

Master Thesis

Machine vision based detection and
size measurement of objects on a
conveyor belt

Md Rezaul Kabir Tutul
17th September 2018

Supervisors:

Prof. Dr. Heinrich Müller

Dr.-Ing. André ibisch

Lehrstuhl Informatik VII
Graphische Systeme
TU Dortmund

Contents

1	Introduction	4
1.1	Problem	4
1.2	Contributions	5
1.3	Organization	5
2	Experimental setup and solution outline	6
2.1	Experimental setup	6
2.2	Outline of solution	7
3	Camera calibration	8
3.1	Camera model	8
3.1.1	Pinhole camera model	8
3.1.2	Extrinsic parameters	9
3.1.3	Intrinsic parameters	9
3.1.4	Lens distortion	10
3.2	Calibration methods	11
3.2.1	Faugeras-Toscani method	11
3.2.2	Zhang method	11
3.2.3	Robust calibration	11
3.3	Experimental camera calibration procedure	12
3.3.1	Image preparation	12
3.3.2	Image addition and calibration	13
3.3.3	Examine reprojection error	14
4	Object detection	15
4.1	Segmentation	15
4.1.1	Threshold based method	15
4.1.2	Edge based method	16
4.1.3	Region based method	17
4.1.4	Clustering based segmentation method	17
4.1.5	Watershed based methods	18
4.2	Background subtraction	18
4.2.1	Gaussian mixture model	19
4.2.2	Expectation maximization	20
4.3	Experimental result of object detection	20

5	Feature detection and description	23
5.1	Minimum eigenvalue algorithm	23
5.2	Feature tracking	24
5.3	Experimental result of feature detection and tracking	26
6	Multiview geometry	28
6.1	Epipolar geometry	28
6.1.1	Epipolar constraint	29
6.1.2	Fundamental matrix	30
6.1.3	Triangulation	32
7	Measurement analysis	34
7.1	Technique 1	34
7.2	Technique 2	40
8	Conclusion	44

List of Figures

1	Experimental Setup	6
2	Outline flowchart	7
3	Used calibration pattern	13
4	Detected objects from undistorted frame	22
5	Feature detection and tracking	27
6	Epipolar geometry	28
7	Point cloud using triangulation	32
8	Original and undistorted frame	34
9	The labeled objects	34
10	The center point and bounding box of each object	35
11	The region of interest	35
12	The features (a), match features (b), and epipolar inliers (c) .	36
13	The center points and the 3D match points from the top and side view	37
14	The real-time object dimension measurement	38
15	The segmented color objects	40
16	The feature detection and tracking	40
17	Original and clustered point cloud	41
18	The cuboid over each clustered point cloud	41
19	Object dimension measurement	42
20	Real time measurement using cuboid over the point cloud . .	43

1 Introduction

Nowadays measuring three-dimensional object automatically is in demand at places which handle a large number of objects such as couriers, freight handlers and airlines companies. Faster and more precise measurements are the requirement of such companies. The application of machine vision based measurement in the industry is expanding due to recent progress in computer hardware and computer vision algorithms. There are numerous known techniques to achieve 3D measurement and reconstruction of different objects. One of the most popular approaches is the structure from motion that has been implemented in this thesis to recover the 3D information of a scene for the measurement purpose.

The field of segmentation concentrates on detecting objects in the images and videos with the help of computer system. Researchers have achieved notable progress in detecting the objects from images and videos, but detecting the objects in natural scenes is still a young topic for research. The object detection in a natural scene is difficult because of the variation in the background and lighting condition. Over the past decade, researchers have invested much time and effort to solve these challenges. To explain the variability present in the natural scene and to solve the object detection problem in the unconstrained images and videos are an extremely difficult challenge. In a video, there are two sources of information that can be used to detect and track the objects. The sources are visual features and motion information. In this thesis, motion information has been used to detect, label and reconstruct the objects on a conveyor belt. Then the measurement has been taken with the help of recovered 3D and pixel information covered by each object on the conveyor belt.

1.1 Problem

The objective of this thesis is to design and develop a vision-based system using a fixed single camera in order to accomplish the following goals:

- Camera calibration
- Arbitrary object detection on the conveyor belt
- Measure the sizes of detected objects
- Visualize the measured values of objects

1.2 Contributions

The contributions of this thesis as follows:

- Reliable camera calibration procedure using the *Zhang* method
- Detection and labeling of arbitrary moving objects on the conveyor belt
- 3D information extraction from the objects motion
- Measuring and visualizing of the labeled object sizes

1.3 Organization

The remainder of this thesis has been organized as follows:

In chapter 2 the solution outline and experimental setup have been described. In chapter 3 the camera calibration methods, the theory of pinhole and fish-eye camera model, the theory behind extrinsic, intrinsic and lens distortion parameter estimation and the procedure or analysis of camera calibration that is performed in this thesis have been presented. In chapter 4 the segmentation methods, the theory behind background subtraction models, the experimental result after background subtraction and morphological operations have been described. In chapter 5 the fundamental of features, the theory of feature detection, description and tracking method with the experimental result have been illustrated. In chapter 6 the two view geometry, the theory for the estimation of fundamental matrix and the triangulation procedure have been explained. In chapter 7 the two techniques of the 3D object measurement from the conveyor belt that are performed in this thesis and the comparison between the results of these techniques have been explained stepwise. In chapter 8 the problems and difficulties during the experiment has been described as a conclusion of this thesis.

2 Experimental setup and solution outline

2.1 Experimental setup

The experimental setup consists of a conveyor belt containing objects, a single fixed camera mounted to the conveyor belt and a computer or machine that has Intel Corei3 processor with 2.4 GHz speed and build in Intel graphics. These specifications of a computer or machine are not sufficient for this kind of experiment. At least 3GHz processor speed and extra graphics card of a machine are recommended. In this experiment, *Basler acA1920-40gc* camera has been used that has $6mm$ focal length, $11.3 \times 7.1mm$ sensor size and 86.7° horizontal angle of view. The camera calibration has been done by moving the camera around the fixed cheese-board pattern. The distance between the camera and object surface has been kept approximately 150cm. It is recommended to place the camera in parallel to the object surface or conveyor belt, which provides more accurate measurements but in this experiment, the camera has not been kept perfectly parallel to the object surface because of some equipment problem. The measurement of the Z dimension and the consistent 3D reconstruction can be affected because of the long distance between the camera and object surface. Figure 1 shows the experimental setup.



Figure 1: Experimental Setup

2.2 Outline of solution

Figure 2 shows the pipeline of solution that has been followed to solve the measurement problems, where the camera calibration step determines the extrinsic and intrinsic parameters. These parameters remove the lens distortion and give the focal length of the camera in a pixel unit. Only interested objects have been detected from the conveyor belt and measured their X, Y dimensions in the object detection step. From the numerous techniques of automatic feature detection, the minimum eigenvalue algorithm detects and describes the features of interesting objects in the feature detection and description step. Then the corresponding features of the detected features have been tracked in virtually created second view of the camera using the *Lucas-Kanade* method in the corresponding feature tracking step. 2D coordinates of the detected and tracked features have been triangulated into the 3D coordinates based on the camera projection matrices, which includes translation and rotation of the cameras. Finally, in the 3D reconstruction by triangulation step, the 3D feature points represent the sparse 3D structure of the scene. After recovering 3D information the Z dimension of all detected objects have been measured in the measurement step. These steps have been elaborately described in further chapters.

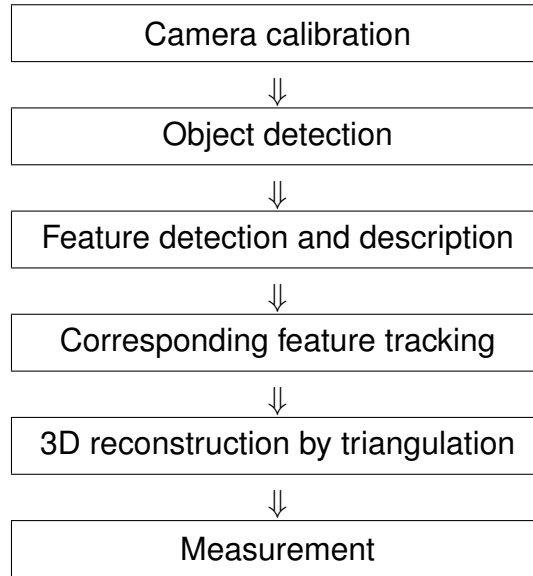


Figure 2: Outline flowchart

3 Camera calibration

The camera calibration is known as camera resectioning that estimates the distortion parameter for the lens and the extrinsic, intrinsic parameter for the image sensor of a camera. These parameters are used to correct the lens distortion, measure the size of objects in the world unit or to determine the camera position in a scene. The camera calibration is used in application such as machine vision to detect and measure the objects. It is also used in robotics for the navigation system and the 3D reconstruction purpose.

The camera parameter consists of intrinsic, extrinsic and distortion coefficients. The 3D world points and their corresponding 2D image points are required to estimate the camera parameter. The correspondences are achieved using the multiple images of calibration pattern such as the chess-board pattern. These correspondences are used to solve the camera parameter.

3.1 Camera model

The camera model includes pinhole camera model and lens distortion [23]. An ideal pinhole camera does not have lens and it also does not account lens distortion.

3.1.1 Pinhole camera model

The pinhole camera is a very simple camera with a small aperture and without the lens. The principle of the pinhole camera is, the infinitely small pinhole ensures that exactly single light ray originating from a point in a scene falls onto the corresponding point in the image plane. Despite its simplicity, the pinhole model often provides an acceptable approximation in the machine vision system. The pinhole camera defines the central perspective projection model. The perspective projection creates inverted images. The apparent size of the objects in the image is defined by their distance to the camera.

The pinhole camera parameter returns 4-by-3 matrix known as a camera matrix or perspective projection matrix. The 3D world points are mapped into the 2D image points using this matrix. The calibration algorithm recovers the camera matrix using the intrinsic and extrinsic parameters [1].

The camera position in a 3D scene is represented by the extrinsic parameter, on the other hand, the intrinsic parameter represent the optical center and camera focal length. The extrinsic parameter transforms the world points into the camera coordinates and the intrinsic parameter maps the camera coordinates into the image plane. The calculation of camera matrix can be shown as:

$$W \cdot [x \ y \ 1] = [X \ Y \ Z \ 1] \cdot P \quad (1)$$

$$P = \begin{bmatrix} R \\ t \end{bmatrix} \cdot K \quad (2)$$

where, the W is the scale factor, the $[x \ y \ 1]$ is the image point, the $[X \ Y \ Z \ 1]$ is the world point, the P is the camera matrix, the R and t are the extrinsic rotation and translation respectively, and the K is the camera intrinsic.

3.1.2 Extrinsic parameters

The rotation R and translation t describe by the extrinsic parameter where the R is a 3-by-3 and t is a 1-by-3 matrix. The origin of the camera coordinate system is at its optical center and its X_c and Y_c axis define the image plane. The transformation from the world points to camera points can be written as:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = R \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} + t \quad (3)$$

where the $[X_c, Y_c, Z_c]$ is the camera coordinate of a point. The R and t are the rotation matrix and translation vector, and the $[X_w, Y_w, Z_w]$ is the world coordinate of the same point.

3.1.3 Intrinsic parameters

The intrinsic parameter consists of the focal length, the optical center that is known as principal point and the skew coefficient. The camera intrinsic can be defined by the following equation.

$$\begin{bmatrix} f_x & 0 & 0 \\ s & f_y & 0 \\ c_x & c_y & 1 \end{bmatrix} \quad (4)$$

where the $[c_x \ c_y]$ is the optical center or principle point and the $[f_x \ f_y]$ is the focal length in pixel unit. The skew coefficient s is non zero if the image axes are not perpendicular. The skew $s = \tan\alpha$.

The intrinsic parameter for the fish-eye camera includes the polynomial mapping coefficients of the projection function. The alignment coefficients are related to the sensor alignment and the transformation from the sensor plane to a pixel location in the camera image plane. Following equation can be used to map an image point into its corresponding 3D vector.

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \lambda \begin{bmatrix} u \\ v \\ a_0 + a_2\rho^2 + a_3\rho^3 + a_4\rho^4 \end{bmatrix} \quad (5)$$

where the (u, v) are the ideal image projection of the world points, the λ defines the scale factor, the a_0, a_2, a_3, a_4 represent the polinomial coefficients of the *Scaramuzza* model where the $a_1 = 0$ and the $\rho = \sqrt{u^2 + v^2}$ [18].

3.1.4 Lens distortion

The pinhole camera can not gather enough light from the scene that known as the major disadvantage of a pinhole camera. The lens can be used to gather more light from the scene and keep the image in focus. The camera model includes radial and tangential lens distortion to accurately model a real camera.

When the light rays bend more near the edges of a lens than they do at its optical center, then the radial distortion occurs. The smaller the lens, the greater the distortion. The radial distortion coefficients can be obtained using the following equation.

$$x_{undistorted} = x(1 + k_1r^2 + k_2r^4 + k_3r^6) \quad (6)$$

$$y_{undistorted} = y(1 + k_1r^2 + k_2r^4 + k_3r^6) \quad (7)$$

The pixel point at (x, y) coordinate in the input image represents the undistorted point $(x_{undistorted}, y_{undistorted})$ at output image. The radial distortion manifest the presence of barrel or fish-eye effect of camera lens. The $[k_1 \ k_2 \ k_3]$ are the radial distortion coefficients of lens.

The tangential distortion appears when the image taking lenses are not perfectly parallel to the image plane. The tangential distortions can be corrected using the following equations.

$$x_{undistorted} = x + [2p_1xy + p_2(r^2 + 2x^2)] \quad (8)$$

$$y_{undistorted} = y + [p_1(r^2 + 2y^2) + 2p_2xy] \quad (9)$$

3.2 Calibration methods

The widely recognized camera calibration methods are the method by *Zhang*, robust calibration and the *Faugeras-Toscani* method [23, 17, 9].

3.2.1 Faugeras-Toscani method

The *Faugeras-Toscani* camera calibration method is based on the estimation of the perspective projection matrix using a nonplanar image pattern [9]. Linear and non-linear; both approaches can be used by this method. At least six non-coplanar feature 3D points need to be automatically or manually detected.

3.2.2 Zhang method

At least two views of planner pattern are required for *Zhang* calibration algorithm. More views of planner pattern need to be used to obtain the more accurate calibration result. Ten to twenty planner patterns or chess-board patterns are required to return the 3D information of a scene using this method.

3.2.3 Robust calibration

The three-dimensional cube with different color faces can be used as a pattern in this calibration method. A manual selection of the two adjacent color faces on the acquired image allows the system to detect automatically the six vertices associated to these faces and thus initialize the perspective projection

matrix using the *Faugeras-Toscani* algorithm. By minimizing the distance between the projected cube edges and the image contours, the refinement of the perspective projection matrix estimation can be achieved.

3.3 Experimental camera calibration procedure

The *MATLAB* single camera calibration application has been used to calibrate the experimental camera. This application follows the *Zhang* [23] calibration algorithm. The steps that have been followed to calibrate the experimental camera given below.

3.3.1 Image preparation

The chess-board images have been used as a calibration pattern which is a very convenient calibration target. The chess-board which contains one side an even number of squares and other side an odd number of squares has been used. The chess-board pattern has been attached with a flat surface. Imperfections on the surface can affect the accuracy of the calibration. The camera has been kept completely focused on the calibration pattern. Autofocus should not use and the distance between the camera and calibration pattern has been kept approximately same as the distance between the interested object and the camera. Image modification and changing autofocus or zoom can return the unexpected re-projection error. A short video of a chessboard pattern has been taken from the different angle of view by moving the camera and all the video frames have been extracted in the PNG format. The PNG format of pattern gives less re-projection error instead of JPEG or other image formats. All the best viewed chess-board patterns have been shorted. Then the calibration images have been selected such as they had an angle less than 45^0 with respect to the camera. If the background of the image which contains chess-board pattern is very cluttered then the *MATLAB* calibration application takes a long time to process the calibration result. Approximately 25% of the captured images have been filled by the used chess-board pattern. The pattern samples that have been used to calibrate the experimental camera are shown in Figure 3. The *MATLAB* calibration application requires the chess-board square size and the square size of the experimental chess-board is 25mm.

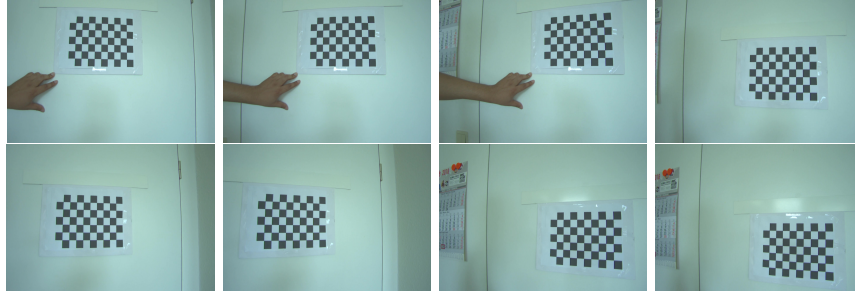


Figure 3: Used calibration pattern

3.3.2 Image addition and calibration

The selected images have been added to the *MATLAB* single camera calibration application. The corners of the chess-board have been automatically detected using the *Zhang* algorithm [23]. The calibration accuracy has been adjusted by adding or removing the images that contain the high re-projection error and adjusting the settings. Initial guesses have been used to obtain the expected calibration accuracy otherwise this application uses the linear least square method. The initial intrinsic matrix has been used as an initial guess by calculating the focal length and principal point in pixel unit using the known camera specification and the following equation.

$$f_{pixel} = (F_{mm}/S_{widthinmm}) \cdot (I_{widthinpixel}) \quad (10)$$

where the f_{pixel} is the focal length in *pixel* unit, the F_{mm} is the focal length in *mm* which can be obtained from the camera specification, the $S_{widthinmm}$ is the sensor width in *mm* and the $I_{widthinpixel}$ is the image width in *pixel* unit.

$$c_x = (I_{widthinpixel})/2 \quad (11)$$

$$c_y = (I_{heightinpixel})/2 \quad (12)$$

The principle point $[c_x, c_y]$ have been calculated by dividing the image height and width pixel value by 2 respectively. The value f_{pixel} is approximately

equal to f_x and f_y . Then the initial intrinsic matrix has been calculated by the below equation.

$$\begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ c_x & c_y & 1 \end{bmatrix} \quad (13)$$

The calibration algorithm assumes a pinhole camera model and the two basic steps of this algorithm are:

1. Solution for the intrinsics and extrinsics in a closed form, assuming that the lens distortion is zero [23]
2. Estimation of all the parameters simultaneously, including the distortion coefficients, using nonlinear least-squares minimization. Then use the closed-form solution from the preceding step as the initial estimate of the intrinsics and extrinsics. Set the initial estimate of the distortion coefficients to zero [23, 13]

3.3.3 Examine reprojection error

The distance between detected and re-projected points is known as re-projection error. This distance has been calculated in pixel unit. The *MATLAB* camera calibration application calculates the re-projection errors by projecting the chess-board points from the world coordinates, defined by the chess-board into the image coordinates [4, 23]. The application then compares the re-projected points to the corresponding detected points [4]. As a general rule, the mean re-projection errors less than one pixel are acceptable [13, 23]. The re-projection error using this application has been obtained 0.38 pixel.

4 Object detection

The object detection deals with the detecting instances of semantic objects of a certain class such as humans, buildings, or packages on the conveyor belt in the digital images and videos. It has large applications in many areas of computer vision, including image retrieval and video surveillance. Detecting the arbitrary objects on a cluttered scene is still a challenging task.

4.1 Segmentation

The segmentation algorithms are widely used to detect the arbitrary objects on a scene. Image segmentation is the process of partitioning an image into parts or regions. The objects in an image are divided into parts and these division are based on the characteristics of pixels in the same image. Dividing the objects in an image can also be done based on their color values or textures. The goal of the segmentation is to change the representation of an image into another form which is more meaningful and easier to analyze. The output of a segmented image is a set of segments that represent the entire original image. The pixels in each segment is similar with respect to some characteristics such as color, intensity or texture. The most widely used segmentation methods in image processing are

- Threshold based method
- Edge based method
- Region based method
- Clustering based method
- Watershed based method

4.1.1 Threshold based method

The thresholding methods are the most widely used and simplest method for image segmentation to analyze the individual objects in a scene. These methods divide the image pixels according to their intensity level [22]. These methods are applied to the images that have lighter intensity based objects than the background. The selection of these methods can be automatic or manual depending on prior knowledge or information of the image. The types of the thresholding based methods are

Global thresholding: The global thresholding can be done by selecting an appropriate threshold value of T . An input image $p(x, y)$ with the selected constant value of T can be converted into the segmented image $q(x, y)$ as follows:

$$q(x, y) = \begin{cases} 1, & \text{if } p(x, y) > T \\ 0, & \text{if } p(x, y) \leq T \end{cases}$$

Variable thresholding: The value of T can be varied over the image in variable thresholding. There are two types of variable thresholding which are local thresholding and global thresholding. The value of T in local threshold depends upon the neighborhood of x and y , on the other hand, the value of T is a function of x and y in global thresholding.

Multiple thresholding: The multiple threshold values such as T_0 and T_1 are used in multiple thresholding. The output of this method can be computed as:

$$q(x, y) = \begin{cases} m, & \text{if } p(x, y) > T_1 \\ n, & \text{if } p(x, y) \leq T_1 \\ 0, & \text{if } p(x, y) \leq T_0 \end{cases}$$

4.1.2 Edge based method

The edge based segmentation method depends on the rapid change of intensity value in an image because the single intensity value does not provide good information about the edge. The edge detection techniques locate the edges where either the first derivative of intensity is greater than a particular threshold or the second derivative has zero crossings. The edges are detected and connected together to form the object boundaries that is the principle of this method. There are two basic edge based segmentation. They are gray histograms and gradient-based method. To analyze and compute the boundaries of objects in an image *Sobel operator*, *Canny operator* and *Robert operator* can be used. The output of these methods are binary image.

4.1.3 Region based method

The regions in an image are a group of connected pixels with the similar properties [16]. In this approach, pixels are assigned to a particular object or region. In region-based segmentation, the pixels that correspond to an object are grouped together and marked as a region. There are two basic techniques of region-based segmentation that described below.

Region growing: The region growing means segment the image into a group of regions that are based on the growing of seeds or initial pixels. These seeds are selected manually or automatically based on a particular application. The growing of seeds are controlled by the connectivity between the neighboring pixels. The basic algorithm of the region growing method is, it starts from the initial pixels and the regions are iteratively enlarged by adding the neighboring pixels.

Region splitting and merging methods: The splitting stands for the iteratively dividing an image into the regions which contain similar characteristics, and the merging contributes to merge the adjacent similar regions. The procedure for the split and merge are given as [16]

1. Start with the whole image
2. When variance is too large then break it into quadrants
3. Merge adjacent regions that are similar enough
4. Repeat step (2) and (3) until no more splitting or merging occurs

The split and merge operations are execute on the quad-tree representation of an image.

4.1.4 Clustering based segmentation method

The clustering based techniques segment the image into clusters with similar pixel characteristics. The basic categories of clustering methods are the hierarchical and partition based method. The concept of trees is applied to the hierarchical based method. In this technique, the root of the trees represent the whole database and the internal nodes represent the clusters, on the other hand, the partition based method uses the optimization technique to

minimize an objective function iteratively. The types of the clustering based segmentations are given below [15]

Hard clustering: The hard clustering is the simplest clustering technique that divides the image into a set of clusters where one pixel can only belong to a single cluster. This method uses membership function with values either 1 or 0. If the certain pixel belongs to a particular cluster then 1 is defined else 0. The K-means clustering technique is the most popular example of the hard clustering. At first K-means clustering computes the center then each pixel is assigned to the nearest center.

Soft clustering: The soft clustering is a natural type of clustering technique because in real life exact division is not working due to the presence of noise. Thus soft clustering techniques can be applied for image segmentation where the division is not strict. Example of such type techniques is fuzzy c-means clustering. The pixels are partitioned into clusters based on partial membership in the fuzzy c-means clustering method. The single pixel can belong to more than one clusters and this degree of belonging is described by membership values [21]. This technique is more flexible than other techniques.

4.1.5 Watershed based methods

The concept of topological interpretation used in this segmentation technique. The intensity of an image represents the basins with a hole in its minima for the water spilling. When the water reaches the border of the basins, the adjacent basins are merged together. The dams are required for the separation between the basins and these dams are constructed using the dilation. The watershed methods consider the gradient of an image as the topographic surface. The pixels that have more gradient are represented as the boundaries which are continuous [20, 15].

4.2 Background subtraction

The methods for the real-time segmentation of the moving regions in an image sequences involves background subtraction or thresholding the error between an estimate of the image without moving objects and the current image [19]. The real-time segmentation of the moving regions in an image

sequences or videos is the fundamental step in many vision based systems including the automated visual surveillance, the human-machine interface, and the very low-bandwidth telecommunications [14]. The background subtraction calculates a reference image and then subtract the each new frame from this image and threshold the result. This result is a binary segmentation of the image which highlights the regions of a non-stationary objects [14]. Several algorithms are introduced for this purpose. The *OpenCV* and *MATLAB* have been implemented few such algorithms which are very easy to use. During this thesis the *vision.ForegroundDetector* function has been used from the *MATLAB* and the *BackgroundSubtractorMOG*, *BackgroundSubtractorMOG2*, and *BackgroundSubtractorGMG* function have also been examined from the *OpenCV* library to obtain the expected output. All the functions work based on the gaussian mixer model except the *BackgroundSubtractorGMG* function in *OpenCV*. The theory of the gaussian mixer model using expectation maximization algorithm is described below for modeling the background in real-time.

4.2.1 Gaussian mixture model

An image is represented by a matrix where each element of this matrix is pixel [8]. The value of this pixels is the number that shows intensity level or color of the image and it is considered as the matrix element value. Let the X is a random variable that takes this value. For a probability model determination, the mixture of gaussian distribution is considered by following equation [8].

$$f(x) = \sum_{i=1}^k p_i N(x|\mu_i, \sigma_i^2) \quad (14)$$

where k is the number of components or regions and $p_i > 0$ is the weights such that $\sum_{i=1}^k p_i = 1$.

$$N(\mu_i, \sigma_i^2) = \frac{1}{\sigma \sqrt{2\pi}} \exp\left(-\frac{(x - \mu_i)^2}{2\sigma_i^2}\right) \quad (15)$$

where mean and variance are known as μ_i and σ_i^2 . For a given image X the lattice data are the values of the pixels, and the mixer of gaussian is the pixel based model. The parameters are $\theta = (p_1, \dots, p_k, \mu_1, \dots, \mu_k, \sigma_1^2, \dots, \sigma_k^2)$ and the number of regions in the mixer of gaussian can be guessed by the histogram of lattice data.

4.2.2 Expectation maximization

An image is a mixer of gaussian and the expectation maximization(EM) algorithm is used to find out this gaussian. The expectation maximization algorithm has two step E and M . The algorithm can be defined as:

1. The expectation maximization finds the mean and variance of each gaussian distribution and the pixels are labeled according to the distribution number it relates to.
2. Probabilistic approach.
3. Recursively find the best solutions to mean and variance.

The number of gaussian distribution can be defined by looking at peaks of histogram.

4.3 Experimental result of object detection

Several background subtraction functions from the *OpenCV* and *MATLAB* have been tested for this experiment and finally one function from *MATLAB* has been selected. The *BackgroundSubtractorMOG* function from the *OpenCV* uses a method to model each background pixel by a mixture of K Gaussian distributions where $K = 3$ to 5 . The weights of mixture represent the time proportions that those colours stay in the scene. The probable background colours are the ones which stay longer and more static [14]. It has some optional parameters like length of history, number of gaussian mixtures, threshold etc which have been tuned for the expected output.

The function *BackgroundSubtractorMOG2* in *OpenCV* has extra feature for shadow detection but when this parameter is set to true the algorithm becomes slow. The very important feature of this algorithm is that it takes the appropriate number of gaussian distribution for each pixel where the function *BackgroundSubtractorMOG* takes K gaussian distributions throughout the algorithm. The function *BackgroundSubtractorMOG2* provides better adaptability to varying scenes due to the illumination changes etc [24].

The function *BackgroundSubtractorGMG* combines the statistical background image estimation and per-pixel *Bayesian* segmentation. It models the background using first few frames by default 120 frames. It employs probabilistic

foreground segmentation algorithm that identifies possible foreground objects using *Bayesian* inference [7]. The estimates are adaptive and newer observations are more heavily weighted than old observations to accommodate the variable illumination. To remove the unwanted noise several morphological filtering operations like closing and opening have been done after the background subtraction operation.

Finally, the function *vision.ForegroundDetector* has been used for the experiment from the *MATLAB*. This function compares the color or grayscale video frame to a background model to determine whether the individual pixels are the part of the background or the foreground [14, 19]. It then computes a foreground mask. The working principle of this function is more likely *BackgroundSubtractorMOG* in *OpenCV*.

The technique of morphology in image processing is based on the shape and form of objects. Structuring element is applied to an input image by morphological methods that creates an output image at the same size. Each pixel value in the input image is based on a comparison of the corresponding pixel in the input image with its neighbors. By selecting the size and shape of the neighbor the morphological operation is applied that is sensitive to the specific shapes in an input image. The morphological operations are performed on the binary image. These operations are erosion, dilation, opening, and closing. The idea of erosion is just like soil erosion, it erodes away the boundaries of the foreground objects. A pixel in the original image (either 1 or 0) is considered 1 only if all the pixels under the kernel is 1, otherwise it is eroded (made to zero). It means all the pixels near the boundary are discarded based on the size of the kernel. The thickness or size of the foreground object is reduced. It is useful for removing the small white noises or detach the two connected objects. The dilation is opposite of erosion. Here, a pixel element is 1 if at least one pixel under the kernel is 1. It increases the size of foreground object. In cases like noise removal, erosion is followed by dilation. The erosion removes white noises, but it also shrinks the objects but the object areas can be increased by dilation. It is also useful in joining broken parts of an object. Opening operation is just another name of the erosion followed by the dilation. The closing is reverse of opening, the dilation followed by erosion. After applying the *MATLAB* background subtraction function *vision.ForegroundDetector* and the morphological operations on a short video of moving objects the result in Figure 4 has been achieved for

each frame.

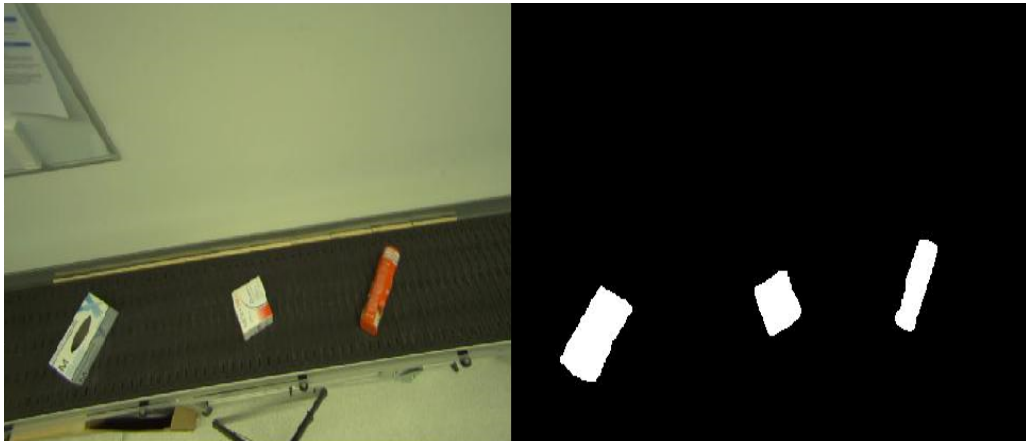


Figure 4: Detected objects from undistorted frame

5 Feature detection and description

In computer vision, the detection includes methods for computing abstractions of image information and making local decisions at every image point whether there is an image feature of a given type at that point or not. The result of this features is the subset of the image domain in the form of isolated points, continuous curves or connected regions.

Several methods have been proposed for feature detection and already in use for different application of computer vision and image processing. In this thesis most widely used Scale-invariant feature transform (SIFT), Speeded-up robust features (SURF), and Minimum eigenvalue algorithm for feature detection has been tested but depending on the output performance minimum eigenvalue based feature detection has been selected.

5.1 Minimum eigenvalue algorithm

The first attempt at eigenvalue based feature detection is done by Chris Harris and Mike Stephens in their paper *A Combined Corner and Edge Detector* in 1988 that is known as Harris corner detector. This algorithm basically finds the difference in intensity for a displacement of (u, v) in all directions that can be written as:

$$E(u, v) = \sum_{x,y} w(x, y) [I(x + u, y + v) - I(x, y)]^2 \quad (16)$$

where E is the difference between the original and moved window. The u, v is the window displacement in the x, y direction. $w(x, y)$ is the window at position (x, y) that acts like a mask. The I is the intensity of image at position (x, y) . The $I(x + u, y + v)$ is the intensity of moved window. The window function is either a rectangular window or gaussian window.

To maximize the function $E(u, v)$ corners need to be detected. It means, the second term of the function $E(u, v)$ need to be maximized:

$$\sum_{x,y} [I(x + u, y + v) - I(x, y)]^2 \quad (17)$$

this term can be expanded using the *Taylor* series or rewriting this term by its derivatives:

$$E(u, v) \approx \sum_{x,y} [u^2 I_x^2 + 2uv I_x I_y + v^2 I_y^2] \quad (18)$$

this equation can be arranged in a matrix form:

$$E(u, v) = [u, v] \left(\sum \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \right) \begin{bmatrix} u \\ v \end{bmatrix} \quad (19)$$

the summed-matrix can be renamed and put it to be M :

$$M = \sum w(x, y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \quad (20)$$

the final equation can be written as below form:

$$E(u, v) = \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} \quad (21)$$

the score is created, basically, that is an equation which determines that the window contains a corner or not:

$$R = \det(M) - k(\text{trace}(M))^2 \quad (22)$$

where $\det(M) = \lambda_1 \lambda_2$ and the $\lambda_1 \lambda_2$ are the eigen values of M . The $\text{trace}(M) = \lambda_1 + \lambda_2$. These eigen values decide whether a region is corner, edge or flat. When $\lambda_1 \sim \lambda_2$ the R is large and the region is corner. Later in 1994, Shi and Tomasi made a small modification to it in their paper *Good Features to Track* that provides better result compared to Harris corner detector. In this paper they just proposed the R should be minimum eigen values:

$$R = \min(\lambda_1, \lambda_2) \quad (23)$$

If it is a greater than the threshold value, then it can be considered as a corner.

5.2 Feature tracking

The feature detection methods and procedures have already been discussed in the above sections. The detected features have to be tracked in the next

and consecutive frames of a given video. Several methods have been proposed for feature tracking. One of them is *Lucas-Kanade* method. Before tracking a point the motion of the detected point have to be estimated. To estimate the motion of a point the optical flow can be applied. The concept of optical flow introduced by the American psychologist James J. Gibson in the 1940s to describe the *visual stimulus provided to animals moving through the world*. The optical flow is the pattern of the apparent motion of objects in an image between two consecutive frames that caused by the movement of object or camera. The optical flow has several application in different areas such as structure from motion, video compression and video stabilization. It works depending on below assumptions:

1. The pixel intensities of an object in an image do not change between consecutive frames
2. Neighboring pixels have similar motion

Consider a pixel $f(x, y, t)$ in first frame and It moves by distance (dx, dy) in next frame after dt time. According to the assumption pixel intensities have not changed due to movement which can be expressed by the following equation:

$$f(x, y, t) = f(x + dx, y + dy, t + dt) \quad (24)$$

an expansion of the expression on the right hand side into a *Taylor* series at the point (x, y) yields the below equation:

$$f(x, y, t) = f(x, y, t) + \frac{\delta f}{\delta x} dx + \frac{\delta f}{\delta y} dy + \frac{\delta f}{\delta t} dt + \{residual\} \quad (25)$$

when ignoring the terms of higher order or residual within the *Taylor* expansion following equation can be obtained:

$$f_x dx + f_y dy + f_t dt = 0 \quad (26)$$

where $f_x = \frac{\delta f}{\delta x}$ and $f_y = \frac{\delta f}{\delta y}$. With the definition of velocities in x/y direction $u = \frac{dx}{dt}$, $v = \frac{dy}{dt}$, the so-called motion constraint can be obtained that is shown by the following equation:

$$\varepsilon_m = f_x u + f_y v + f_t = 0 \quad (27)$$

The f_x and f_y can be found which are considered as the image gradient. Similarly, f_t is the gradient along time. But (u, v) is unknown. It is not possible to solve because one equation with two unknown variables. This problem can be solved by the *Lucas-Kanade* method. This method takes a 3×3 patch around the point and all the 9 points have same motion. The (f_x, f_y, f_t) can be found for these 9 points. Now need to solve 9 equations with two unknown variables. A better solution can be obtained with least square fit method. Below is the final solution. There is two equation two unknown problem and this equations can be solved to get the solution:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum_i f_{x_i}^2 & \sum_i f_{x_i} f_{y_i} \\ \sum_i f_{x_i} f_{y_i} & \sum_i f_{y_i}^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i f_{x_i} f_{t_i} \\ -\sum_i f_{y_i} f_{t_i} \end{bmatrix} \quad (28)$$

The idea behind of this method is simple, some points need to give for tracking and the optical flow vectors can be received for those points. But still, there is some problem because up to this the small motions have been dealt. So it fails during large motion. In this case the pyramids need to be considered. the upper values of the pyramid remove the small motion and the large motion consider as small motion. Now the *Lucas-Kanade* can be applied to get the optical flow along with a scale.

5.3 Experimental result of feature detection and tracking

To detect features in moving objects the *MATLAB* function *detectMinEigenFeatures* has been used. This function uses minimum eigenvalue to detect the features in an image or a video frame. The features have been detected only in the conveyor belt region because the other regions of the image are static and the change of image intensity is happened into this region. So the detected feature points in the conveyor belt region in a frame have been tracked after three frames to find the match points and epipolar inliers that estimate the fundamental matrix, unknown rotation, and translation between these two frames. The detection of features can be increased by reducing the value of the tunable *MinQuality* parameter of this function for the dense reconstruction but reducing the *MinQuality* parameter value can distort the scene reconstruction. The *MinQuality* parameter value has been selected after several simulations where it gives the consistent result. To track the detected features in another view the *MATLAB* function *vision.PointTracker* has been used. This function uses the *Lucas-Kanade* method to track the

given points in other view. Figure 5 shows the detected features using the minimum eigenvalue algorithm in the 50th frame and the tracked features in the 53rd frame of the same video using the *Lucas-Kanade* method. The frame differencing depends on the object motion. For example, larger the motion smaller the frame difference but the very large motion of objects the *Lucas-Kanade* tracking method can break down.

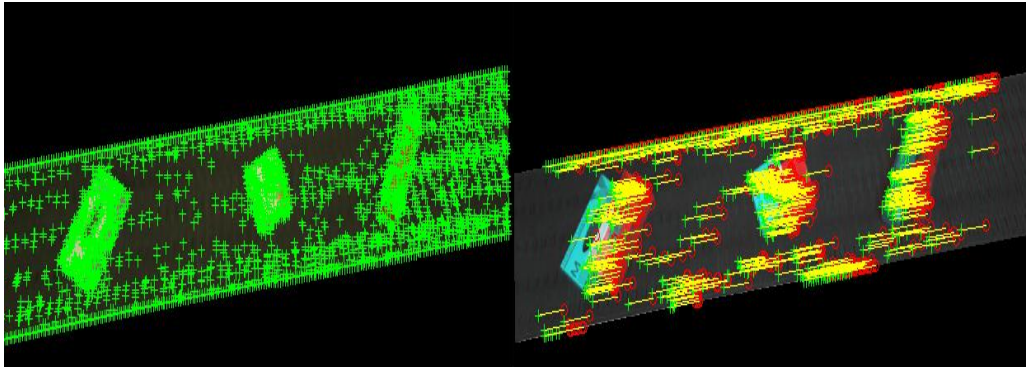


Figure 5: Feature detection and tracking

6 Multiview geometry

There exist complex geometric relations between multiple views of a 3D scene. These relations are based on the camera motion and calibration as well as to the scene structure. In this chapter, the theory behind the 3D reconstruction of objects through two view geometry has been discussed.

6.1 Epipolar geometry

When an image is taken by the pinhole camera an important information is lost that known as the depth of an image or the distance of each point in an image from the camera because it is a 3D to 2D conversion. This depth information can be recovered using two camera or a moving single camera. Human eyes work in similar way where the human uses two cameras (two eyes) which is called stereo vision. The question is how to do it? and the simple answer is using the epipolar geometry. The epipolar geometry is known as the two view geometry [11]. The two perspective views may be acquired simultaneously, for example in a stereo rig or by a moving camera. From the geometric viewpoint, the two situations are equivalent. Most of the 3D scene points must be visible in both views simultaneously. Image points in left view and their corresponding points in right view represents the projection of the same 3D scene point. The knowledge of image correspondences enables scene reconstruction from the images.

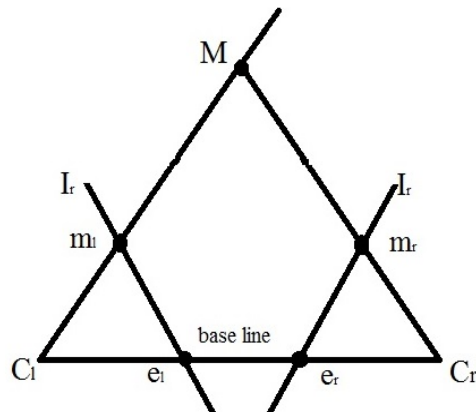


Figure 6: Epipolar geometry

Algebraically each perspective view has 3×4 camera projection matrix P which represents the mapping between the 3D world and a 2D image. From Figure 6 the camera projection matrices of the left and right view are P_l and P_r . The 3D point M is then imaged in the left view and in the right view as: [11]

$$\zeta_l m_l = P_l M \quad (29)$$

$$\zeta_r m_r = P_r M \quad (30)$$

Geometrically the position of an image point m_l in the left image plane I_l can be found by drawing optical ray through the left camera projection center C_l and the scene point M . The ray intersects the left image plane I_l at m_l . Similarly, the optical ray connecting the C_r and M that intersects right image plane I_r at m_r . The relationship between the image points m_l and m_r is given by the epipolar geometry. The corresponding image points must lie on the particular image lines which can be computed without information on the calibration of the cameras. This implies that given a point in one image one can search the corresponding point in the other along a line. Any 3D point M and the camera projection centres C_l and C_r define a plane which known as epipolar plane. The projections of the point M is the image points m_l and m_r lie in the epipolar plane if they lie on the rays connecting the corresponding camera projection centre and point M . The epipolar lines I_l and I_r are the intersections of the epipolar plane with the image planes. The line connecting the camera projection centres C_l and C_r is called the baseline. The baseline that intersects each image plane in a point is called epipole. By construction the left epipole e_l is the image of the right camera projection centre C_r in left image plane and similarly, the right epipole e_r is the image of left camera projection centre C_l in the right image plane. All epipolar lines in the left image go through e_l and all epipolar lines in the right image go through e_r . In some cases, it is not possible to locate the epipole in the image because they may be outside the image which means, one camera doesn't see the other.

6.1.1 Epipolar constraint

The epipolar plane is defined by the camera projection centres and one image point. Therefore, the epipolar line in the right image can be found by given

a point m_l , where the corresponding point m_r must lie. To find the matching point in other image it is not necessary to search the whole image just search along the epiline. So it provides better performance and accuracy. This is called Epipolar Constraint. As mentioned before, the right epipolar line corresponding to m_l geometrically represents the projection onto the right image plane. Then the following equation describes the right epipolar line: [11].

$$\zeta_r m_r = e_r + \zeta_l P_{3 \times 3, r} P_{3 \times 3, l}^{-1} m_l \quad (31)$$

This is the equation of a line through the right epipole e_r and the image point $m'_l = P_{3 \times 3, r} P_{3 \times 3, l}^{-1} m_l$ which represents the projection onto the right image plane of the point at infinity of optical ray of m_l . The equation for the left epipolar line can be obtained in the similar way. The epipolar geometry can be described analytically in several ways depending on the amount of prior knowledge about the stereo system. Three general cases can be identified:

1. If the intrinsic and extrinsic parameters of a camera are known, the epipolar geometry can be described in terms of the projection matrices
2. If only the intrinsic parameter is known, the coordinates need to be normalized and the epipolar geometry is then described by the essential matrix
3. If neither intrinsic nor extrinsic parameters are known the epipolar geometry is described by the fundamental matrix.

6.1.2 Fundamental matrix

Algebraic representation of epipolar geometry is known as fundamental matrix. In the following fundamental matrix can be driven from the mapping between a point and its epipolar line, and then specified the properties of the matrix. Consider a pair of normalized cameras, without loss of generality, the world reference frame is then fixed onto the first camera, hence:

$$P_l = [I|0] \quad (32)$$

$$P_r = [R|t] \quad (33)$$

using these two projection matrices following equation can be obtained : [11]

$$\zeta_r m_r = e_r + \zeta_l K_r R K_l^{-1} m_l, \quad (34)$$

where the $e_r = K_r t$ and the above equation states that point mr lies on the line through the er and $K_r R K_l^{-1} m_l$. Then the fundamental matrix F can be written as: [11]

$$F = e_r \times K_r R K_l^{-1}, \quad (35)$$

which gives the relationship between the corresponding image points in pixel coordinates. The defining equation of the fundamental matrix is therefore: [11]

$$m_r^T F m_l = 0 \quad (36)$$

The F is 3×3 , rank-two homogeneous matrix. It has seven degrees of freedom since it is defined up to a scale and the determinant of F is zero. F is completely defined by pixel correspondences that means the intrinsic parameters are not needed. For any point m_l in the left image, the corresponding epipolar line l_r in the right image can be expressed as $l_r = F m_l$ and similarly, for point m_r it is $l_l = F^T m_r$. The left epipole e_l is the right null-vector of the fundamental matrix ($F e_l = 0$) and the right epipole is the left null-vector of the fundamental matrix ($e_r^T F = 0$). Methods of estimating the fundamental matrix given below:

Normalized eight-point algorithm (Norm8Point): Method for reliable results. The inputs and the corresponding points of this input must match precisely for this method.

Least median of squares (LMedS): This method can be used if at least fifty percent of input points and their corresponding points are inlier.

Random sample consensus (RANSAC): This method can be selected if the distance threshold for inliers are necessary.

M-estimator sample consensus (MSAC): This method works as like random sample consensus (RANSAC) but this method can converge more quickly than the RANSAC.

Least trimmed squares (LTS): This method can be selected if the minimum percentage of inliers between input points and their corresponding points are known. Generally, the LTS method converges more quickly than the LMedS method.

6.1.3 Triangulation

Consider the camera matrices P_l and P_r and m_l, m_r are two corresponding points satisfying the epipolar constraint $m_r^T F m_l = 0$. It means that the m_r lies on the epipolar line $F m_l$ and the two rays back-projected from image points m_l and m_r lie in a common epipolar plane. Since they lie in the same plane, so they intersect at some point. This point is reconstructed 3D scene point M . Figure 7 shows the result of 3D points using triangulation.

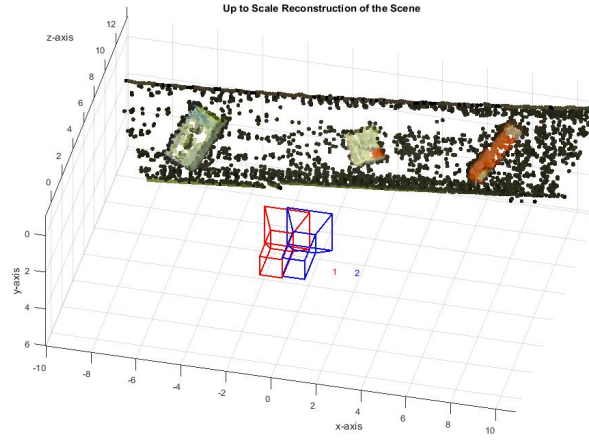


Figure 7: Point cloud using triangulation

Analytically, the reconstructed 3D point M can be found by solving for parameter ζ_l and ζ_r [11]. That can be rewritten as:

$$m'_l = -\frac{1}{\zeta_l} e_r + \frac{\zeta_r}{\zeta_l} m_r, \quad (37)$$

the unknowns are ζ_r and ζ_l . Both encode the position of M in space, as ζ_r is the depth of M with respect to the right camera and ζ_l is the depth of M with respect to the left camera. Three points m_r, e_r and m'_l are known and they are also collinear, so the ζ_l can be solved by using the below equation in closed form expressions:

$$\frac{1}{\zeta_l} = \frac{(m'_l \times m_r) \cdot (m_r \times e_r)}{\|m_r \times e_r\|^2} \quad (38)$$

the reconstructed 3D point M can then be calculated by inserting the value ζ into following equation:

$$M = \begin{bmatrix} -P_{3 \times 3}^{-1} \mathbf{P}_4 \\ 1 \end{bmatrix} + \zeta \begin{bmatrix} P_{3 \times 3}^{-1} m \\ 0 \end{bmatrix}, \quad (39)$$

where P_4 is the fourth column of normalized P and m is the image point. In reality, camera parameters and image locations are known only approximately [11]. Back-projected rays therefore not actually intersect in space.

7 Measurement analysis

Two techniques have been used to measure the object dimension. In this chapter, both techniques have been described by stepwise.

7.1 Technique 1

In this technique, the pixel information of each object and the $3D$ coordinate of the center point of each object have been used to measure the object dimension. The steps that have been followed in this technique are given below:

Step 1: After the camera calibration, the intrinsic and lens distortion parameters have been obtained which is used to remove the distortion of frames. One example is given by the Figure 8.

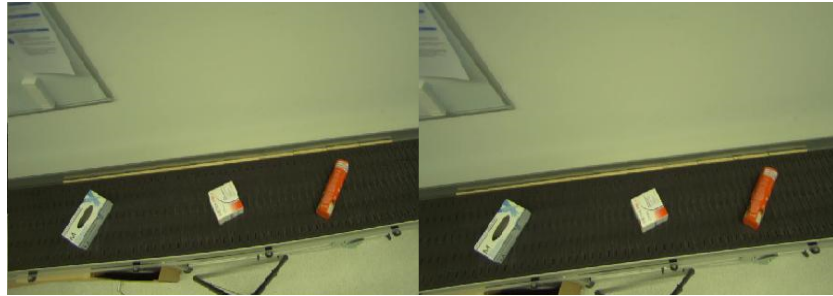


Figure 8: Original and undistorted frame

Step 2: The background subtraction has been applied to obtain only the interested object regions. Then the object regions have been labeled to measure the length and width of each object. Figure 9 shows the labeling of objects with different colors.



Figure 9: The labeled objects

Step 3: The oriented bounding box has been drawn over the each labeled object region. The center point for each bounding box has been calculated. Figure 10 shows the bounding box and center point of each object.

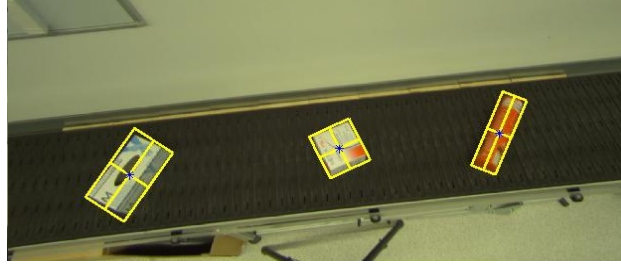


Figure 10: The center point and bounding box of each object

Step 4: The length and width of each bounding box have been stored as the length and width of each object in the pixel unit that have been calculated by *Euclidean* distance formula. Then the pixel unit has been scaled to the *cm* unit.

Step 5: To measure the height of each object the 3D coordinate of the center point in each bounding box is required. To obtain this 3D coordinate the two-view geometry has been applied. The 50th and 53rd number undistorted frames have been taken from the video. Then the conveyor belt region has been taken as a region of interest from both frames because the objects have been changed their position only into this region and the other regions are static. Figure 11 shows the changing and selected region of frames.

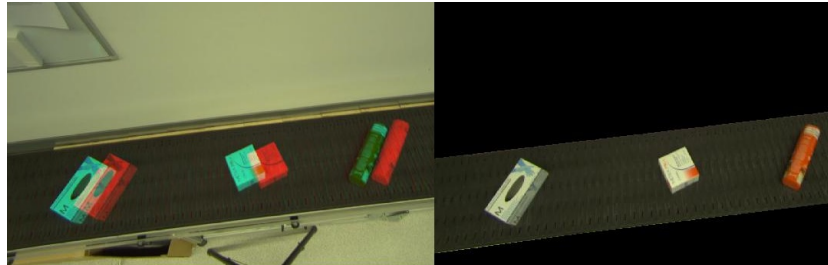


Figure 11: The region of interest

Step 6: The minimum eigenvalue algorithm has been used to detect and describe the features in the selected region of interest of the 50th frame. Then these features have been tracked to the 53rd frame using the *Lucas-Kanade* feature tracking method to find the match points and epipolar inliers that has been shown in Figure 12.

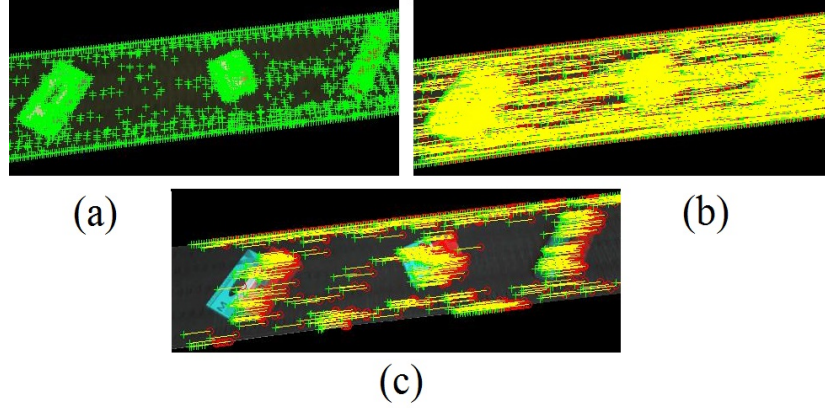


Figure 12: The features (a), match features (b), and epipolar inliers (c)

Step 7: The match points and the camera intrinsic matrix have been used to obtain the fundamental matrix. This fundamental matrix has been decomposed to obtain the unknown rotation and translation between the 50th and 53rd frames that is known as the singular value decomposition, and this procedure has been applied with the help of the intrinsic camera matrix and inlier points between these frames, where the intrinsic camera matrix is:

$$\text{Intrinsic camera matrix} = \begin{bmatrix} 1019 & 0 & 0 \\ 0 & 1020 & 0 \\ 940 & 599 & 1 \end{bmatrix} \quad (40)$$

Step 8: The 50th frame has been considered as the first camera that is at the origin looking along the Z -axis. Thus, its rotation matrix is identity, and its translation vector is 0. The 53rd frame has been considered as the second camera that has rotation and translation and the obtained rotation and translation in step 7 have been used to the second camera.

Step 9: The 3×4 perspective projection matrices have been calculated

for both cameras using the obtained rotation and translation and all the frames into this video must satisfy these perspective projection matrices if their difference are 3 and no change in object motion. It means the perspective projection matrices that are obtained from the 50th and 53rd frames by following the above procedure have been applied to all frames in the video. The perspective projection matrices or the camera matrices are:

$$\text{Camera matrix for 50th frame} = \begin{bmatrix} 1019 & 0 & 0 \\ 0 & 1020 & 0 \\ 940 & 599 & 1 \\ 0 & 0 & 0 \end{bmatrix} \quad (41)$$

$$\text{Camera matrix for 53rd frame} = \begin{bmatrix} 1.0220 & 0.0021 & 0 \\ -0.0025 & 1.0185 & 0 \\ 0.9368 & 0.6015 & 0.0010 \\ -1.0285 & 0.0829 & -0.0 \end{bmatrix} \quad (42)$$

Step 10: The 2D center point of bounding boxes and their corresponding center point in another view have been triangulated to 3D point using the obtained perspective projection matrices. For example, the corresponding center point of the 50th frame bounding boxes is the center point of bounding boxes in the 53rd frame and both of them have been used to triangulate the 3D point. Figure 13 shows both center point of both views bounding boxes in blue dot and all the 3D points that have been matched between the 50th and 53rd frames.

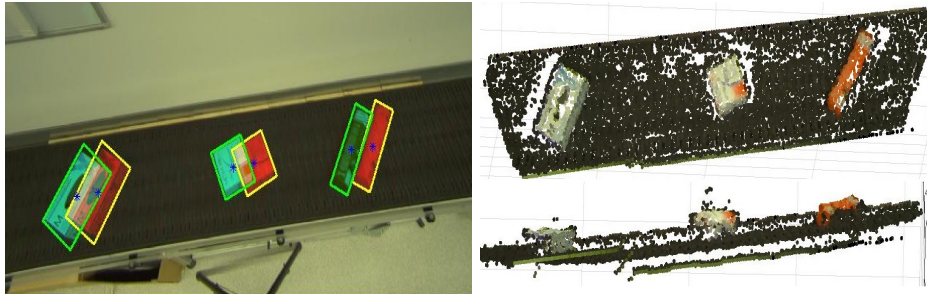


Figure 13: The center points and the 3D match points from the top and side view

The sum of the square root of the bounding box center 3D point is the

distance of this bounding box center point from the camera center point. This distance has been summed with a known object height in a specific region and kept constant, and this constant value is the distance of the conveyor belt from the camera. Then the distance of each object center point has been subtracted from the conveyor belt distance in the defined region and scaled to obtain the object height. Some measurement example has been shown in Figure 14. The measurement has been stored when the object comes to the approximately middle point of the blue box region.

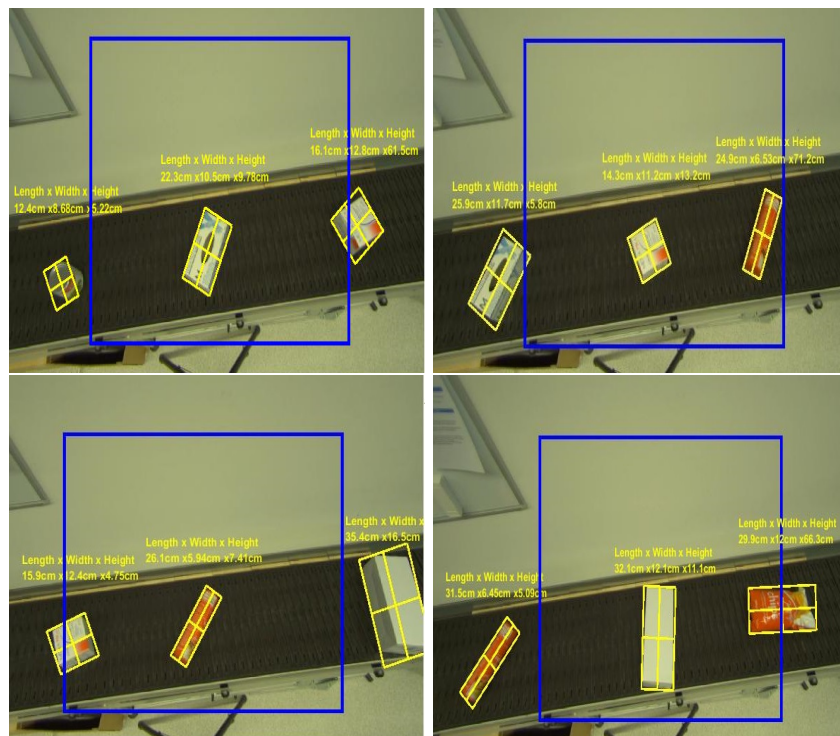


Figure 14: The real-time object dimension measurement

The average measurement error analysis has been shown in below Table 1:




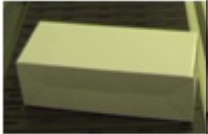
Object	Original dimension	Measure dimension	Error
	Length = 23.5cm Width = 12.5cm Height = 7cm	Length = 22.3cm Width = 10.5cm Height = 9.78cm	20%
	Length = 13.5cm Width = 8.5cm Height = 13cm	Length = 14.3cm Width = 11.2cm Height = 13.2cm	13%
	Length = 28.5cm Width = 7cm Height = 7cm	Length = 26.1cm Width = 5.94cm Height = 7.41cm	9.6%
	Length = 33cm Width = 10.5cm Height = 14cm	Length = 32.1cm Width = 12.1cm Height = 11.1cm	16%

Table: 1 Error analysis

7.2 Technique 2

In this technique, the recovered $3D$ information of the object has been used to measure their dimension. The steps that have been followed in this technique are given below:

Step 1: The mask has been applied to the binary segmented objects to return them as the three channel colored objects. Figure 15 shows the result.

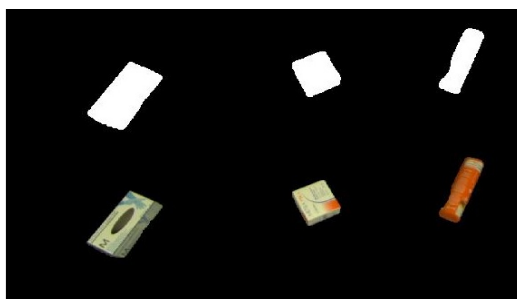


Figure 15: The segmented color objects

Step 2: The minimum eigenvalue algorithm has been applied to the segmented colored objects for feature detection. Then the detected features have been tracked in another view to find the matches. Figure 16 shows the result.

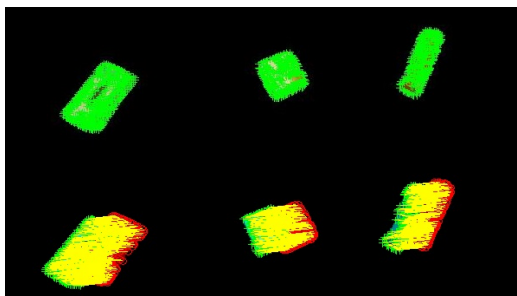


Figure 16: The feature detection and tracking

Step 3: The matched points have been triangulated to $3D$ points using the camera perspective projection matrices which are obtained in step 9 of technique 1. Then the K-means clustering has been applied to the $3D$ point

cloud to separate the point cloud for each object. Figure 17 shows the original point cloud and the labeled point cloud for each object in different color.

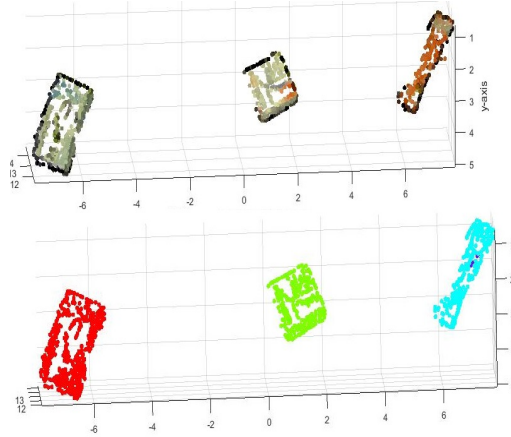


Figure 17: Original and clustered point cloud

Step 4: The minimum cuboid has been drawn over the clustered point cloud to measure the object dimensions. Figure 18 shows the cuboid over each object point cloud.

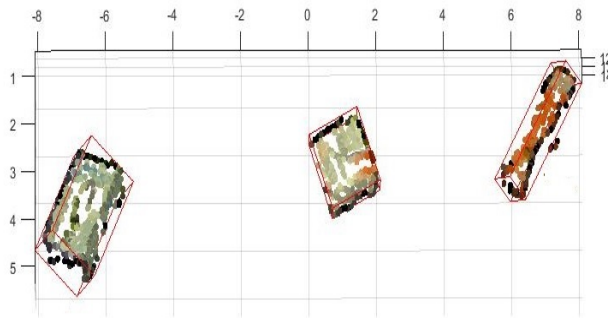


Figure 18: The cuboid over each clustered point cloud

Step 5: The *Euclidean* distance formula has been applied to the coordinates of the cuboid for calculating the dimensions of each object. Figure 19 shows the coordinates of the cuboid that have been used to calculate the object dimensions.

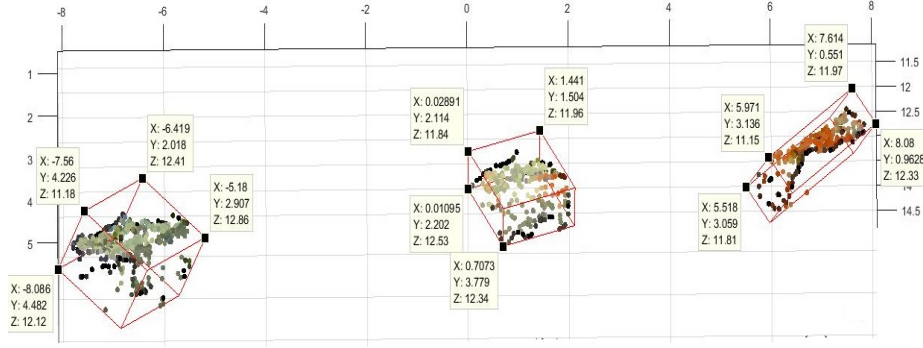


Figure 19: Object dimension measurement

The length, width, and height of the left object in Figure 19 have been calculated as:

$$\text{Length} = \sqrt{(-7.56 + 6.419)^2 + (4.226 - 2.018)^2 + (11.18 - 12.41)^2} = 2.77$$

$$\text{Width} = \sqrt{(-6.419 + 5.18)^2 + (2.018 - 2.907)^2 + (12.41 - 12.86)^2} = 1.6$$

$$\text{Height} = \sqrt{(-7.56 + 8.086)^2 + (4.226 - 4.482)^2 + (11.18 - 12.12)^2} = 1.10$$

The original length, width, and height of this object are 28cm, 7cm, and 7cm respectively. To convert the measured value into the original value the multiplicative scale factor 8.45 has been used, then the measured values are: Length = 23.4cm, Width = 13.5cm, and Height = 9.3cm. Figure 20 shows the visualization using this technique. Some measurement result and the average error have been shown in Table 2 using this technique.

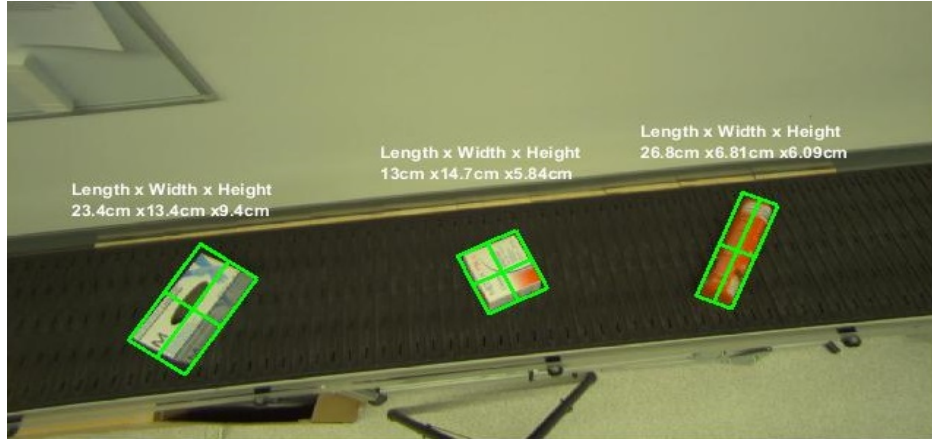


Figure 20: Real time measurement using cuboid over the point cloud




Object	Original dimension	Measure dimension	Error
	Length = 23.5cm Width = 12.5cm Height = 7cm	Length = 23.4cm Width = 13.4cm Height = 9.4cm	10%
	Length = 13.5cm Width = 8.5cm Height = 13cm	Length = 13cm Width = 5.84cm Height = 14.7cm	16%
	Length = 28.5cm Width = 7cm Height = 7cm	Length = 26.8cm Width = 6.81cm Height = 6.09cm	6.5%

Table: 2 Error analysis

The technique 2 is computationally very expensive. The feature detection and matching procedure in each frame make the system very slow to compute the $3D$ points in technique 2. The noise reduction from the point cloud is very challenging task to draw the minimum bounding box for each object.

The error comparison shows that the both techniques provide good result. The technique 1 is quite faster than the technique 2 because it computes a single $3D$ point (e.g object center point).

8 Conclusion

The measurement of the three-dimensional object using a single camera is a very complex task, and this task has been solved in this thesis. The measurement of the length and width of an object from the $2D$ information is possible if the segmentation is perfect. In this thesis, the segmentation of the moving object has been done very precisely with the help of the object motion. The measurement error has been found 1 to $3cm$ and this error can be reduced by placing the experimental camera perfectly parallel to the conveyor belt. The measurement of object height makes the system very slow because of a large number of image point calculation. The very consistent point cloud has been achieved in order to analyze the perfection of fundamental matrix, rotation matrix, and translation vector of the virtually created second camera. The measurement of planner object height using this procedure is not feasible. The object height between 1 to $3cm$ is considered as a planner object.

The camera selection is very important for this type of experiment. The camera that has been used in this experiment is a fish-eye camera that has positive radial distortion. To calibrate the camera, the pinhole camera model has been used because if the fish-eye camera has the horizontal angle of view less than 180^0 then the pinhole camera model is applicable for the fish-eye camera calibration. The standard camera should be used for this kind of experiment that has very less lens distortion because the lens distortion is not perfectly removed by the conventional camera calibration algorithm. The lens distortion problem has been suffered a lot during this thesis.

The curved $3D$ reconstruction of the scene has been obtained in several times during this thesis because of the fish-eye or large radial lens distortion. Finally, an approximately good $3D$ reconstruction has been achieved to estimate the virtually created camera perspective projection matrix to measure the object height in a defined region. The error in the measurement of the object length and width can be precisely removed by placing the camera parallel to the conveyor belt but the experimental camera has not been parallel to the conveyor belt because of some equipment problem. The obtained measurement result has satisfied the thesis goal.

References

- [1] Jean-Yves Bouguet. *Camera Calibration Toolbox for Matlab*. Computational Vision at the California Institute of Technology, http://www.vision.caltech.edu/bouguetj/calib_doc/, Accessed: August 10, 2018.
- [2] Chuong B Do and Serafim Batzoglou. *What is the expectation maximization algorithm?* Nature Publishing Group, Vol. 26, 2008, pp. 897-899.
- [3] MATLAB Documentation. *Pin hole camera model and camera calibration parameters*. <https://www.mathworks.com/help/vision/ug/camera-calibration.html>, Accessed: August 8, 2018.
- [4] MATLAB Documentation. *Single camera calibrator app*. <https://www.mathworks.com/help/vision/ug/single-camera-calibrator-app.html>, Accessed: August 8, 2018.
- [5] OpenCV Documentation. *Camera calibration and 3D reconstruction fundamentals*. https://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html, Accessed: August 8, 2018.
- [6] OpenCV Documentation. *Understanding of features and their detection and description methods*. https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_table_of_contents_feature2d/py_table_of_contents_feature2d.html, Accessed: August 13, 2018.
- [7] OpenCV Documentation. *Background subtraction methods and procedures*. https://docs.opencv.org/3.2.0/db/d5c/tutorial_py_bg_subtraction.html, Accessed: August 9, 2018.
- [8] Rahman Farnoosh, Gholamhossein Yari, and Behnam Zarpak. *Image Segmentation Using Gaussian Mixture Model*. IUST International Journal of Engineering Science, Vol. 19, 2008, pp. 29-32.
- [9] Faugeras and Toscani. *Camera calibration for 3d computer vision*. International Workshop on Machine Vision and Machine Intelligence, 1987 pp. 240-247.

- [10] Gernot A. Fink. *Lecture Notes for Computer Vision*. TU Dortmund, 2015, pp. 67-68.
- [11] Andrea Fusiello. *Elements of Computer Vision: Multiple View Geometry*. <http://www.diegm.uniud.it/fusiello/teaching/mvg/mvg.html>, 2005, Course note part-1, pp. 23-40.
- [12] Rachid Guerchouche and François Coldefy. *Camera Calibration Methods Evaluation Procedure for Images Rectification and 3D Reconstruction*. The 16-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision in co-operation with EUROGRAPHICS, 2008, pp. 205-210.
- [13] Janne Heikkilä and Olli Silven. *A Four-step Camera Calibration Procedure with Implicit Image Correction*. IEEE International Conference on Computer Vision and Pattern Recognition, 1997, pp. 1106-1112.
- [14] P. KaewTraKulPong and R. Bowden. *An Improved Adaptive Background Mixture Model for Realtime Tracking with Shadow Detection*. In Proc. 2nd European Workshop on Advanced Video Based Surveillance Systems, AVBS01, VIDEO BASED SURVEILLANCE SYSTEMS: Computer Vision and Distributed Processing, 2001.
- [15] Dilpreet Kaur and Yadwinder Kaur. *Various Image Segmentation Techniques: A Review*. International Journal of Computer Science and Mobile Computing, Vol. 3, 2014, pp. 809-814.
- [16] Manjot Kaur and Pratibha Goyal. *A Review on Region Based Segmentation*. International Journal of Science and Research, Vol. 4, 2015.
- [17] F. Coldefy R. Guerchouche and T. Zaharia. *Accurate camera calibration algorithm using a robust estimation of the perspective projection matrix*. Proc. of SPIE Mathematics of Data/Image Pattern Recognition, Compression, and Encryption with Applications IX, Vol. 6315, 2006.
- [18] Davide Scaramuzza, Agostino Martinelli, and Roland Siegwart. *A Toolbox for Easy Calibrating Omnidirectional Cameras*. Proceedings to IEEE International Conference on Intelligent Robots and Systems, 2006, pp. 5695-5701.

- [19] Chris Stauffer and W.E.L Grimson. *Adaptive background mixture models for real-time tracking*. Computer Vision and Pattern Recognition, IEEE Computer Society Conference, Vol. 2, 1999, pp. 2246-252.
- [20] Qing-Qiang Yang Wen-Xiong Kang and Run-Peng Liang. *The Comparative Research on Image Segmentation Algorithms*. IEEE Conference on ETCS, 2009, pp. 703-707.
- [21] Mahesh Yambal and Hitesh Gupta. *Image Segmentation using Fuzzy C Means Clustering: A survey*. Journal of Advanced Research in Computer and Communication Engineering, Vol. 2, Issue 7, 2013.
- [22] Yu-Jin Zhang. *An Overview of Image and Video Segmentation in the last 40 years*. Proceedings of the 6th International Symposium on Signal Processing and Its Applications, 2001, pp. 144-151.
- [23] Zhengyou Zhang. *A Flexible New Technique for Camera Calibration*. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, 2000, pp. 1330-1334.
- [24] Zoran Zivkovic. *Efficient Adaptive Density Estimation per Image Pixel for the Task of Background Subtraction*. Pattern Recognition Letters, Vol. 27, Issue 7, 2006, pp. 773-780.