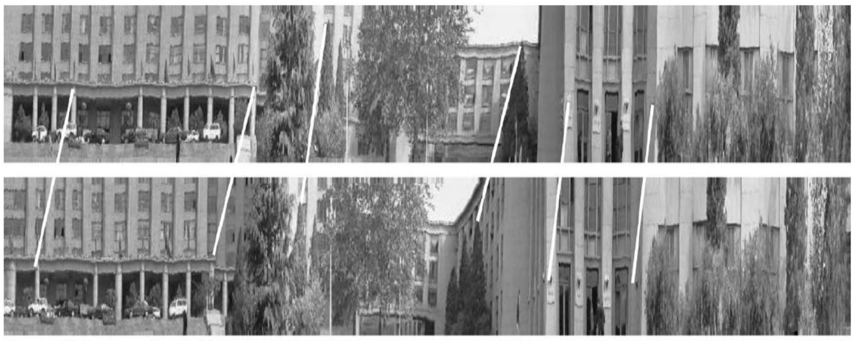


Fig. 9: A Typical Panoramic view Image [42]

In [15], the epipolar planes have been used in a different way. Instead of considering individual epipolar planes, they are summed up into a single average epipolar plane which is called Condensed Image Slices (CIS). The same summation is applied to Panoramic View Images which yields Condensed Motion Slices (CMSs). CMSs and CISs are used to estimate the shaking parameters of a jittering camera and thereby rectify the camera motion. The overall goal of this paper is the Route Panorama acquisition.

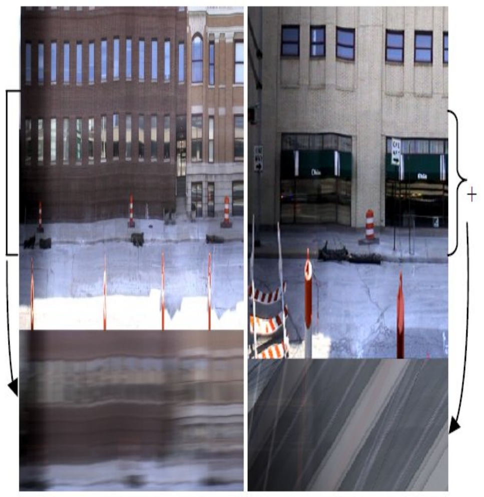


Fig. 10: Condensed Image Slice (Bottom Right) and Condensed Motion Slices (Bottom left) [15]

In [21] a compressed representation for multi-view videos is presented which allows for efficient storage and retrieval in Free Viewpoint TV systems. To achieve this goal, a sequence of depth images are considered instead of natural images, which leads to the concept of Epipolar Plane "Depth" Images. Epipolar Plane Depth Images (EPDIs) are used in conjunction with EPIs in order to generate the global depth map and texture map of a scene and finally reconstruct the scene photo-realistically from an arbitrary view point. Figure 11 shows a typical EPDI.

The work in [31], proposed the "polydioptric camera". A polydioptric camera is created by arranging a group of pinhole cameras in a matrix (Figure 12). In fact, it is the physically implementable counterpart for the abstract concept of a plenoptic camera (in which the space between individual cameras should be zero which is not implementable). Using the information from polydioptric cameras, a new set of motion equations is developed. These equation are used to estimate the rigid motion of the camera and solve the Structure from Motion (SfM) problem. Figure 13 depicts the difference in capture process between the polydioptric and pinhole camera. The overlapping that

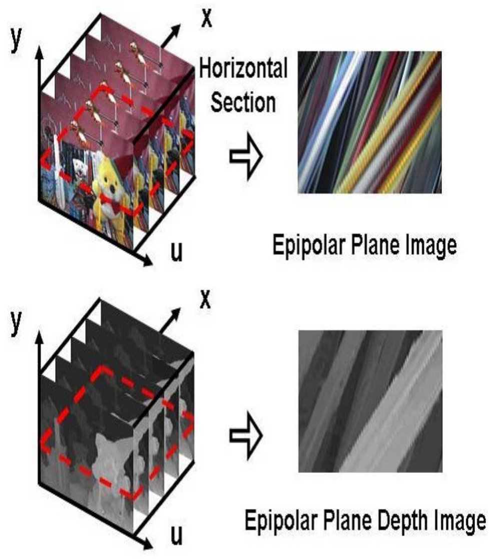


Fig. 11: Epipolar Plane "Depth" Image [15]

occurs in the epipolar plane between images captures by the polydioptric camera, allows for more efficient motion estimation.

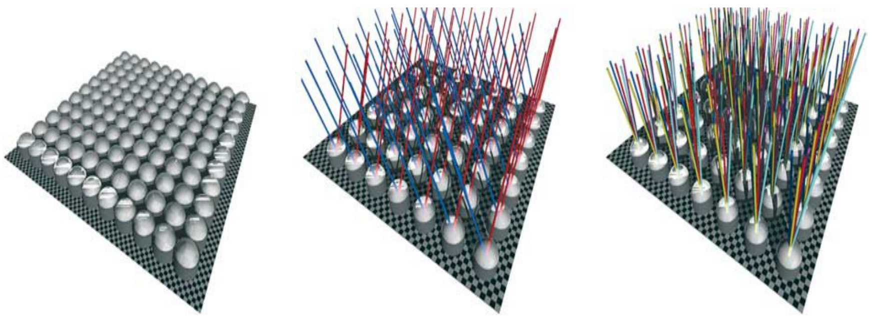


Fig. 12: Constructing a Polydioptric Camera [31]

The work in [20] is among the few attempts to use plenoptic information for a recognition task. The goal of this paper is to detect vehicles parked permanently on the sides of the streets. To achieve this goal, a line scan camera is mounted on the roof a car and traversed along the street. This setting acquires epipolar plane images. The presence of a parked car is then detected by analyzing the spatial distribution of slopes of epipolar lines. Presence of a car corresponds to a peaky distribution (Figure 14). When no car is present, the peaks are not observed (Figure 15)

In [29], a typical application of epipolar images, i.e depth estimation is studied. Again, the linear camera motion constraint is relaxed and a more general epipolar image is obtained. Using equations derived from epipolar geometry constraints, the problem of depth estimation is solved and a depth map for the given scene is obtained.

Another class of slightly related works in the literature are those that have addressed the problem of three-dimensional object recognition. Here, by three-dimensional we mean the $x-y-z$ space, not the $x-y-t$ as in the plenoptic function. Although these approaches do not address the problem of object recognition in plenoptic space directly, they could be useful since objects carve some spatial shape in the plenoptic space which could be considered some kind of a 3d objects and approached using traditional 3d object recognition schemes. The work in [25] is one such method. This work uses a three-dimensional extension of the SIFT keypoint detector to extract meaningful features from three-dimensional shapes. A probabilistic Hough Transform is used after feature extraction in order

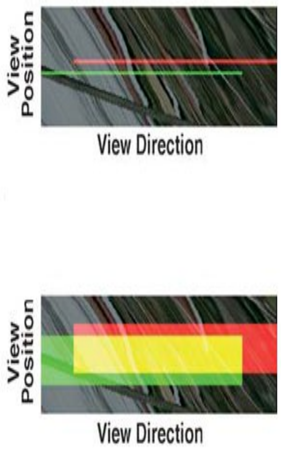


Fig. 13: Capturing a Plenoptic Function Using a Pinhole Camera (Top) and a Polydioptric Camera (Bottom) [31]

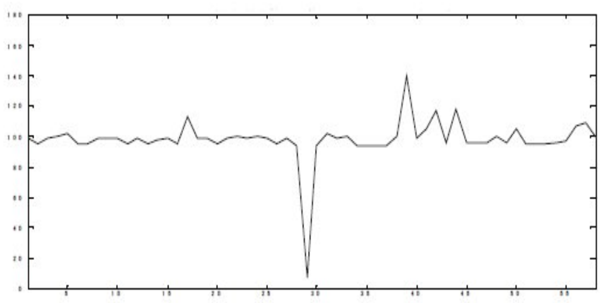


Fig. 14: Spatial Distribution of Epipolar Lines' Slopes In Presence of a Foreground Object (Car) [20]

to extract implicit shape models and improve recognition accuracy. Figure 16 shows sample 3d SURF features extracted from a 3d model. [24] is another similar work in this class. In [41] a thorough comparison of different three-dimensional feature extraction techniques has been conducted.

In [28], the regularity and redundancy of an epipolar image sequence is exploited to extract accurate feature correspondences between the first and the last image of the sequence. The paper uses a cost function based on the variance of intensity along candidate epipolar lines. Recall that in an epipolar plane, the intensity should not change along epipolar lines.

Wang et. al. proposed a multi-view object recognition algorithm based on properties of epipolar volumes [38]. In their approach, a disparity estimation method is utilized to improve the performance of single-view object recognition.

Another idea which is in some sense related to feature extraction from epipolar volumes is the concept of space-time interest points [26]. Space-time interest points are extensions of the concept of feature points (which are spatial) to both space and time dimensions. It means that space-time interest points are corners in space-time volumes corresponding to videos for example [39].Figure 17 may be useful in understanding what space-time interest points are. In the left and right figures we can see how the space-time domain looks like for two typical events: collision of a ball with a wall and movement of a corner. The blue circle in each graph represents a typical space-time interest point.

In [39], a method based on space-time interest points is proposed for human action recognition. Interest points are detected simultaneously with their scale using the three dimensional Hessian

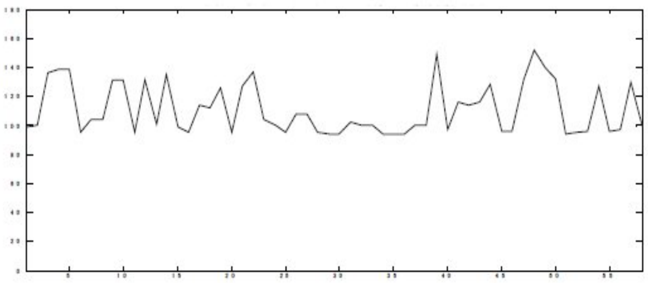


Fig. 15: Spatial Distribution of Epipolar Lines' Slopes When Only background is Available [20]

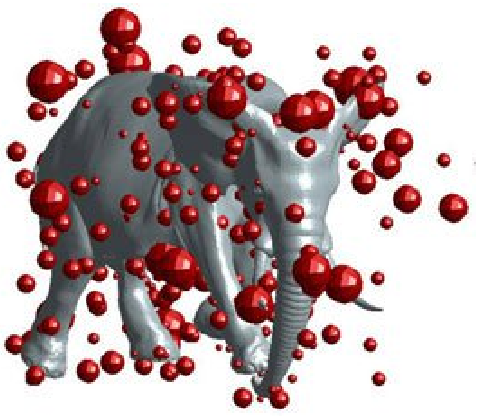


Fig. 16: Sample 3d SURF Features Extracted From a Three-Dimensional Shape[25]

matrix. In their approach, 3d box filters are used to approximate second derivatives of Gaussian, in order to increase speed. These filters are depicted in figure 18.

The main application of space-time interest points is in activity recognition from video sequences [9] which is far from plenoptic image retrieval. However they bear some similarities with the field of plenoptic image processing. Both approaches are extension of simple two dimensional feature points and try to use these features for recognition. However, in space-time interest point detection usually no assumption is made regarding the camera motion and therefore information in epipolar planes is not utilized. A detailed introduction to space-time interest points with their properties can be found in [26].

There are also some works which address a recognition task from multiple views, but without assuming linear camera motion. [19] is one of them which approaches the problem of object recognition using multiple views. Their approach works by first extracting a set of scale-invariant features from the images and then deriving probabilistic model for those features. The recognition is performed using a Maximum Likelihood approach.

[18] is another work which tries to use stereo vision (using two images from slightly different viewpoints) to solve the object recognition problem. Again a probabilistic maximum likelihood approach is used to match features between the two viewpoints. Note that in a plenoptic setting, we don't need expensive computation to obtain matchings since we can do this easily using the regularity of the plenoptic function.

Another area in which some research works have been done in feature extraction and retrieval of multi-dimensional, volumetric data is the field of Medical Image Analysis. Many types of medical images (like CT scans,...) are inherently multi-dimensional (They are sliced). Therefore, dealing with them and analyzing their information requires extracting features from these volumetric data. The nature of medical images and the types of typical shapes present in them is definitely quite different from that of natural scenes, like the urban scenes we are concerned with here. However, the techniques used for analysis of multidimensional medical images can be useful in leading the direction of our thought to the direct direction of developing methods for plenoptic function analysis.

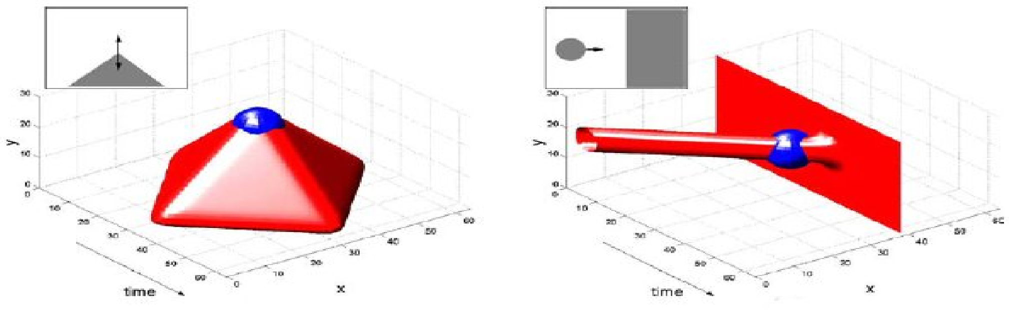


Fig. 17: Space-Time Domain and Space-Time Interest Points [26]

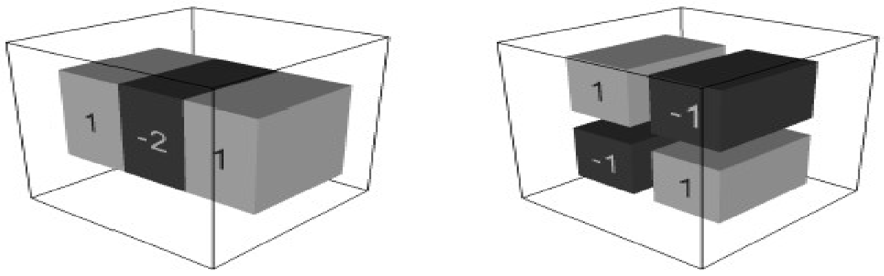


Fig. 18: Box Filters to Approximate Three-Dimensional Gaussian [39]

The works in [2] and [32] are examples of multidimensional medical image analysis systems. They use a direct extension of SIFT, called 3d SIFT, in order to extract features from medical volumetric images. Their 3d SIFT is in many ways similar to that of [25] which we saw was used for 3d object recognition.

As a generalization to the above works, [12] proposed n-SIFT, which is an arbitrary-dimension generalization of the famous SIFT method. n-SIFT's main application has been in medical image analysis. However, nothing prevents it from being used in other similar areas as well.

3 The Proposed Feature Extraction Method

Recall from the introduction the importance of the epipolar planes in extracting information from three dimensional plenoptic function. We showed there that: 1) Each scene point (i.e. distinguishable feature) maps to straight line in one of the epipolar planes in the plenoptic function which we call "feature path", and 2) The slope of a feature path is proportional to the depth of its corresponding feature point.

In this section we propose and study different way to exploit this information in obtaining high-quality feature points.

By high quality we mean that the feature points should be discriminative and repeatable. That is, the same set of feature points should be detected under illumination, scale and orientation changes and the features must be discriminative enough to distinguish between different scenes.

3.1 Detection and Extraction of Lines

Now, we know that the first step in any feature extraction method for epipolar volumes should be detection and extraction of line segments in epipolar planes.

The most common method for line extraction from images is the Hough transform [36]. In the Hough Transform, a line is parametrized as the set of (x, y) points satisfying the following equation:

$$x \cos\theta + y \sin\theta = \rho \tag{5}$$

We can see that in this model that the set of parameters (ρ and θ) is different from those used the traditional representation of two dimensional lines (slope and intercept). The reason for using this parametrization is that using the traditional model $y = mx + h$, the slope m is not bounded whereas θ in Hough model is bounded to $-\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2}$.

Having this model we now define the new $\rho - \theta$ space. The duality between the original $x - t$ space [2] and the transformed $\rho - \theta$ space is such that each point in the $x - y$ plane is mapped to a harmonic curve in the $\rho - \theta$ plane. Moreover, each point in the $\rho - \theta$ space corresponds to a unique line in the $x - y$ plane.

The Hough Transform works by constructing a $\rho - \theta$ plane of the candidate lines of the $x - y$ plane. Usually an edge detection algorithm is applied to the image as the preprocessing step. There are two reasons for this. First reason is that Hough Transforms works on binary images. Therefore some kind of thresholding is essential to apply to the image. The second reason is that edge detection helps in removing irrelevant points since line usually lie in th edges of the image.

As well as edge detection and before that, a low-pass filter is also appropriate to apply to the image. The reason for this is to reduce noise before edge detection is applied so that it doesn't detect noisy points as edges.

Figure 19 shows pseudo-code of the Hough Transform algorithm.

A set of collinear points in the $x-y$ plane correspond to intersection of their corresponding curves in the $\rho - \theta$ curve which means more intensity in the points of intersection. Therefore, prospective lines correspond to $\rho - \theta$ points with highest intensity (i.e. peaks) and we should select points with intensity higher than a threshold τ_1.

```
**Require:** The two dimension matrix $I$ representing an image.
 1: Apply a low-pass filter to $I$.
 2: Apply an edge detection algorithm to $I$ and convert it to a binary image.
 3: Discretize the range of $\theta$ values in the vector $\Theta$.
 4: Discretize the $\rho$ parameter into $n_\rho$ distinct values.
 5: Construct the $Length(\Theta) \times n_\rho$ output matrix $H$.
 6: Set all elements of $H$ initially to 0.
 7: for each feature point $(x, y)$ in $I$ do
 8:     for each $\theta \in \Theta$ do
 9:         $\rho = x \cos\theta + y \sin\theta$
10:         $H(\theta, \rho) = H(\theta, \rho) + 1$
11:     end for
12: end for
13: return the output image $H$
```

Fig. 19: The Hough Transform Algorithm

After selecting peaks from the $\rho-\theta$ plane, we can form the lines in the $x-y$ plane by determining their two endpoints. In this step we can prune the set of detected lines and only retain the most stable ones. For example we can omit the lines shorter than a threshold τ_2. Figure 20 depicts the visualized results of running the Hough Transform on an epipolar plane.

Improving Line Detection Using Characteristics of Epipolar Planes The Hough transform is the most common way to detect arbitrary lines in an image. However, due to the increased number of parameters and noise and textures of the images, some erroneous results are also returned by the algorithm.

As well as the problem of erroneous lines, Hough transform also faces the problem of discretizing values of ρ and θ which causes the algorithm to run quite slowly. We can use the prior information

[2] Remember that Hough Transform is applied to epipolar planes which are in the $x - t$ space

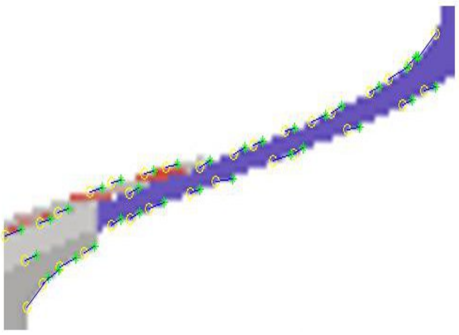


Fig. 20: Lines Detected by the Hough Transform on an Epipolar Plane

on characteristics of epipolar planes to improve both the accuracy and time complexity of the Hough transform.

To improve runtime complexity of the Hough transform we notice the fact that, by assuming camera movement in a single direction, all lines in all epipolar planes will have either positive or negative slope, and not both. If the camera moves from left to right, all epipolar lines will have positive slope and if the camera moves from right to left, all epipolar lines will have negative slope.

We can use this fact to compress the search space for θ and thereby time complexity of the Hough transform. We don't need to discretize and search the whole range of θ $(-\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2})$. We have to just search half of it (either $-\frac{\pi}{2} \leq \theta \leq 0$ or $0 \leq \theta \leq \frac{\pi}{2}$). This reduces running time of the algorithm by a factor of 2.

To further reduce the running time of the algorithm, we note the fact that Hough Transform's runtime complexity is proportional to the number of feature points (in a binary image). Applying a low-pass filter to the image before thresholding and edge detection reduces the number of "on" points in the output image. Moreover, low-pass filter has the extra effect of reducing the effect of textures and noise in computing the hough transform.

To improve accuracy, we note that each epipolar line (i.e. a feature path) corresponds to a single unique feature. It means that all point along an epipolar line should ideally have equal intensity (or color). Therefore we can do a post-processing step after the hough transform and ignore lines for which variance of intensity along the points is above some threshold $\sigma_{intensity}$. More complicated post-processing steps may similarly apply. For example,we can easily see that the sign of slope of all epipolar lines should be the same. Another useful prior is that the number of comprising a line should be in some sense proportional to the number of images in the sequence.

3.2 Selection of Feature Points

After applying Hough Transform to all epipolar planes, we have the epipolar line segments which are paths traversed by image features. Now we can use these lines to extract visual features from the epipolar volume.

As noted earlier, each epipolar line segment corresponds to a single unique image features. So we can simply use the starting point of each line segment as a feature point. Considering the epipolar plane at which the line resides, we can compute the full three dimensional coordinates of the feature point $((x, y, V_x))$ and the specific image (V_x index) in which it lies. These information will be used to describe the point as a feature vector.

Note that all points of a line segment represent the same feature. Therefore it does not actually matter which point (along the line segment) is selected as the feature points. We can equivalently select the endpoint or middle point of the line segment as well.

Applying the same extraction to all epipolar planes and aggregating the results, we collect the complete set of feature points describing the whole epipolar volume.

Using this approach we avoid correspondence estimation between images of the volume. We also don't need to perform general feature tracking since we are using the prior information about the camera motion to efficiently recover feature paths. Therefore we are using all available information in extracting this set of feature points.

Th size of the feature set is significantly smaller than the case when we simply perform traditional feature extraction on all images of the volume and aggregate the results. This is because of the fact that many redundant features (points along the same epipolar line) are ignored. A second reason would be that many unstable features (those that appear only on a few images) are not selected using our proposed approach since we omit lines shorter than a threshold. A feature should be present in (at least) more than one image to be selected by our approach.

Figure 21 depicts the pseudo code for the proposed feature detection method.

```
Require: The input epipolar image volume $V$.
 1: Set $S = \varnothing$.
 2: for each epipolar pplane $P$ in $V$ corresponding to $Y = y_0$ do
 3:     Apply a low-pass filter to $P$.
 4:     Apply edge detection to $P$ and convert it to a binary image.
 5:     Apply the Hough Transform to $P$ and extract all line segments.
 6:     for each starting point $(x_i, V_{xi})$ of a line segment do
 7:         Select $(x_i, V_{xi})$ as a feature points.
 8:         Add point $(x_i, y_0, V_{xi})$ to the set of features $S$
 9:     end for
10: end for
11: return the set of feature points $S$
```

Fig. 21: The Proposed Feature Detection method

3.3 Invariance to Scaling, Rotation and Illumination Changes

An efficient feature detector should be invariant to transforms in the scene. It means that the algorithm should detect and extract the same feature points even after applying some (at least) simple transforms to the image, so that recognition and correspondence becomes possible.

In this section we will intuitively show that our feature detection approach is invariant to simple affine transforms that happen quite frequently in real-world capturing scenarios.

Three of the most common transforms occurring in natural pictures are scaling, rotation and illumination change. We start our discussion by arguing about how transforms in captured images change corresponding epipolar planes.

Scaling Scaling an image can be described by moving the camera further from or closer to the scene, i.e. moving the camera perpendicular to the motion axis. Therefore we can easily conclude that scaling is equivalent to increasing the depth of all scene points. As we already showed, depth of feature points is proportional to their corresponding feature path (epipolar line).

Summing the two facts stated above, we can conclude that scaling the in the image plane corresponds to a kind of rotation in the epipolar planes. It is not a simple rotation since slope of the lines can not exceed infinity (fully vertical). Therefore, more distance object (those with larger Z value) rotate less than closer objects. Figure 22 depicts this phenomenon.

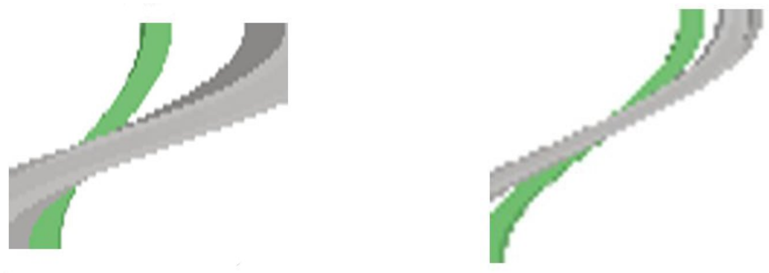
Moreover, we know that Hough Transform is rotation invariant. A rotated line segment is still a line segment and can be detected by the Hough Transform.

Aggregating the facts stated above we can conclude the final lemma as below:

Lemma 1: The proposed feature extraction method is invariant to scaling the image plane. It means that after moving the camera further from or closer to the scene, the same features are still extracted using the proposed algorithm (excluding those points that disappear due to decreased resolution)

Rotation Rotation can not be described in the epipolar plane as easily as scaling. However we try to describe it using simpler basic transforms to justify about invariance of our algorithm to rotation.

When the same scene is captured with two or more different camera orientations (i.e. one is rotated with respect to the other in the image plane) what happens in the epipolar plane is that

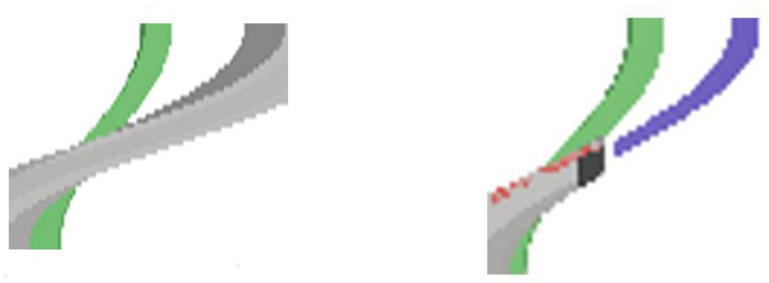


(a) Original Epipolar Plane

(b) The Same Epipolar Plane in the Scaled Volume

Fig. 22: Effect of Scaling on the Epipolar Planes

some point are delivered from their original epipolar plane (i.e. Y index) to another epipolar plane and merge with original points in that plane. If no occlusion occurs in the new plane, then we can expect that those points form the same feature path, but in the new plane. Figure 23 shows a typical effect of rotating the camera on the epipolar plane for a simple scene.



(a) Original Plane

(b) Plane in the Rotated Iamge

Fig. 23: Effect of Rotation on the Epipolar Planes

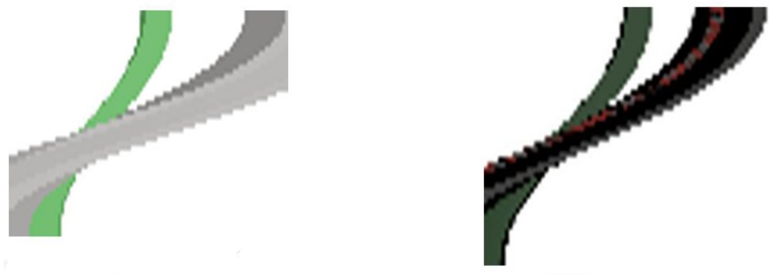
According to the evidence stated above and remembering the fact that extraction algorithm is executed on all epipolar planes, we can conclude the following lemma:

Lemma 2: The proposed feature extraction method is invariant to rotation of the image plane. It means that after rotating the camera , the same features are still extracted using the proposed algorithm (excluding those points that disappear due to occlusion with original point of the destination plane)

Invariance to Illumination Changes It is easier to describe the effect of illumination changes on the epipolar since the effect is quite minor. Change of illumination of the scene is equivalent to changes of intensity of points in the epipolar planes. However, since an edge detection algorithm is applied to the planes before Hough Transform and the only important factor in determining edges is the relative intensity of scene points, we can conclude the following lemma:

Lemma 1: The proposed feature extraction method is invariant to illumination changes in the image plane, provided that the modified illumination is applied uniformly to all of the scene and all captured images. It means that after changing the light source, its position or its power, the same features are still extracted using the proposed algorithm (excluding those points that disappear due lack of brightness)

Figure 24 shows how changing illumination affects epipolar planes.


(a) Original Epipolar Plane

(b) The Same Epipolar Plane with Illumination Changed

Fig. 24: Effect of Scaling on the Epipolar Planes

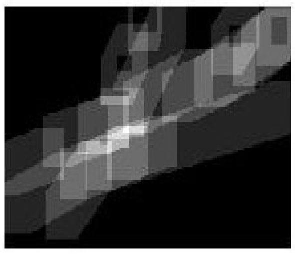


Fig. 25: Significance of Points in the Sliding Window Approach

3.4 Alternative Approaches

Extracting feature points directly from the output of the Hough Transform has the benefit of utilizing the information of the epipolar planes. However, the set of selected points is highly dependent on the edge detection output and also the traditional criteria for feature detection (corner,...) are not explicitly addressed (although they are usually met). In this section we will have a look at other possible ways for feature extraction in the plenoptic space.

The Sliding Windows Approach Instead of using starting point of each epipolar line as the feature point, we can use information about the local distribution of lines and their orientations in different parts of the epipolar planes to find more robust feature points.

To accomplish this goal, we consider a window of fixed size w around each prospective feature point. We find all epipolar lines which fully or partially lie inside this window. The number of lines and the distribution of their orientations is a good measure of how significant a prospective feature point is. We call this measure the "Significance Ratio".

The significance ratio measures the number of epipolar lines passing through neighborhood of a point. The more lines passing from neighborhood of a point, the more significant the point is and more probably it will be a feature point. As well as the number of lines, we may also consider their orientations. The windows in which there are lines with many different orientations can be considered more significant than those with many equal-orientation lines in neighborhood. Variance of orientations (or slopes) of lines could be considered a good measure of diversity. By considering line variances, we are implicitly assuming that feature points lie in area in which depth changes. This is usually true.

In the most general case, we can compute the significance ratio for all points in all planes and select as features those points whose significance ratio exceeds a predefined threshold. Figure 25 shows the significance map for an epipolar plane. Brighter points have higher significance.

Two-Dimensional Corner Detection As Preprocessing The sliding window approach described above runs significantly slow, especially with a naive implementation. This is mostly because of the fact that in the most general case, significance ratio should be computed for all points of plane and maximal points get selected as features.

To improve running time of the sliding windows approach we should reduce the number of points for which significance ratio is calculated. It means that we have to obtain an initial set of candidate feature points and then prune the set using the significance ratio.

The obvious way for obtaining the initial feature set would be using a traditional two-dimensional feature detection algorithm as the pre-processing step to obtain a reduced set of feature points. Then we can compute the significance ration on these points and retain only those points for which the ratio exceeds a threshold $r_{threshold}$. However, this approach does not consider correspondences between points is therefore vulnerable to redundancy.

The Hybrid Two-Step Approach The modified sliding window approach described above has the main advantage that it uses the plenoptic function to prune the set of selected feature points. However, the runtime complexity is still relatively high and moreover, the sliding window approach does not explicitly consider the problem of correspondent points in different image planes. Multiple image points that are mappings of the same scene point should not be retained in the final feature set since they are highly redundant. In this section we propose an improved hybrid approach which addresses these problems.

Our proposed approach is a two step method taking into account both the image plane and epipolar plane information in finding feature points.

In the first step, we apply a two-dimensional corner/feature detection approach to all of the image planes and extract a relatively large set of prospective feature points.

In the second step, we move to the epipolar plane space and extract the lines formed (only) by the prospective points of the previous step. To achieve this goal, first we repartition the points from consecutive image planes to consecutive epipolar planes. Then, we form a binary image in place of each epipolar plane by considering a bright point ("1") for points which correspond to a prospective feature.

We apply the Hough Transform to this synthetic epipolar plane and extract the lines using the transform space. Finally, we retain only the starting point of each line and omit other points, in oder to minimize redundancy.

Our new approach has some advantages over the methods proposed earlier. First of all, it uses information in both image space and the epipolar plane space. Therefore, this approach is more accurate in determining corners and other features of the scene.

The second advantage is that it uses plenoptic information in determining correspondence between points in consecutive images and thereby avoids redundancy in the final feature set.

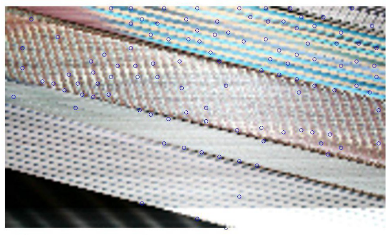
Moreover, in the two-step approach Hough transform can be made significantly faster since it is not executed on the whole epipolar plane but only on a selected subset of its points. Since runtime complexity of the Hough Transform is dependent on the number of bright points in the input binary image, the overall running time is significantly reduced. Figure 26 depicts sample feature points detected by the two step approach in the image and epipolar plane space.

3.5 Matching Between a Single Image and an Image Sequence

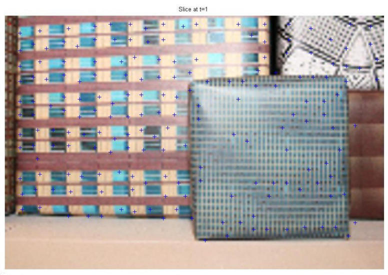
So far, we have considered only the problem of extracting and describing feature points of a three-dimensional epipolar plane. We have assumed so far that in both training and testing phase, we have access to a sequence of images taken using a moving camera from a static scene (epipolar plane volume settings).

However, in a real-world scenario we can't usually assume a full plenoptic image as the test sample. Instead, a single image or a much shorter sequence is given as the input test sample for which we should find the corresponding scene. In this section we will see how this can be approached using our proposed methods.

The fundamental prerequisite to be able to do matching between a single image and an image sequence, is that equivalent or similar types of features get extracted from both data types (image and sequence).



(a) Epipolar Plane



(b) Image Plane

Fig. 26: Two-Step Point Detection in the Image Plane and the Epipolar Plane

This is the case in our two-step method proposed above. In the two-step method, we extract a set of two-dimensional feature points from all images in the sequence, and then remove unstable and redundant points using the regularity constraints of the plenoptic function.

This process can be easily adapted to the case of varying number of images in the sequences, and the extreme case of a single image as well. In a single image the epipolar planes are not available. Therefore, the first step becomes more dominant and less or no redundancy removal is performed. However the whole process is still executable. Since the type of extracted features is exactly the same, we expect that the correct scene is found by matching features of the image against features of the sequence.

3.6 Feature Description

After detecting feature points, we should describe them using multi-dimensional feature vectors. This is done using a feature description method. There are various feature description algorithms, generating vectors of various dimensionality. An appropriate feature description should be as short as possible while retaining distinguishability between different features.

To obtain a feature vector, the feature point itself as well as its neighboring points are considered and based on intensity of these points, a feature vector is extracted. The extracted vector should be as much as possible invariant to affine transformations in the image.

Various feature descriptors have been proposed in the literature so far. Among them, SIFT[27] has shown good performance. However SIFT is unacceptably slow, especially for real-time applications. In recent years, a simpler and faster feature descriptor has been proposed called BRIEF[10] which generates a binary vector, instead of SIFT's real vector. BRIEF is not as accurate as SIFT but its significance speed improvement makes it a choice for real-time applications.

For our proposed plenoptic feature extraction methods, there are two possible choices. One choice is to consider a three-dimensional (x,y and V_x) neighborhood of feature points and therefore a three-dimensional feature descriptor. For example, BRIEF can be easily extended to three dimensions.

The other choice is to use a traditional two dimensional descriptor by considering only the image plane in which the feature point falls. This is reasonable since feature paths all represent a single unique feature and therefore it may be redundant to consider multiple image planes to extract a feature descriptor.

Using a two dimensional descriptor has the extra advantage that it makes it easier to perform matching between volumes of different thicknesses and even matching between a volume and a single image, as explained in the previous section.

4 Experiments

In this section we will demonstrate some of the experimental results obtained by the proposed feature extraction methods.

4.1 The Environment

We implemented the algorithms in MATLAB. For testing the algorithms we collected both synthetic and real datasets. Since the application for which we proposed the methods in the previous section is location recognition, we made virtual and real (although miniature) models of urban furnitures (buildings with different texture,...) for testing our methods.

For capturing a real scene, we used a precise plenoptic device which is able to capture four-dimensional plenoptic function of a scene by allowing accurate two-dimensional motion control of the camera. We constructed a physical model of an urban scene with miniature buildings of various sizes and with various textures. As we already said, the reason for using an urban scene is that the urban image recognition will be principal testing environment for our system. Figure 27 depicts some samples images from this set.

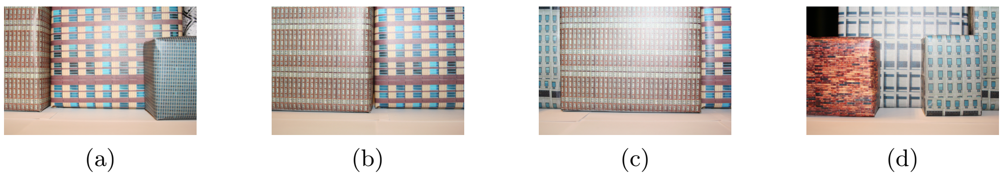


Fig. 27: The Constructed Physical Model

For the synthetic model, we used the Blender rendering software [3]. We created a simple scene with a single building-like cube structure and a simple texture, so that we could easily see the effects of changing the light source and camera location and parameters (Figure 28).

Moreover, we created a set of more realistic building models in order to have easier control over scaling, orientation and lightning conditions (Figures 29 and 30).

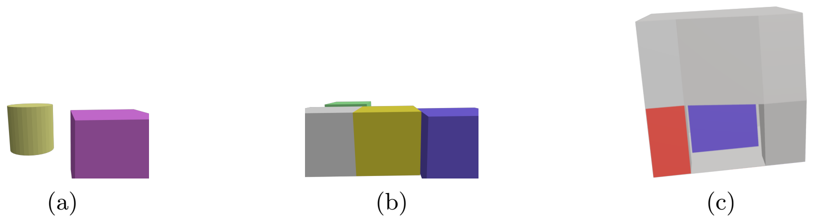


Fig. 28: The Virtual Models Created Using Blender

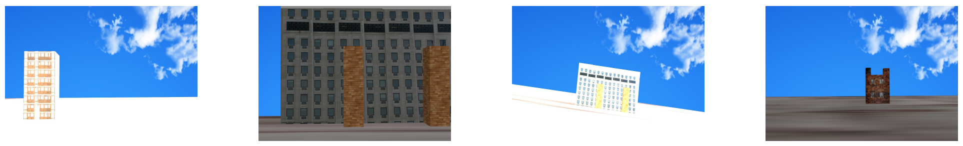


Fig. 29: More complex Virtual Models Created Using Blender

[3] http://www.blender.org/

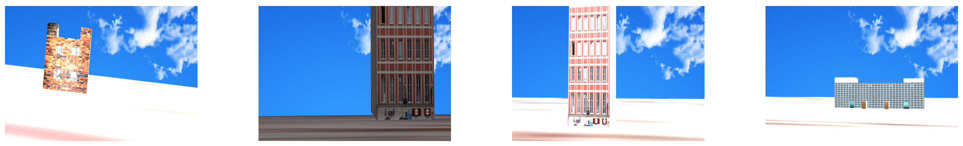
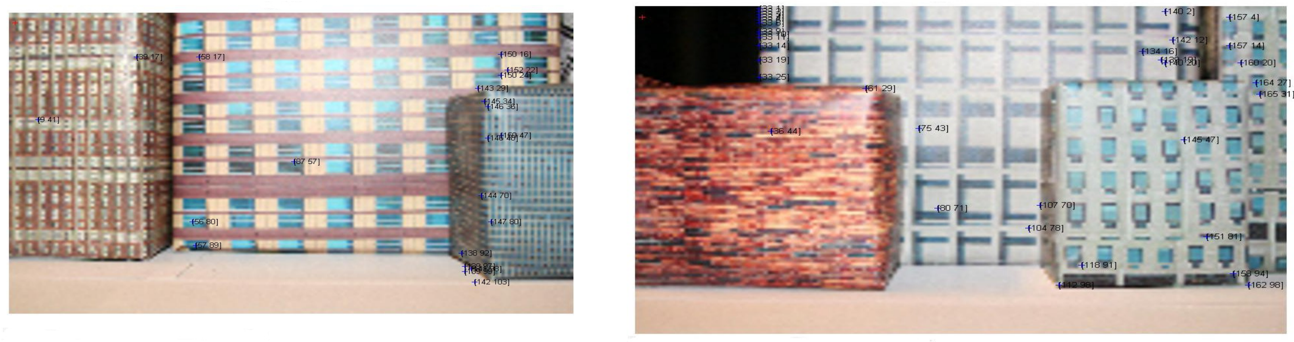


Fig. 30: Different Scales, Orientations and Lightning Conditions Between Training and Test Images

4.2 Performance of Feature Detection

Figure 31 shows the detected feature points on two images of an epipolar volume. We have used the method of selecting starting points of line segments here. We can notice that most corner points have been detected, with other corners probably being detected in neighboring image planes.



(a) (b)

Fig. 31: Detected Features on Two Samples of the Epipolar Volume of the Physical Model

Also, as an example of how affine transformations affect the process of feature detection, notice the figure 32 which shows detected points before and after scaling the scene. We can see that most of the points have been repeated in the scaled scene, despite those points that have disappeared due to decreased resolution. Again we have used the method of selecting starting points of line segments.

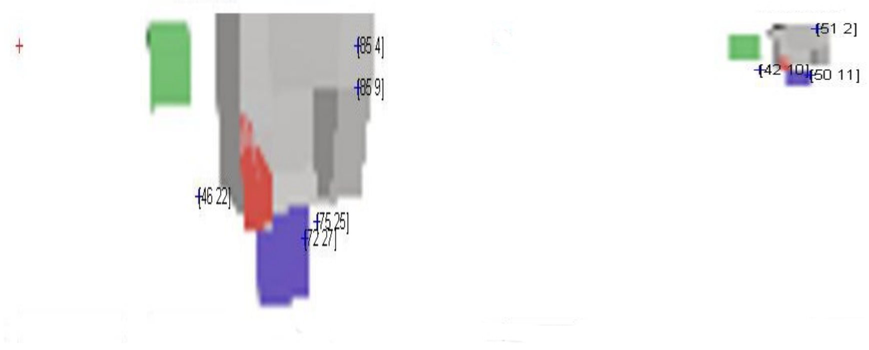
4.3 Feature Description

For feature description in the image plane, we have to choose between two possible paradigms.

The first paradigm which is older, is to use real-valued descriptor vectors, the most popular class among which are SIFT-based methods [27]. These description methods extract a real-valued feature vector from each feature patch. SIFT and its variants are considered accurate and efficient in terms of matching and recognition accuracy. The SURF algorithm [4] is a variant of SIFT which claims to be faster and more appropriate for real-time applications. We will use SURF in the first part of our experiments as the representative for real-valued descriptors.

SIFT and its variants have been widely used for object recognition, image matching and similar tasks [35]. They have proven accurate and efficient in describing image patches. However, since these methods use real-valued vector components they are inefficient for storage, especially for large-scale data [10]. Moreover, the descriptor extraction method utilized by SURF involves many computationally complex steps which make it quite slow to run, especially in real-time tasks.

BRIEF Due to the limitations mentioned above, computer vision researchers have focused in recent years on developing efficiently and easily computable feature description methods which also for the sake of storage efficiently use single bits instead of real-valued vector components.



(a) Original Plane (b) Scaled Plane

Fig. 32: Effect of Scaling on the Detected Points

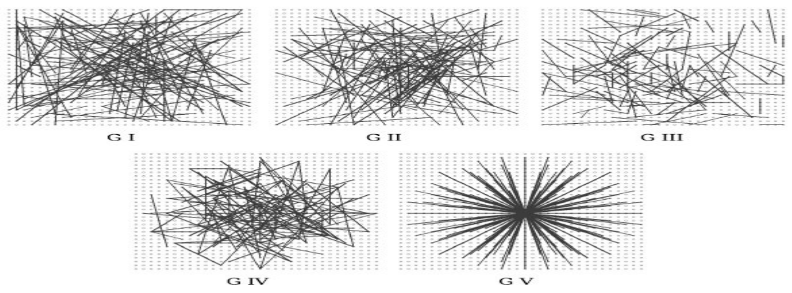


Fig. 33: BRIEF performs a number of randoms intensities tests between different pairs of points in the image patch (a small rectangle around the feature point). This figure shows different possible strategies for selecting pairs of points [10].

The BRIEF method developed in the computer vision laboratory of EPFL is one of the best-known binary descriptors [10]. The goal of BRIEF is simplicity of both vector representation (binary values) and vector construction.

The BRIEF feature patch description is based on the very simple idea of comparing intensity of two randomly selected pixels of the patch and storing the result as a bit. That single bit simply represents whether the intensity of one pixel is higher than the other or not. The complete feature vector is formed by performing a reasonable number of such tests on different pixel pairs in an image patch.

It is evident that intensity comparison tests which are building blocks of BRIEF method, can be executed in a very fast and efficient manner and therefore feature vectors can be formed very quickly. Moreover, since each feature component includes only a single bit, the overall feature vector can very efficiently be stored and retrieved using binary hashing algorithms. These two properties make BRIEF, and binary descriptors in general, very suitable for real-time large-scale applications including our desired image-based localization. Figure 33 gives an idea of how brief works.

The main remaining question regarding BRIEF is how efficient it is, in terms of recognition and matching accuracy. As experiments have shown, BRIEF feature descriptors are not sufficiently invariant against affine transformations, especially rotation. This is not surprising considering the very simple idea behind BRIEF [34]. However, because of speed advantages, it is still desirable for large-scale applications and matching errors are usually tolerable.

ORB Descriptor To overcome the problem of rotation-variance in BRIEF, an extended method called ORB (ORiented Brief) has recently been proposed [34]. ORB claims to be a rotation-invariant version of BRIEF. In ORB, before performing intensity comparison tests, the principal orientation of the patch is found and then the patch is rotated to cancel the effect of that orientation. Finally, the intensity tests are performed on the rotated patch with the hope the effect of orientation is removed. In the experiments explained later, we will use ORB as a representative for binary descriptors.

4.4 Line Detection in a Point Cloud

As we already mentioned in 3.4, the Hough Transform in the two-step approach can be performed much more efficiently, since the number of ”on” points is much less than in the preliminary approach. However, other parts of the Hough's algorithm which are independent of the intensity of image pixels (the two loops dependent on the resolution of ϕ and θ) still take a relatively high computation time. Can we perform better than this?

We try to improve line detection by exploiting the fact that the input to our line detection procedure is indeed a point cloud, not a binary image as we assumed for the Hough Transform. Indeed, we discretize the point cloud (the set of points detected by the initial feature detection algorithm) into a B/W (binary) image in order to run the Hough Transform and detect lines. Maybe this is not the best thing to do.

An alternative approach would be to perform line detection directly on the point cloud, rather than on the synthetic binary image. It means that we have to cluster the point cloud into partitions each containing collinear points. Considering the latter interpretation, we may think of a customized clustering algorithm in which the similarity measure between two arbitrary points is determined based on how well they fit on a line with respect also to other points of the set. The difficulty with this approach is that every two points definitely lie on a line (unless they are the same) so we should take other points into account.

To overcome the aforementioned problem, we can treat line detection as a model fitting problem. The models are linear (i.e. lines) and we have to fit the data (i.e. feature points) to the models. Since we have obviously more than one line per point cloud, we have to perform some kind of multiple model fitting. The question of which point belongs to which model, leads us to the very well-known Expectation-Maximization algorithm [7].

In the most common application of the EM algorithm which is Gaussian Mixture Modeling, a predefined number of Gaussian models are fit to a set of sample points. We customize GMM as a linear mixture modeling approach, in which instead of Gaussian ellipses, we fit the points into lines. The measure of fitness of a point in this case is the distance between the line and the point, in place of the distance of the point to the center of the Gaussian as in the GMM. The rest of the algorithm is exactly the same as the traditional EM, with Expectation and Maximization steps. Such an algorithm has already been studied in the statistics literature as the Regression Mixture Modeling [30].

Using a regression mixture model instead of the Hough Transform for line detection dramatically improves the computational complexity of the feature extraction algorithm. In our experiments, the feature extraction algorithm using RMM executed almost 10 times faster then the same procedure using Hough Transform for line detection.

Figure 34 depicts the pseudo-code for the regression mixture model line detection algorithm. Please note that some regularizations could also be performed while computing the distance, for example in order to avoid having lines with points very far apart. The value of K is a parameter to the algorithm. In our experiments, the optimal value of K is a fraction of the number of point (precisely, we got best results with $K = 0.7N$). Please note that it is not a drawback of RMM compared to Hough Transform, since HT also requires a number of parameters (the threshold is the main one) to be set prior to execution of the algorithm.

4.5 Experiments with SURF

For the first set of experiments, we used SURF algorithm for feature description. Line detection was performed using regression mixture modeling and for vocabulary extraction we used simple k-means clustering with Euclidean distance.

Since we used random initialization, k-means clustering becomes a randomized algorithm and the results slightly differ for each execution. Because of this, we performed 10 repetitions of each experiment and recorded the arithmetic mean of the results.

```
Require: The set $\mathcal{P}$ of points forming the point cloud.
Require: The number of lines $K$
 1: Randomly assign a model $m \in \{1, \ldots, K\}$ to each point $p \in \mathcal{P}$, Thereby partitioning it into $K$ initial
    clusters $\{\mathcal{P}_1, \ldots, \mathcal{P}_k\}$.
 2: repeat
 3:   for each partition $\mathcal{P}_i$, $i \in \{1, \ldots, K\}$ do
 4:     Fit a line to the points in $\mathbb{P}_i$ using some model fitting algorithm (e.g. RANSAC), achieving linear
        model $\mathcal{L}_i$
 5:   end for
 6:   for each point $p \in \mathcal{P}$ do
 7:     Reassign the label $m$ of $p$ as the closest line to $\mathcal{L}_i$,$i \in \{1, \ldots, K\}$ to it.
 8:   end for
 9: until the partitions are not changed anymore
10: return the output lines $\{\mathcal{L}_1, \ldots, \mathcal{L}_k\}$
```

Fig. 34: Line Detection Using Regression Mixture Modeling

The overall results for different vocabulary sizes are depicted in table 1. We can see in this table that, as we expected, using k-means algorithm for vocabulary creation, indeed works and achieves promising results.

Table 1: Recognition Results Obtained by Using SURF Feature Descriptor

Vocabulary Size	**Training Time**	**Test Time**	**Accuracy**
128	48.4	0.1	73.8
256	73.0	0.2	81.4
512	98.5	0.4	82.2

The last remaining question is with regard to the optimal vocabulary size. In small visual vocabularies, the discriminativeness of visual words is highly reduced since one visual word may represent multiple feature points from different classes. On the other hand, having too many visual words degrades the performance too, since repeatability of features is reduced because of the fact that some features may be represented by multiple visual words. Therefore, the optimal vocabulary size lies somewhere between these two extremes.

Hierarchical Clustering We see in table 1 that the main problem with k-means vocabulary creation is its infeasible computational complexity. Indeed, we can not easily test vocabulary sizes of more than 256 words on a moderate computer due to the increased training time (recall that training time includes the time required for vocabulary creation).

To improve time complexity of vocabulary creation, we use the idea of vocabulary trees [22] in a simple way which allows us to perform clustering on a small fixed number of clusters (we chose 2 for our experiments for reasons which will be clarified later), but achieving arbitrarily large vocabulary sizes (to be more precise, any power of 2).

In this approach, we execute k-means clusterings with $k = 2$ many times (instead of only once), but progressively on fewer number of data samples. We first cluster the complete data set into two clusters, yields two disjoint sets of data with smaller (ideally half) size than the original set. Each of these smaller data sets is itself consecutively divided into two clusters and this division process continues until a predefined depth L is achieved. Using this approach, we can achieve a vocabulary of size 2^L but with computational complexity much lower than that of running a 2^L-means clustering algorithm.

Using this approach, A complete binary tree of depth L is formed whose leaves represent the final clusters (visual words) [33]. To get an idea of how tree-based hierarchical clustering works, let's take a look at how it runs for the simple case of $L = 2$. It means that finally we will have 4 clusters.

Suppose we have the initial dataset D of $card(D)$ samples. The process is as follows:

1. Apply a 2-means clustering to D, partitioning D into two datasets D_0 and D_1. It will be the level 1 of the tree. Note that $card(D_0) + card(D_1) = card(D)$.
2. Apply a 2-means clustering to D_0, partitioning it into two datasets D_{00} and D_{01}. It will be the level 1 of the tree.
3. Apply a 2-means clustering to D_1, partitioning it into two datasets D_{10} and D_{11}. It will be the level 1 of the tree.
4. Steps 2 and 2 for the second level in our clustering tree.
5. Now, The four data sets D_00,D_01,D_10 and D_11 are the desired data clustered.

Using a branching factor of 2 for the cluster tree is a natural choice since it allows easier encoding of final obtained cluster.

The results of using hierarchical instead of flat clustering for vocabulary creation is depicted in table 2. It can be easily verified that this approach dramatically improves time complexity of training procedure.

Table 2: Using Hierarchical Clustering in SURF-Based Recognition

Vocabulary Size	**Training Time**	**Test Time**	**Accuracy**
128	2.2	0.1	73.2
256	4.0	0.3	81.6
512	5.7	0.4	81.9
1024	9.2	0.7	82.1
2048	15.2	1.3	82.0

We can also infer from table 2 that the optimal vocabulary size for our experiments is 1024.

4.6 Experiments with ORB Feature Description

As we already mentioned in 4.3, SURF feature description is not suitable for real-time and large-scale applications. Therefore we test our feature extraction algorithm also with the ORB method.

We already saw that ORB produces binary features. Therefore it seems reasonable to use Hamming distance which measures distance between two bit strings, as the similarity distance used by k-means, instead of Euclidean distance. The results of using ORB feature description with k-means and Hamming distance is depicted in table 3.

Table 3: Recognition Results for ORB-Based feature Description and K-Means Vocabulary Creation

Vocabulary Size	**Training Time**	**Test Time**	**Accuracy**
256	70.4	0.2	59.5
512	91.7	0.5	64.9
1024	130.5	0.9	67.1

What we can see in table 3 is that using ORB with k-means yields poor results. The reason would probably lie behind the vocabulary creation method and the binary nature of description vectors. Notice that the underlying assumption behind k-means is that data dimensions follow a Gaussian distribution (although this assumption applies for euclidean distance and using Hamming distance slightly violates it) and that data dimensions are independent. However, Gaussian distribution is not an appropriate model for binary data.

Because of the fact mentioned above, an alternative idea would to use EM algorithm but with a more appropriate model assumption. A suitable statical model for binary data is the Bernoulli distribution. Using Bernoulli distribution to model data, we develop a Bernoulli Mixture Modeling (BMM) algorithm [23] which fits data to a mixture of multi-dimensional Bernoulli distributions. Figure 35 depicts the pseudo-code for the BMM algorithm.

```
Require: The set $\mathcal{D}$ of feature vectors.
Require: The number of visual words (models) $K$
 1: Randomly assign a model $m \in \{1, \dots, K\}$ to each vector $v \in \mathcal{D}$, Thereby partitioning it into $K$ initial
    clusters $\{\mathcal{D}_1, \dots, \mathcal{D}_k\}$.
 2: repeat
 3:   for each partition $\mathcal{D}_i$, $i \in \{1, \dots, K\}$ do
 4:     Fit a Bernoulli model to the vectors in $\mathbb{D}_i$ by averaging all the vectors in $\mathbb{D}_i$, achieving mean vectors
        $M_i$.
 5:   end for
 6:   for each point $p \in \mathcal{P}$ do
 7:     Reassign the label $m$ of $p$ as $m = \arg\max_i \in \{1, \dots, K\} \sum_j \log[M_{ij}^{p_j}(1 - M_{ij})^{1-p_j}]$.
 8:   end for
 9: until the clusters are not changed anymore
10: return the means of clusters $\{\mathcal{M}_1, \dots, \mathcal{M}_k\}$ as visual words.
```

Fig. 35: Clustering by Bernoulli Mixture Models

Replacing BMM for k-means in the vocabulary creation algorithm, we repeat the experiments. Results are shown in table 4.

Table 4: Recognition Results for ORB-Based feature Description and BMM Vocabulary Creation

Vocabulary Size	Training Time	Test Time	Accuracy
256	70.4	0.2	59.7
512	91.7	0.5	64.6
1024	130.5	0.9	67.0

What we can infer from table 4 is that results are still unsatisfactory. We may conclude from this fact that independence assumption of features in BMM is largely violated in ORB-created binary feature description vectors. Therefore, we should change our vocabulary creation method in a more dramatic way.

Vocabulary Creation Based on Exact Matching The reason that we can not achieve good results with k-means for ORB probably lies in the different nature and properties of data. SURF-generated feature descriptors are highly redundant and spread in specific regions of the feature space. We exploit this redundancy in the k-means algorithms to create fine-grained visual words.

However, as our experiments with k-means and ORB show, the feature vectors created by ORB are much less redundant and therefore we can not expect meaningful clusters originating from them. Maybe this is the reason behind k-means' fail.

Moreover, the principal motivation for applying k-means to the SURF descriptors is quantizing the vector space so that we can have meaningful visual words, in analogy with textual words (hence the name vector quantization). But ORB feature vectors are binary strings and are therefore, already quantized. It may look unnecessary to perform another quantization step.

For the reasons explained above, the idea which comes to mind is that for ORB features, we do not perform clustering at all. Each uniqe bit string can be considered a visual word by itself. We construct a visual vocabulary this way.

Although the number of visual words will be large in this approach, since we are only seeking identical bit strings, we can easily and efficiently perform matching using a hash table. All we need is a hash function which maps bit strings to binary values. The simplest such function would be the decimal value of the binary number represented by the bit string.

Our experiments showed that on average, 80% of feature vectors are unique. Therefore, for a total data set of 24158 feature vectors, a vocabulary of size 20634 is formed which is quite large. We will later see what problems this large vocabulary size causes and how we can solve them.

The results of using ORB feature descriptors and exact matching-based vocabulary creation is shown in table 5.

Table 5: Recognition Results for ORB and Matching-Based Vocabulary Creation

Vocabulary Size	**Training Time**	**Test Time**	**Accuracy**
20634	11.2	10.1	77.5

We can see in table 5 that now, results are quite satisfactory in terms of recognition rate. However, accuracy is not as good as the SURF-based approach. This could be predictable since, as we already mentioned, quality of ORB descriptors is not as good as SURF. After all, ORB uses a much simpler extraction algorithm which allows feature extraction with very low time complexity.

Another issue we may notice in table 5 is that the time required for testing is quite high. This is expectable since the number of visual words is quite large in this case. However, this is extremely undesirable since in our application, testing is performed on users' cellphones which are limited in computational resources and in fact, testing is the part of the system which has to be run real-time and online. Therefore, a high testing time makes the algorithm far from practical. In fact, we would prefer training to take longer (since it will be run once and offline) if it would make testing run faster.

What can we do to improve testing time? Definitely we can not make the vocabulary smaller, since it is in contrast with our idea of vocabulary creation based on exact matching and the size of the vocabulary is only determined by the properties of the descriptors' dataset.

The second thing we can do is to that, instead of reducing the size of the whole feature set, we make each feature smaller by reducing its dimensions (The original size of ORB vectors is 256). However, we can not use traditional dimensionality reduction algorithms like PCA. First reason is the huge number of feature vectors which makes the algorithms take a very long time to run. The second and more important reason is that data are binary and therefore underlying assumptions of traditional dimensionality reductions algorithms does not hold for them. This makes the algorithms impractical for this task.

Instead of dimensionality reduction, we can think about the simpler idea of feature selection to reduce the number of vector components. In this case, we select a subset of vector components with the hope that overall characteristics of vectors is preserved or at least does not change too much in the new space.

Table 6: Effect of Reducing the Number of Features on the Number of Unique Elements in the set

Descriptor Size	**Number of Unique Elements**	**Total Number of Elements**
256	20634	24158
128	20781	24158
64	20891	24158

Table 6 shows that this is indeed true, even for random feature selection. We can see in this table that the number of unique vectors in the set does not change significantly if we retain only a subset of the elements of a vector. We can use this desirable property to dramatically improve the time of testing.

We repeated the ORB Hash-Based experiment, but this time with various reduced vector sizes. The results are summarized in table 7

Table 7: Recognition Results Obtained Using Reduced-Size ORB Feature Vectors

Descriptor Size	**Training Time**	**Test Time**	**Accuracy**
256	11.2	10.1	77.5
128	10.7	4.9	77.1
64	9.5	2.3	76.7

We can see in table 7 that by reducing cardinality of descriptor vectors, the testing time improves significantly whereas the accuracy is only very slightly sacrificed. This time complexity could be more reasonable for a real-time application such as image-based localization.

4.7 Avoiding the Use of Plenoptic Information

To get and idea of what would happen if we did not use information of the plenoptic space, we avoided any plenoptic post-processing and repeated the recognition experiment with ORB and hash-based vocabulary. For this, we simply added up all extracted features from images of the plenoptic sequence into an overall feature vector set for each model, without any further processing (line detection,...). The results are summarized in table 8.

Table 8: Recognition Results Obtained Without Using Plenoptic Information

Vocabulary Size	**Training Time**	**Test Time**	**Accuracy**
56724	19.2	32.1	68.7

We can see in the table above that the vocabulary size largely increases in the case of non-plenoptic recognition. This is because plenoptic processing indeed tries to remove redundant features and thereby reduce feature set size, without losing much information. Without plenoptic processing, we are left with the initial larger feature set.

Having a larger feature set has negative effects on training time, testing time and also on accuracy. The effect on testing time is obvious, the more feature sets, the larger vocabulary and a larger vocabulary means more number of comparison at the time of testing.

At the training time, having a large feature set increases the time for vocabulary creation, especially when using clustering-based methods like k-means.

Finally, the accuracy is also negatively affected when plenoptic information are not used. This is because with constraint that we put of epipolar lines while choosing feature trajectories, most of the noisy and inaccurate features are removed. Equivalently, if we do not use plenoptic information, many noisy and incorrect features are introduced to the vocabulary which degrade the recognition performance of the algorithm.

5 Conclusion and Future Works

We proposed and discussed a feature extraction algorithm which is based on exploiting the information contained in the plenoptic function and redundancy of multi-view images. Aggregating all parts of a visual recognition system, we proposed to perform location recognition on a mobile phone, based on the plenoptic function's realizations.

For future improvements of the proposed algorithm, we may improve line detection by using more accurate single-line fitting methods. The prior information on epipolar lines can be used to remove noisy detections. Moreover, we can reduce the set of extracted features by enforcing a minimum distance between every two extracted feature points in the initial or final extracted features.

As a more general improvement, we can look for an image transform which maps the plenoptic space to a new, sparse space in which only informative feature points are retained. A discrete wavelet transform (with a customized mother wavelet) could be a starting point.

In this work, we assumed that different locations can be partitioned into non-overlapping models. However, for neighboring locations (buildings) this assumption may not be completely true. In a single photo of a plenoptic sequence, parts of multiple buildings may be present. This phenomenon could have negative effects on recognition performance.

To overcome this problem, an idea would be to use an overall model for a set of neighboring locations, rather than one model for each specific location. In other words, we may use a temporal model for relating neighboring locations to each others. The very well-known Hidden Markov model (HMM) [7] is a natural solution for this problem. Locations information (e.g. latitude and longitude) could be the states of the model and the visual information could be the observations. For example, the bag of words vector extracted from a single image could be used as an observation vector.

Using a Hidden Markov Model to encode locations and their neighborhood has the extra advantage that we can easily encode the topology of the urban environment in the state transition probabilities and therefore have a fusion between visual and geographical information in a very principled way.

More preprocessing and post-processing steps can also be used to improve the algorithm. For example we may apply non-maximal suppression to the extracted points.

As a conclusion, we saw in this report that image retrieval in the plenoptic space instead of the image space, can have positive effects on the accuracy of the final systems. This is because multiple views of the same scene have more information than only a single photo. To exploit these information successfully and efficiently, we have to use the regularity of the plenoptic space which allows us to aggregately analyze multi-view images.

Working in this new space requires developing novel methods and tools (transforms, feature detectors,...) which are optimized specifically for it and maximally exploit plenoptic space information.

References

1. Adelson, E., and Bergen, J. The plenoptic function and the elements of early vision. *Computational models of visual processing 1* (1991), 3–20.
2. Aman, J., Yao, J., and Summers, R. Content-based image retrieval on ct colonography using rotation and scale invariant features and bag-of-words model. In *Biomedical Imaging: From Nano to Macro, 2010 IEEE International Symposium on* (2010), IEEE, pp. 1357–1360.
3. Apostoloff, N., and Fitzgibbon, A. Automatic video segmentation using spatiotemporal t-junctions. In *Proc. BMVC* (2006).
4. Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. Speeded-up robust features (surf). *Computer Vision and Image Understanding 110*, 3 (2008), 346–359.
5. Berent, J., and Dragotti, P. Segmentation of epipolar-plane image volumes with occlusion and disocclusion competition. In *Multimedia Signal Processing, 2006 IEEE 8th Workshop on* (2006), IEEE, pp. 182–185.
6. Berent, J., and Dragotti, P. Plenoptic manifolds. *Signal Processing Magazine, IEEE 24*, 6 (2007), 34–44.
7. Bishop, C., and en ligne), S. S. *Pattern recognition and machine learning*, vol. 4. springer New York, 2006.
8. Bolles, R., Baker, H., and Marimont, D. Epipolar-plane image analysis: An approach to determining structure from motion. *International Journal of Computer Vision 1*, 1 (1987), 7–55.
9. Bregonzio, M., Gong, S., and Xiang, T. Recognising action as clouds of space-time interest points. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on* (2009), IEEE, pp. 1948–1955.
10. Calonder, M., Lepetit, V., Strecha, C., and Fua, P. Brief: Binary robust independent elementary features. *Computer Vision–ECCV 2010* (2010), 778–792.
11. Chen, D., Baatz, G., Koser, K., Tsai, S., Vedantham, R., Pylvanainen, T., Roimela, K., Chen, X., Bach, J., Pollefeys, M., et al. City-scale landmark identification on mobile devices. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on* (2011), IEEE, pp. 737–744.
12. Cheung, W., and Hamarneh, G. N-sift: N-dimensional scale invariant feature transform for matching medical images. In *Biomedical Imaging: From Nano to Macro, 2007. ISBI 2007. 4th IEEE International Symposium on* (2007), IEEE, pp. 720–723.
13. Criminisi, A., Kang, S., Swaminathan, R., Szeliski, R., and Anandan, P. Extracting layers and analyzing their specular properties using epipolar-plane-image analysis. *Computer vision and image understanding 97*, 1 (2005), 51–85.

14. Fan, L., Yu, X., Shu, Z., and Zhang, Q. Multi-view object segmentation based on epipolar plane image analysis. In *Computer Science and Computational Technology, 2008. ISCSCT'08. International Symposium on* (2008), vol. 2, IEEE, pp. 602–605.
15. Flora, G., and Zheng, J. Adjusting route panoramas with condensed image slices. In *Proceedings of the 15th international conference on Multimedia* (2007), ACM, pp. 815–818.
16. Gelman, A., Berent, J., and Dragotti, P. Layer-based sparse representation of multiview images. *EURASIP Journal on Advances in Signal Processing 2012*, 1 (2012), 61.
17. Hartley, R., Zisserman, A., and ebrary, I. *Multiple view geometry in computer vision*, vol. 2. Cambridge Univ Press, 2003.
18. Helmer, S., and Lowe, D. Using stereo for object recognition. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on* (2010), IEEE, pp. 3121–3127.
19. Helmer, S., Meger, D., Muja, M., Little, J., and Lowe, D. Multiple viewpoint recognition and localization. *Computer Vision–ACCV 2010* (2011), 464–477.
20. Hirahara, K., and Ikeuchi, K. Panoramic-view and epipolarplane image understandings for street-parking vehicle detection. In *ITS Symposium* (2003).
21. Ishibashi, T., Yendo, T., Tehrani, M., Fujii, T., and Tanimoto, M. 3d space representation using epipolar plane depth image. In *Picture Coding Symposium (PCS), 2010* (2010), IEEE, pp. 22–25.
22. Jain, A., and Dubes, R. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.
23. Juan, A., and Vidal, E. Bernoulli mixture models for binary images. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on* (2004), vol. 3, IEEE, pp. 367–370.
24. Knopp, J., Prasad, M., and Van Gool, L. Orientation invariant 3d object classification using hough transform based methods. In *Proceedings of the ACM workshop on 3D object retrieval* (2010), ACM, pp. 15–20.
25. Knopp, J., Prasad, M., Willems, G., Timofte, R., and Van Gool, L. Hough transform and 3d surf for robust three dimensional classification. *Computer Vision–ECCV 2010* (2010), 589–602.
26. Laptev, I. On space-time interest points. *International Journal of Computer Vision 64*, 2 (2005), 107–123.
27. Lowe, D. Distinctive image features from scale-invariant keypoints. *International journal of computer vision 60*, 2 (2004), 91–110.
28. Matousek, M., and Hlaváс, V. Epipolar plane images as a tool to seek correspondences in a dense sequence. In *CAIP* (2003), N. Petkov and M. A. Westenberg, Eds., vol. 2756 of *Lecture Notes in Computer Science*, Springer, pp. 74–81.
29. Mellor, J., Teller, S., and Lozano-Pérez, T. Dense depth maps from epipolar images.
30. Muthén, B., and Asparouhov, T. Multilevel regression mixture analysis. *Journal of the Royal Statistical Society: Series A (Statistics in Society) 172*, 3 (2009), 639–657.
31. Neumann, J., and Fermüller, C. Plenoptic video geometry. *The Visual Computer 19*, 6 (2003), 395–404.
32. Ni, D., Qu, Y., Yang, X., Chui, Y., Wong, T., Ho, S., and Heng, P. Volumetric ultrasound panorama based on 3d sift. *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2008* (2008), 52–60.
33. Nister, D., and Stewenius, H. Scalable recognition with a vocabulary tree. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on* (2006), vol. 2, Ieee, pp. 2161–2168.
34. Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. Orb: an efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on* (2011), IEEE, pp. 2564–2571.
35. Schmid, C., Mohr, R., and Bauckhage, C. Evaluation of interest point detectors. *International Journal of computer vision 37*, 2 (2000), 151–172.
36. Shapiro, L., and Stockman, G. Computer vision. 2001, 2001.
37. Stich, T., Tevs, A., and Magnor, M. Global depth from epipolar volumes-a general framework for reconstructing non-lambertian surfaces. In *Proceedings of Third International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)* (2006), Citeseer, pp. 1–8.
38. Wang, Y., Brookes, M., and Dragotti, P. Object recognition using multi-view imaging. In *Signal Processing, 2008. ICSP 2008. 9th International Conference on* (2008), IEEE, pp. 810–813.
39. Willems, G., Tuytelaars, T., and Van Gool, L. An efficient dense and scale-invariant spatio-temporal interest point detector. *Computer Vision–ECCV 2008* (2008), 650–663.
40. Yang, A., Maji, S., Christoudias, C., Darrell, T., Malik, J., and Sastry, S. Multiple-view object recognition in smart camera networks. *Distributed Video Sensor Networks* (2011), 55–68.
41. Yu, T., Woodford, O., and Cipolla, R. An evaluation of volumetric interest points. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on* (2011), IEEE, pp. 282–289.
42. Zhu, Z., Xu, G., and Lin, X. Efficient fourier-based approach for detecting orientations and occlusions in epipolar plane images for 3d scene modeling. *International journal of computer vision 61*, 3 (2005), 233–258.