CZECH TECHNICAL UNIVERSITY IN PRAGUE
FACULTY OF MECHANICAL ENGINEERING
DEPT. OF INSTRUMENTATION AND CONTROL ENGINEERING

DESIGN OF A LASER DISTANCE SENSOR WITH A WEB CAMERA FOR A MOBILE ROBOT

ASHYKHMIN MYKHAILO

DIPLOMA THESIS

PRAGUE 2016

Czech Technical University in Prague | Faculty of Mechanical Engineering

Dept. of Instrumentation and Control Engineering | Academic year: 2015/2016

DIPLOMA THESIS SPECIFICATION

Name of student: Bc. Ashykhmin Mykhailo

Field of study: Instrumentation and Control Engineering

Title: Design of a laser distance sensor with a web camera for a mobile robot

Diploma thesis guidelines:

1. Search for suitable sensors, parts and algorithms
2. Design and assemble the required hardware
3. Design a LabView program
4. Experimentally test your designed device

Extent of graphical work: according diploma thesis supervisor instructions

Extent of cover letter: according diploma thesis supervisor instructions

List of literature:
[1]LabView User Manuals
[2] Almost Everything You Always Wanted to Know About 3D Scanning (But Were Afraid To Ask), online on <http://www.dirdim.com/lm_everything.htm>, accessed on 25.2.2016
[3]Vierling, F., Practical 3D, Mayer & Söhne Druck- und Mediengruppe, 2013, ISBN 978-3000395826

Diploma thesis supervisor:	Doc. Ing. Martin Novák, Ph.D.
Submission date:	12. 4. 2016
Delivery date:	15. 6. 2016

If the student does not submit the bachelor's or diploma thesis until the due date and this fact was previously explained in a written form and the excuse was accepted by the dean, an alternate due date of bachelor's or diploma thesis submission is appointed by the dean. If the student did not ask for an excuse correctly or if the excuse was not accepted by the dean, the student is allowed to enroll for the bachelor's or diploma thesis for a second time.

The diploma candidate accepts hereby that he/she is obligated to work out the diploma thesis on his/her own, without extraneous help except the given consultations. List of literature, other sources and consultants' names must be stated in the diploma thesis.

Diploma thesis specification taken over by: ....
Student Date

Head of Dept: doc. Ing. Jan Chyský, CSc. *Dean:* prof. Michael Valášek, DrSc.

Prague, 25. 2. 2016

Abstract

Mobile robots have become more widespread in commercial, scientific and industrial usage. Modern challenges require robots to autonomously navigate its course and being capable to avoid obstacles in unknown environment. Devices for obstacle detection and distance determination are beneficial for robot navigation.

This thesis presents device and vision processing software application created for mobile robot. Designed device consists of required elements – web-camera and laser emitter, assembled together in movable frame, which can be easily connected to driven motor and to robot. Using web-camera image, the software must recognize released laser line on the surface of obstacle, and determine distance to it.

The vision software is based on brightest pixel extraction from image to detect the position of laser beam in environment and applying triangulation principles for distance calculation. The precision of this approach was demonstrated by successful tests.

Acknowledgment

This diploma thesis is about the design of a hardware device and software application for laser distance determination. During my work I have reviewed possible algorithms and solutions, applied techniques which satisfy necessary conditions and designed system which solve given task evaluating result.

I would like to thank my supervisor, Doc. Ing. Martin Novák, Ph.D., for his enthusiasm and guidance to the diploma thesis work and semester projects during my Master’s degree study at Czech Technical University. Much of my knowledge of LabView software and useful experience comes from these projects.

List of Contents

1. Introduction

Devices with sensors for obstacle detection and distance evaluation are very advantageous for robot navigation. These applications can used for determination the distance to objects, exploration and mapping of environment.

The method is rather low-cost, as hardware solution consist of two main components: web camera and laser emitter, assembled by common frame and does not require additional expensive and complex equipment. Mounted device can easily connected to front panel of mobile robot or car.

Use of web camera and laser emitter in such operations allows to provide real-time contactless measurements. What is especially important in unknown or dangerous areas with low opportunity for human access. For example – abandoned buildings, underground tunnels or mined fields.

The image captured by web camera is then processed in controlling systems or in various software, which perform detection of laser beam on camera image and calculation of obstacle distance.

Image processing is widely distributed in industrial and research applications in computing platforms. With high-resolution web cameras, widely used today, it is possible to design accurate system based on image processing and mathematical algorithm with good reliability of metering.

2. Overview of Task Idea

2.1 Goals

1. Search for suitable sensors, parts and algorithms
2. Design and assemble the required hardware
3. Design a LabView program
4. Experimentally test your designed device

The main purpose of this work is determination of processing image in LabView software installed on the personal computer, with a hardware system formed by web camera and laser emitter in order to detect and measure distance to obstacle.

The initial idea of thesis is that the LabView program can process images in real time and perform the mathematical model for calculating distances to obstacle with laser for an application of distance determination.

To reach the final goal there are some steps to increment.

To review the possible algorithms, choose the most suitable of them and design hardware, which will satisfy all necessary conditions. Then to assemble it together by one frame with chosen dimensions, which is very important and influenced on further mathematical algorithms for calculation distance.

To design the software solution in LabView, which will normalize the image obtaining by camera and extract the data of position of pixel of brightest points, which evaluates the position of laser on the screen. To implement mathematical model for distance determination based on obtained pixel data of laser spots in LabView project.

To run the software application, perform experiments and compare the results with calculated and actual values of distance. To review advantages, limitations and features of whole algorithm. To test the assembled device in connection with driven motor and mobile robot.

2.2 Theoretical Background

Vision is one of the main senses used by humans to move around the world. The amount of information received through the eyes is incomparable with other senses. The goal of the artificial vision, in the field of computer vision, is to construct a computational prototype capable of understand a digital image. For descriptions of scenes, computer vision provides techniques such as pattern recognition, statistical learning, projective geometry, image processing.

Computer vision is an important research area which has a wide range of applications. Different methods are used for the extraction of finding obstacles and distance to object determination from an image captured by the camera. Several approaches like stereo vision, motion parallax and monocular vision are the methods widely used for obstacle detection for the navigation of robot in an unknown environment.

Stereo vision and motion parallax are the techniques used for obstacle detection. But it requires finding similarities between points in multiple images separated over time. This process is realized by searching image similarities can be computationally complicated. It also requires high-quality images and known camera parameters.

Single-eyed vision can be used to determine the distance to obstacle and to navigate the robot. Monocular vision is vision in which each camera is used separately. The field of view is increased. Single-eyed vision type provides depth information when viewing a scene only with one camera.

One technique which is becoming important in the computer vision field is the image range finding, which is defined as the distance between the object into the scene and the image captured by the sensor. This method is focused in the precise estimation of the absolute distance, where are considered low-cost components such as a webcam and a laser pointer, both used to estimate distances obtained from the real environment.

There are numerious possible solutions of image processing algorithms based on range finder to evaluate the result.

Image processing is a form of signal processing where the input signals are images such as photographs or video frames. The output could be a transformed version of the input image or a set of characteristics or parameters related to the image. The computer technologies development that has taken place over the last 20 years has led to great advancements in the field of digital image processing.

There are two parts can be described on the proposal image processing algorithm for obstacle detection and distance measurement. First is the obstacle detection.

As it is difficult to distinguish the laser beam and white colors from other parts of image, for example – bright light, it is necessary to execute image segmentation.

Segmentation is a process of that divides the images into its regions or objects that have similar features or characteristics. Segmentation has no single standard procedure and it is very difficult in nontrivial images. The extent to which segmentation is carried out depends on the problem specification. Segmentation algorithms are based on two properties of intensity values - discontinuity and similarity. First category is to partition an image based on the abrupt changes in the intensity and the second method is to partition on the image into regions that are similar according to a set of predefined criteria.

- Read the Color Image and Convert it to Grayscale
- Use the Gradient Magnitude as the Segmentation Function
- Mark the Foreground Objects

- Compute Background Markers
- Visualize the Result

Second part of the algorithm is to find the distance between obstacle and laser emitter by using the laser range finder. Laser range finder device can directly give the distance between the obstacle and laser emitter. To find distance is right angle triangle technique can be applied.

- Edges found by looking neighboring pixels.
- Region boundary formed by measuring gray value differences between neighboring pixels

Ed= edge(I) takes a greyscale or a binary image I as its input, and returns a binary image Ed of the same size as I, with 1's where the function finds edges in I and 0's elsewhere. By default, edge uses the Sobel method to detect edges but the following provides a complete list of all the edge-finding methods supported by this function:

- The Sobel method finds edges using the Sobel approximation to the derivative. It returns edges at those points where the gradient of I is maximum.
- The Prewitt method finds edges using the Prewitt approximation to the derivative. It returns edges at those points where the gradient of I is maximum.
- The Roberts method finds edges using the Roberts approximation to the derivative. It returns edges at those points where the gradient of I is maximum.
- The Laplacian of Gaussian method finds edges by looking for zero crossings after filtering I with a Laplacian of Gaussian filter.
- The zero-cross method finds edges by looking for zero crossings after filtering I with a filter you specify.
- The Canny method finds edges by looking for local maxima of the gradient of I. The gradient is calculated using the derivative of a Gaussian filter. The method uses two thresholds, to detect strong and weak edges, and includes the weak edges in the output only if they are connected to strong edges. This method is therefore less likely than the others to be fooled by noise, and more likely to detect true weak edges.

One of possible solutions to detect obstacle and perform calculations of obstacle distance using laser beam, which was implemented in this work, is to create image processing algorithm which extracts brightest pixel points from web camera captured image. In facts these points represent position of laser beam in real environment. In order to distinguish the laser beam and white colors from other parts of image, it is necessary to execute image normalization, which realized by conversion to grayscale and RGB segmentation of obtained image. Using RGB values it is possible to detect the laser points by seeking pixels with brightest index – 255 for RGB color model.

Another important step is to evaluate position of brightest pixel on image frame. It can be defined as pixel index – certain number of pixels from left border of image. Result of image processing algorithm should be array of data with values of pixel indexes of brightest points and image which shows only extracted laser beam footprint on obstacle.

Final step is to implement mathematical algorithm to calculate distance based on data from image processing algorithm and geometrical conditions of web-camera and laser emitter using triangulation principles.

To make the design of the whole device, both the webcam and the laser pointer are setup on common holder with fixed position to simplify projection calculation. The holder should be designed according to shapes and dimensions of web camera and laser pointer. This detail can be printed using 3-D printer. The main advantage of this configuration (in fact 3-D scanner technology), is the use of low-cost components and the low computational cost. Additionally, an image processing technique should be implemented as improvement of obtained image to perform a correction for further distance determination. As a complementary part, the distance calculation must be evaluated on real time.

This work has been motivated by previous works of some researchers based on 3-D scanner and rangefinder techniques, which requires software application used computer vision and image processing algorithm.

The designed method is using LabView software for processing the images captured by camera. The whole system consists of a personal computer, laser beam emitter, and camera. From Fig. 1 it can be seen that camera and laser beam emitter are mounted on the holder, which can be connected to the robot, camera continuously capture images and transfer to personal computer for processing the corresponding image. The laser beam emitter creates spots on the obstacle, camera capture the image including laser beam spot. From the captured image designed software application identifies the obstacle and calculate how far it is from the robot.

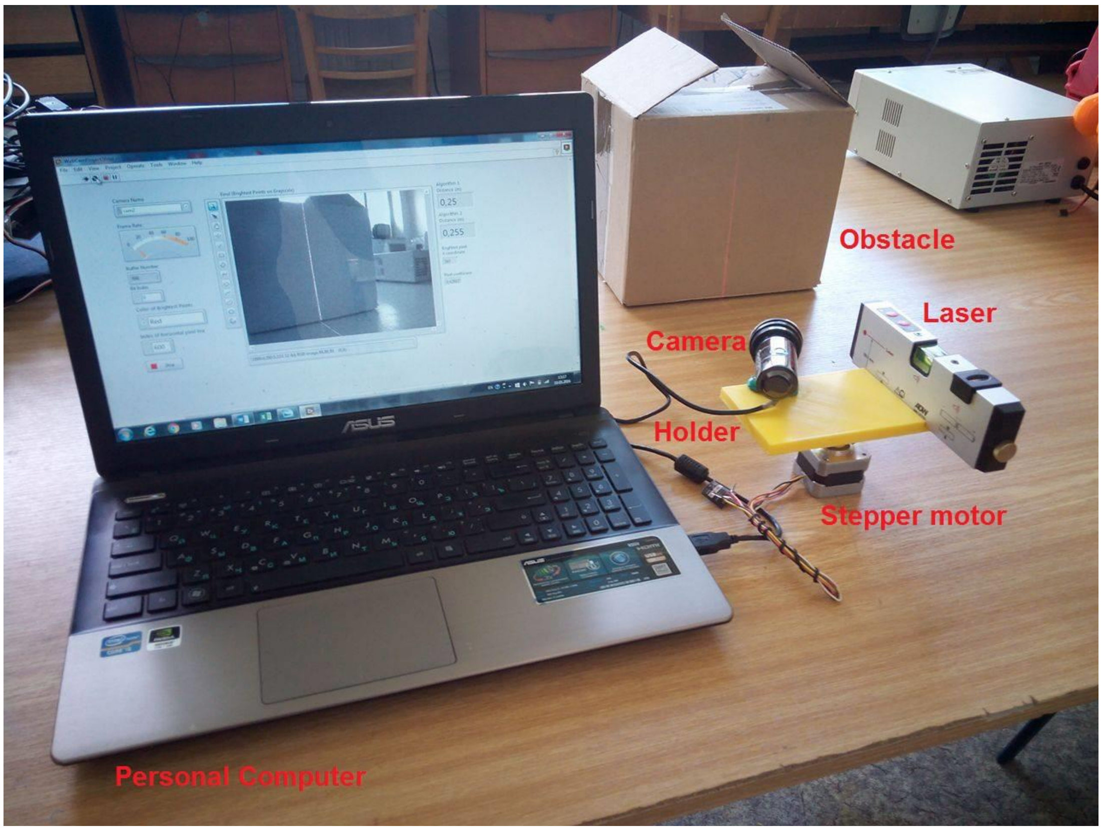
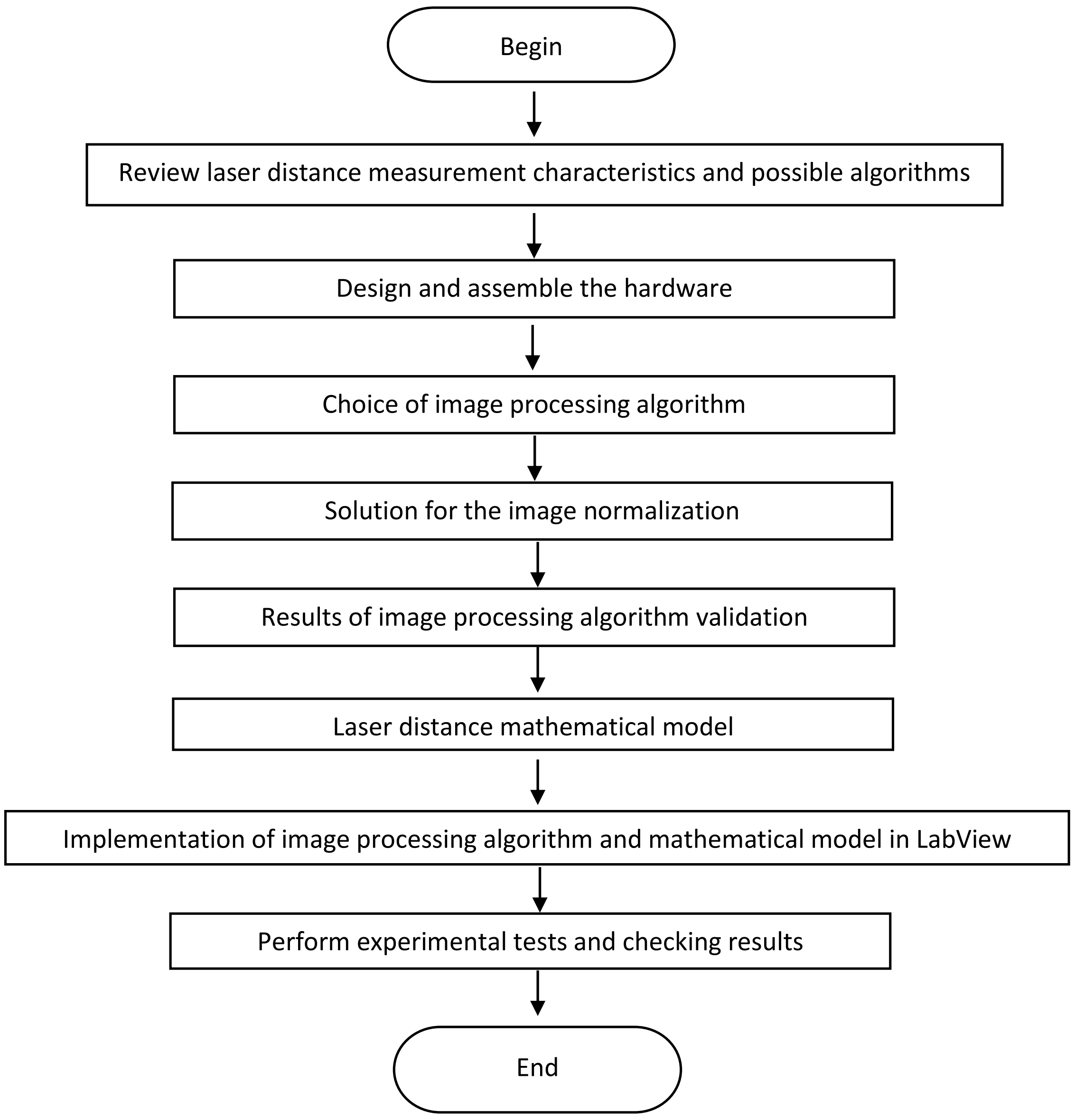


Fig. 1 - System of laser distance to obstacle determination

2.3 Algorithm of Project Design Flow


Begin
Review laser distance measurement characteristics and possible algorithms
Design and assemble the hardware
Choice of image processing algorithm
Solution for the image normalization
Results of image processing algorithm validation
Laser distance mathematical model
Implementation of image processing algorithm and mathematical model in LabView
Perform experimental tests and checking results
End

2.4 Literature Review

In the literature has been published several works related to distance determination with devices using a webcam and a laser pointer. Follow the brief description of the more representative previous works:

• Close-range camera calibration (1971) [1]. This work establishes the importance of photogrammetry close-range finding to obtain object measures, this led to more refined techniques of photogrammetry which were applied to obtain measurements of parabolic antennas.

• A 120x110 position sensor with the capability of sensitive and selective light detection in wide dynamic range for robust active range finding (2004) [2]. In this work, a 120x110 sensitive and light selective position sensor array with dynamic range is presented. The sensor can detect the position of a low-intensity light projected on non-uniform background illumination for a robust range finding system. At this work the authors have been achieved 3-D reconstruction of objects with scanners projections.

• Extrinsic self calibration of a camera and a 3D laser range finder from natural scenes (2007) [3]. This work describe a new approach for the extrinsic calibration of a camera with a 3D laser range finder, that can be done on the fly, this approach does not require any calibration object, only few point correspondences are used, which are manually selected by the user from a scene viewed by the two sensors. The extrinsic calibration is done using a nonlinear refinement process.

• Webcam telemeter (2007) [4]. T. Danko explains the importance of the design of low-weight devices, which can be used in applications where the weight is an important issue. This work describes how to configure a low-precision laser with a low-cost webcam in order to construct a vision machine with range information.

• Robust Range Finder Through a Laser Pointer and a Webcam (2011) [5]. Description of the robust estimation of absolute distance on images obtained from real time video sequences. Experimental tests were performed in order to demonstrate the efficiency of the distance calculation in real time through a geometric model and a simple system of linear regression.

• The Optical Measuring Device for the Autonomous Exploration and Mapping of unknown Environments (2012) [6]. Explanation of the developed rangefinder measurement device suitable for automatic mapping systems. This device uses a high-quality optical capture sensor and a laser beam. In article are presented the measurement results and the distance measurement algorithms based on image processing method – image subtraction. This system is mainly used for distance measurement, but it is suitable for automatic exploration and mapping systems too.

• The Laser Line Detection for Autonomous Mapping Based On Color Segmentation (2013) [7]. The article deals with the laser line detection based on the RGB segmentation and the component labeling. As a measurement device was used the developed optical rangefinder. The optical rangefinder is equipped with vertical sweeping of the laser beam and high quality camera. This system was developed mainly for automatic exploration and mapping of unknown spaces.

• A method for image processing and distance measuring based on laser distance triangulation (2013) [8]. The algorithm of laser triangulation distance measurement is applied through a laser and a CMOS camera, for an electronic virtual white cane.

• Obstacle detection for navigation of robot using computer vision and laser rangefinder (2014) [9]. The focus of the project is to propose one efficient obstacle avoidance algorithm for wheeled mobile robot using edge detection image processing technique and laser rangefinder.

3. Design and Construction of the Equipment

3.1 Required Conditions and Tools

Designed hardware system should allow to perform necessary actions for execution of distance determination algorithm. The main part of the system is device – fixed conjunction of web-camera, laser emitter by means of common detail (holder).

Device should be capable to scan the surrounding area and be able to measure the distance at any point along the laser line. It also requires connection with driven element which will provide rotation around the vertical axis of assembled device.

The distance between camera and laser, angles of their location relating to the front side of holder are important input parameters of further mathematical distance determination algorithm. So another requirement is that all elements of device should be well connected and fixed in stable position relating to each other.

Designed holder should be rotatable solid frame, where web-camera and laser will be assembled. Mounted device should be balanced and not overweighed, what can lead to a drop. As well whole device should be not big or cumbrous for further possible installation to mobile robot.

Required tools for whole hardware system:

• constructed device

• stepper motor or servo

• computer with software application

Required elements for device:

• USB web-camera with high-resolution of pixel-frame

• Laser emitter which provide strong laser beam

• Solid detail (holder for laser and web-camera)

Additional parts:

• Screws, hot glue

• AA batteries (for laser emitter)

• Stepper motor driver

3.2 Constructed Hardware System

Starting point of designing of device was decision how to assemble the web-camera and laser emitter in one solid construction with all necessary requirements and conditions. After reviewing of 3-D scanner principles and related projects it was chosen to design and create new detail – specially for this work and given elements.

Concept of the detail is based on the characteristics of two main components of the hardware system: web-camera and laser emitter. After assembling all these parts compose the device.

From Fig. 2 it can be seen that designed device consists of three elements: web-camera, laser emitter and common detail. Mounted device is connected to driven element – stepper motor, and to computer with software application by USB cable. The composed hardware system is possible to install to mobile robot.

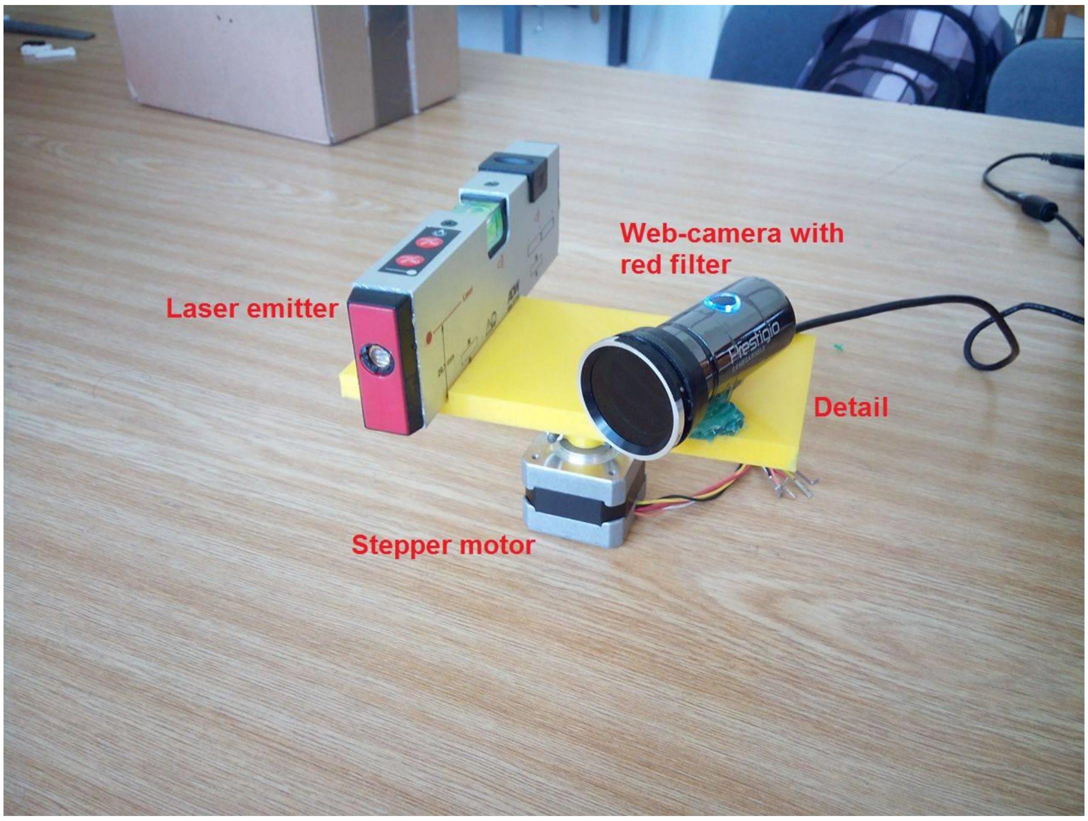


Fig. 2 – Designed device after assembling

In the project is used high-quality Prestigio color web-camera with resolution 1600x1200 pixels. The web-camera is covered by red light filter. In result the color red light image is obtained on the screen. Web-camera is connected and processed the image to computer trough USB cable. Camera does not require any additional power supply.

Designed system uses the laser emitter which spread out red laser beam into the line. It uses two AA batteries as power supply. In comparison with types of laser beam which focused only on one point, laser line allows to obtain the shape of obstacle in camera image.

The detail consists of two parts. First part (Fig. 3) is used as holder for elements of device: web-camera and laser emitter. Second part (Fig. 4) is used for connection with stepper motor. Both parts of the detail connected together in to entire structure using hot glue.

The drawing of designed detail was created in Autodesk Inventor software. Both parts of detail were printed using 3-D printer.

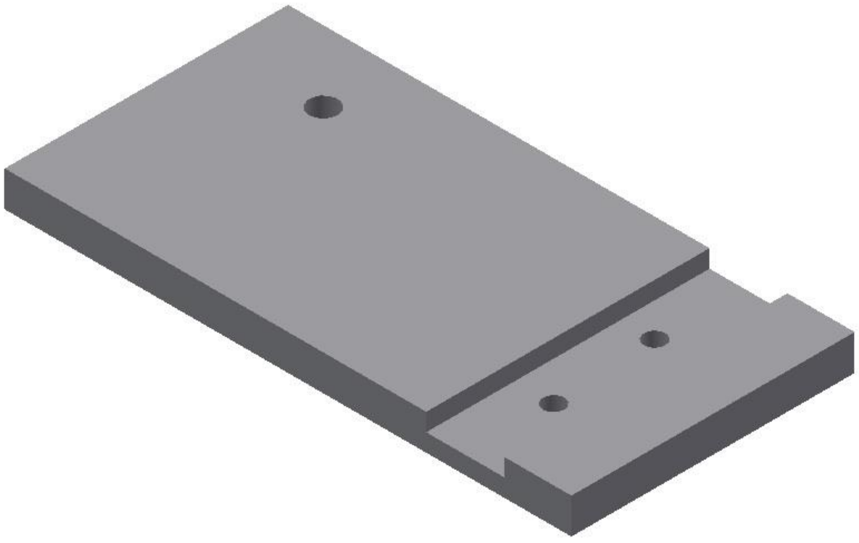


Fig. 3 - Autodesk Inventor drawing of holder for laser emitter and web-camera

Holder (Fig. 3) was designed in respect of geometrical dimensions and weights of laser emitter and web-camera. It has deepening and holes for mounting of elements. To provide the fixed position the laser and camera are tightly joined to the detail.

Laser emitter is mounted in to the deepening of holder with two screws from lower side. This position provides right angle between laser beam and front side of the detail, what is necessary for triangulation model.

Web-camera is connected to the holder by one screw and hot glue. Camera installed with tilt angle inward to the front side of laser. It allows to orient the camera image on the laser beam in surrounding area. This type of construction forms the condition – as close the obstacle to laser emitter, as close the object on the screen to right border of the image.

After assembling center of camera lens surface and laser eye should be located on the same horizontal line. This condition allows to obtain more precise result of measurement.

Holder with assembled components is connected through second part (Fig. 4) of detail to stepper motor, which provides the rotation of device.

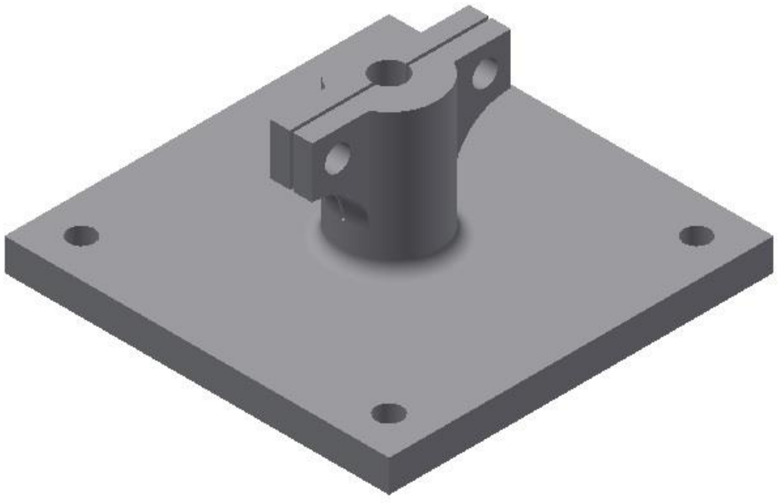


Fig. 4 - Autodesk Inventor drawing of part of detail for connection with stepper motor

4. Image Processing and Analysis

4.1 Image Processing Overview

Image processing is a method to convert an image into digital form and perform some operations on it, in order to get an enhanced image or to extract some necessary information from it. It is a type of signal distribution in which input is image, like video frame or photograph and output may be image or characteristics associated with that image. Usually Image Processing system includes treating images as two dimensional signals while applying already set signal processing methods to them.

It is among rapidly growing technologies today, with its applications in various aspects of industry and technologies. Image processing forms core research area within engineering and computer science disciplines too.

Image processing basically includes the following three steps:

- Importing the image with optical scanner or by digital camera.
- Analyzing and manipulating the image which includes data compression and image enhancement and spotting patterns that are not available to human eyes like satellite photographs.
- Output is the last stage in which result can be altered image or data report that is based on image analysis.

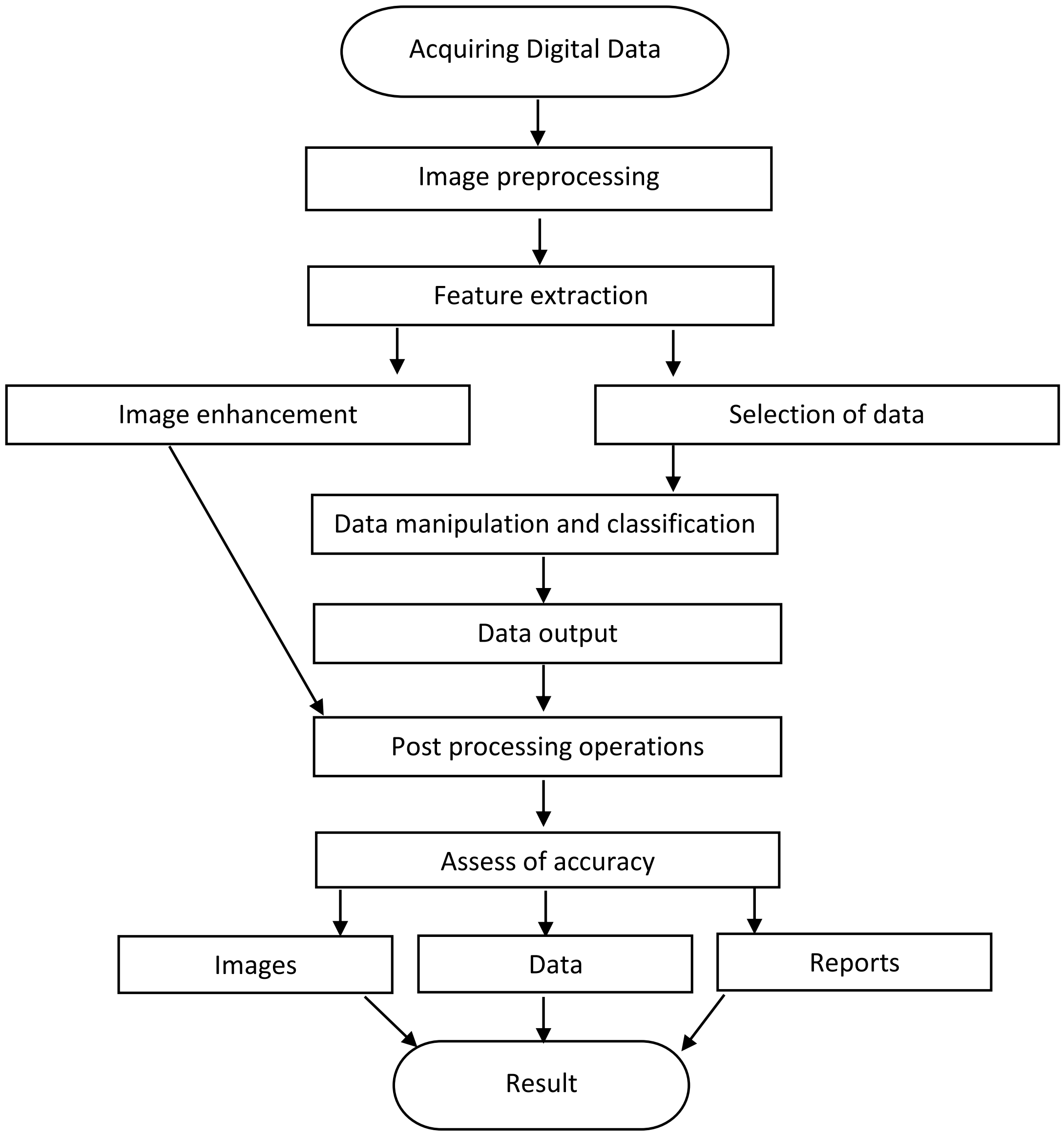
The purpose of image processing is divided into five groups. They are:

1. Visualization - observe the objects that are not visible.
2. Image sharpening and restoration - to create a better image.
3. Image retrieval - seek for the image of interest.
4. Measurement of pattern – measures various objects in an image.
5. Image Recognition – distinguish the objects in an image.

The two types of methods used for Image Processing are Analog and Digital Image Processing. Analog or visual techniques of image processing can be used for the hard copies like printouts and photographs. Image analysts use various fundamentals of interpretation while using these visual techniques. The image processing is not just confined to area that has to be studied but on knowledge of analyst. Association is another important tool in image processing through visual techniques. So analysts apply a combination of personal knowledge and collateral data to image processing.

Digital Processing techniques help in manipulation of the digital images by using computers. As raw data from imaging sensors from separate platform contains deficiencies. To get over such flaws and to get originality of information, it has to carry various phases of processing. The three general phases that all types of data have to go through while using digital technique: pre-processing, enhancement and display, information extraction.

4.2 Image Processing Standard Algorithm



4.3 Acquiring Image. Digital Representation.

In the first step, the color image is acquired by the web-camera with 1600x1200 pixel resolution.

For further image processing algorithm and analysis, it is necessary to interpret the image in some color space format. The most common are: RGB (red, green, blue) and luminance–chrominance formats.

The RGB color model is an additive color model in which red, green, and blue light are summarized together in various ways to reproduce an array of colors. The name of the model comes from the initials of the three additive primary colors: red, green, and blue.

The main purpose of the RGB color model is for the sensing, representation, and display of images in electronic systems, such as televisions and computers, though it has also been used in conventional photography. Before the electronic age, the RGB color model already had a solid theory behind it, based in human perception of colors.

RGB is a device-dependent color model: different devices detect or reproduce a given RGB value differently, since the color elements and their response to the individual R, G, and B levels vary from manufacturer to manufacturer, or even in the same device over time. Thus an RGB value does not define the same color across devices without some kind of color management.

A color in the RGB color model is described by indicating how much of each of the red, green, and blue is included. The color is expressed as an RGB three parts (red, green, blue), each component of which can vary from zero to a defined maximum value. If all the components are at zero the result is black; if all are at maximum, the result is the brightest representable white.

These ranges may be quantified in several different ways:

• From 0 to 1, with any fractional value in between. This representation is used in theoretical analyses, and in systems that use floating point representations.

• Each color component value can also be written as a percentage, from 0% to 100%.

• In computers, the component values are often stored as integer numbers in the range 0 to 255, the range that a single 8-bit byte can offer. These are often represented as either decimal or hexadecimal numbers.

• High-end digital image equipment are often able to deal with larger integer ranges for each primary color, such as 0…1023 (10 bits), 0...65535 (16 bits) or even larger, by extending the 24-bits (three 8-bit values) to 32-bit, 48-bit, or 64-bit units (more or less independent from the particular computer's word size).

In digital technique the number of available colors in an image — typically 256-cubed (8 bits in three color channels with values of 0–255)— although each color in the RGB24 format has only 8 bits representing 256 codes for each of the R, G, and B components. However, the advantage is that an indexed-color image file can be significantly smaller than it would be with only 8 bits per pixel for each primary.

By using an appropriate combination of red, green, and blue intensities, many colors can be displayed. Current typical vision devices use up to 24-bits of information for each pixel: 8-bit per component multiplied by three components (24bits = 256^3, each primary value of 8 bits with values of 0–255). With this system up to 16,777,216 unique combinations of R, G and B values are allowed, providing millions of different hue, saturation, and lightness shades. Increased shading has been implemented in various ways, among others using a fourth greyscale color channel as a masking layer - RGB32 format.

Using the RGB32 video format provides in modern processors faster access to video data because the data is aligned to machine's word boundaries. For this reason many applications use it instead of RGB24 even when there is no transparency mask information in the fourth (A) byte, since in general the improved processing speed outweighs the memory overhead introduced by the unused A byte per pixel.

All luminance–chrominance formats used in the different TV and video standards such as YIQ for NTSC, YUV for PAL, YD_BD_R for SECAM, and YP_BP_R for component video use color difference signals, by which RGB color images can be encoded for broadcasting or recording and later decoded into RGB again to display them. These intermediate formats were needed for compatibility with pre-existent black-and-white TV formats. Also, those color difference signals need lower data bandwidth compared to full RGB signals.

Similarly, current high-efficiency digital color image data compression schemes such as JPEG and MPEG store RGB color internally in YC_BC_R format, a digital luminance-chrominance format based on YP_BP_R. The use of YP_BP_R also allows to perform loss subsampling with the chrome channels (typically to 4:2:2 or 4:1:1 ratios), which it aids to reduce the resultant file size.

In this work used RGB32 format where each pixel of the image contains one byte for each of the red, green and blue components plus an additional byte for transparency mask (A component) in successive places of memory. Since one byte occupies 8 bits, the total number of bits consumed by one pixel is 4*8 = 32 and thus the 32 at the end of the format's name. Hence the values 0-255 (0 – darkest index, 255 – brightest index) will be used for component indexing and data operations.

4.4 Image Preprocessing. Conversion to Grayscale

In algorithm phase of image preprocessing, the newly acquired camera image is prepared for further analysis. The majority of this process is converting the color image from RGB to grayscale.

The color segmentation during image processing can be the most crucial part of image processing algorithm based on laser line detection. Correct segmentation influent the distance measurement accuracy. The laser emits light in the red part of spectrum, thus the image representation is separated into three color components R, G, and B. The red channel also represents the laser light. Other components are important for the white light background estimation. The laser footprint separation from the background can be done by thresholding in each color component.

To obtain better result of color segmentation and indexing components it is necessary to perform image normalization. It can be performed by image gain correction in R component of the image with respect of color of the laser.

In our system we have web-camera with red-light filter and red light laser emitter. To obtain normalized image with avoidance of red color it was decided to apply conversion of color image to grayscale format. In this solution the footprint of laser beam in image will be in form of white (brightest) color.

In the RGB format, every pixel is represented by a level of each of the three primary colors: red, blue and green. In the grayscale representation, every pixel would only be represented by one gray level. The grayscale converted image carries only intensity information. Images of this sort are composed exclusively of shades of grey, varying from black at the weakest intensity to white at the strongest. The image contrast enhancement has improved the perceptibility of objects in the scene followed by RGB to gray conversion.

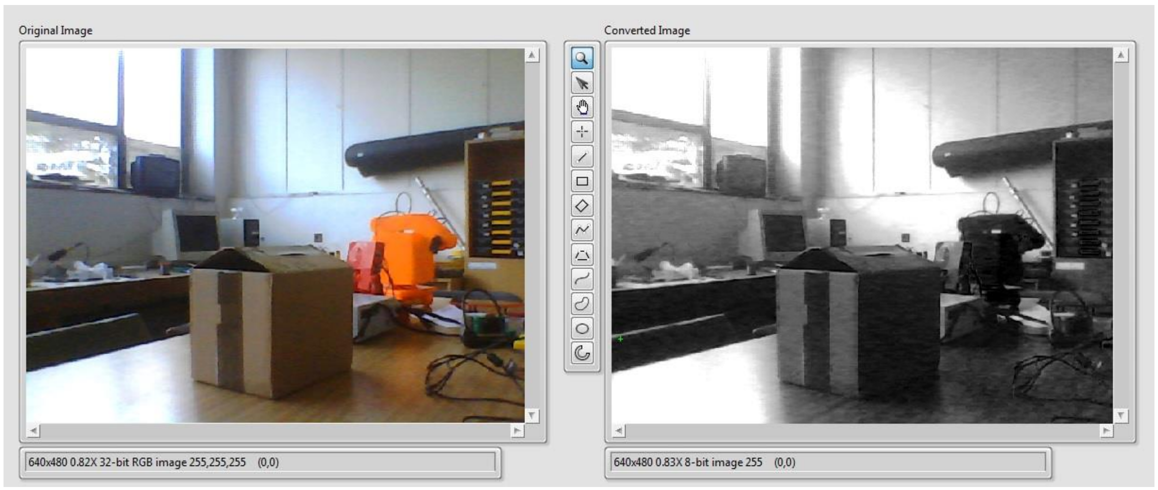

Original Image
Converted Image
640x480 0.82X 32-bit RGB image 255,255,255 (0,0)
640x480 0.83X 8-bit image 255 (0,0)

Fig. 5 - Color image in RGB format and its grayscale conversion

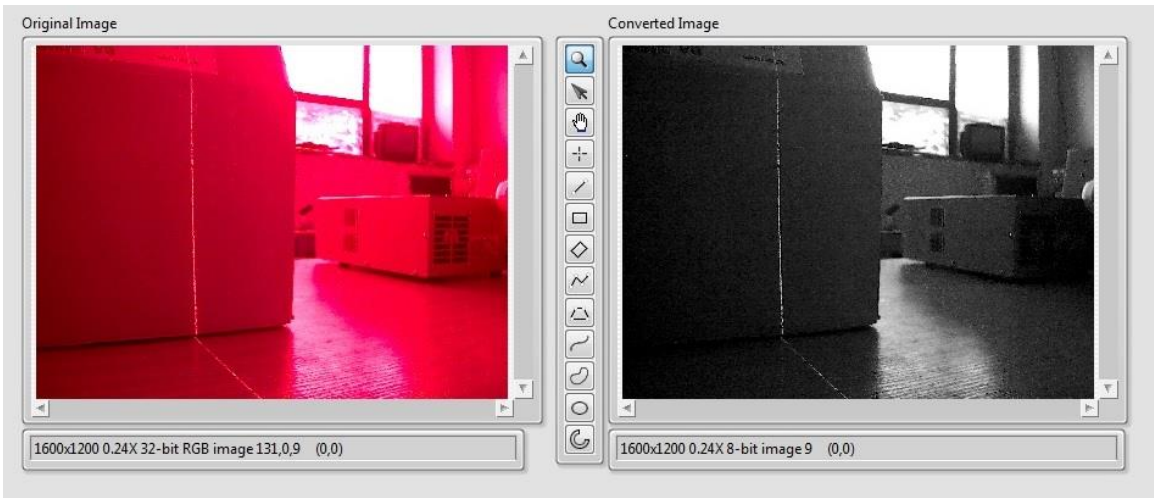
On Fig. 5 it can be seen the result of color RGB image conversion to grayscale.

Each pixel of the RGB image can be broken down to a set of three 8 bit values, with each value corresponding to the amount (intensity) of red, green and blue. Each color component of the image is referred to as an image plane or channel. Reducing each pixel to a single value from the original set of three values shrinks the number of operations by two thirds and further simplifies image processing. This process is known as converting to grayscale, as the resulting image pixels will be stored as a single 8-bit number, often represented by a gray value between black and white.

There are several methods used to convert an RGB image to grayscale. A simple method is to add up 1/3 of each color value for the new grayscale intensity. However, this method is rarely used since different colors appear to have different brightness to the human eye. For example, the NTSC television standard uses the formula Y = 0.299R + 0.587G + 0.114B to convert the RGB color space into a single luminance value. For another point of comparison, Intel's Image Processing Library uses the formula Y = 0.212671R + 0.715160G + 0.072169B.

In this work conversion of color image to grayscale is realized by converting the image to an array and then performing array operations to retain eight of the original thirty-two bits of the image.

In result we obtain normalized image (Fig. 6 - Result of grayscale conversion with designed hardware system and array of data, which can be used for further data operations in image processing algorithm.



```
1600x1200 0.24X 32-bit RGB image 131,0,9    (0,0)
1600x1200 0.24X 8-bit image 9    (0,0)
```

Fig. 6 - Result of grayscale conversion with designed hardware system

4.5 Image Processing. Laser Line Detection

After the image preprocessing step, the source image has been already converted to grayscale. Next phase is image processing part, which will distinguish laser beam from environment in the converted image.

The most important step in the presented image processing algorithm is the laser footprint detection in the captured frame.

This can be done by several methods. The position of the laser beam can be determined, for example, from one image by the color segmentation followed by thresholding or by other image processing approaches.

Second approach represents mostly the background estimation methods, which use difference between two or more images. The first image is taken with a laser point on. The background is then estimated from the one or more images taken with laser beam off. The resulted image is disturbed by noise and the chip saturation. The noise can be removed by Gaussian filter or by thresholding. But after thresholding the edges are not solid.

Alternative possible solution is based on mathematical methods. The technique of the fuzzy weighted averaging filtering is used to remove the noise and enhance detection sensitivity for low energy lasers. The system also uses the image subtraction.

Problem point in the routine of image processing, the application must recognize which pixels might correspond to laser. At the end of this process, the grayscale image will be converted to a binary representation where white pixels represent laser footprint and black pixels represent other space. Two popular approaches to achieve this result are to use an edge detector, such as a Sobel or Canny, or to apply a pixel intensity threshold. Results of similar projects has shown the edge detector methods to be unreliable and sensitive to noise, so it was decided to use the pixel intensity approach.

A simple approach to performing an intensity threshold is to set an intensity level where brighter pixels are considered part of a line and darker pixels are environment. The intensity threshold method assumes that the laser line will generally be the brightest part of the image.

While successful for many other conditions, the threshold approach gives wrong result when the line is not the brightest object in the image.

The weaknesses in the threshold approach can be decreased by reducing the effect of large, bright objects. One way to do this is to take each row of an image and only accept the one brightest (highest intensity) pixel. Using this method, high intensity areas, will only affect the rows in which they are present. Also of importance, this method prevents the number of falsely detected pixels from being large in comparison to the number of pixels correctly detecting lines. Unless there are many false high intensity areas in the image, most of the line will still be detected. This algorithm of focusing on each row independently and choosing the brightest pixel came to be known as “brightest pixel thresholding” or simply “brightest pixel per row”.

In this work was applied image processing algorithm called “brightest pixel per row”, which extracts brightest pixel points in every pixel row of the captured image. Using index values of RGB color model it is possible to detect the points of laser footprint by seeking pixels with brightest RGB index – 255.

As our web-camera has resolution 1200x1600 pixels, the image contains 1200 pixel rows. So in result it will be data array of 1200 numbers of the brightest points with RGB indexes of brightness.

To distinguish the laser points from other possible sources with same brightness – light or white color, all other indexes except brightest (with RGB value – 255) are set to 0. After this operation, array will contain data with only two possible values – 255 (brightest points - laser) or 0 (black color which represents other environment).

Second important step is to evaluate position of brightest pixel point on image frame. It can be defined as pixel index (ppx) – certain number of pixels from left border of image (in fact x-coordinate). According to resolution of used web-camera pixel index can be evaluated as number 0…1599 (0 – left border of the image, 1599 – right border).

Using data of RGB values of brightest points per row and its pixel indexes it is possible to obtain new image with extracted laser points based on converted grayscale image.

On Fig. 7 it can be seen converted grayscale image of captured image by web-camera and binary output image obtained using this brightest pixel threshold method, which represents extracted laser line.

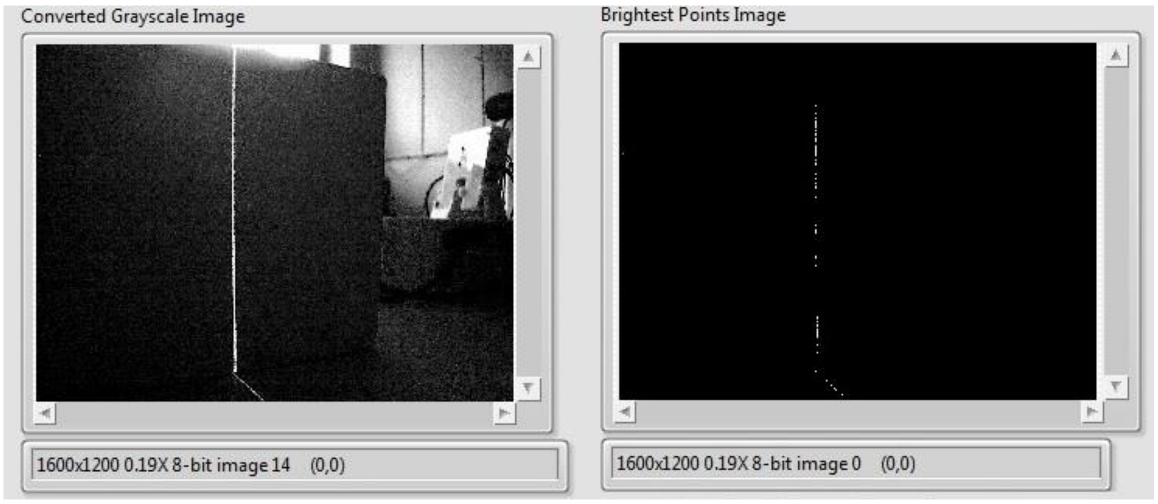

Converted Grayscale Image
Brightest Points Image
1600x1200 0.19X 8-bit image 14 (0,0)
1600x1200 0.19X 8-bit image 0 (0,0)

Fig. 7 – Laser line extraction in image processing algorithm

Result of image processing algorithm is array of data with values of pixel indexes of brightest points and image which shows only extracted laser beam footprint on obstacle.

4.5 Image Post Processing. Overlaying of image

Image post processing techniques are used to edit obtained images and data for better accuracy of results or to add some additional features and functions.

To obtain in result the image with extracted laser line and environment at one frame it was implemented overlay of images in image processing algorithm.

Firstly, it colors in red color areas of the threshold binary image – brightest points image. Second step - to overlay it on the grayscale image.

In result final image consists of grayscale image and red threshold image with laser line footprint (Fig. 8).

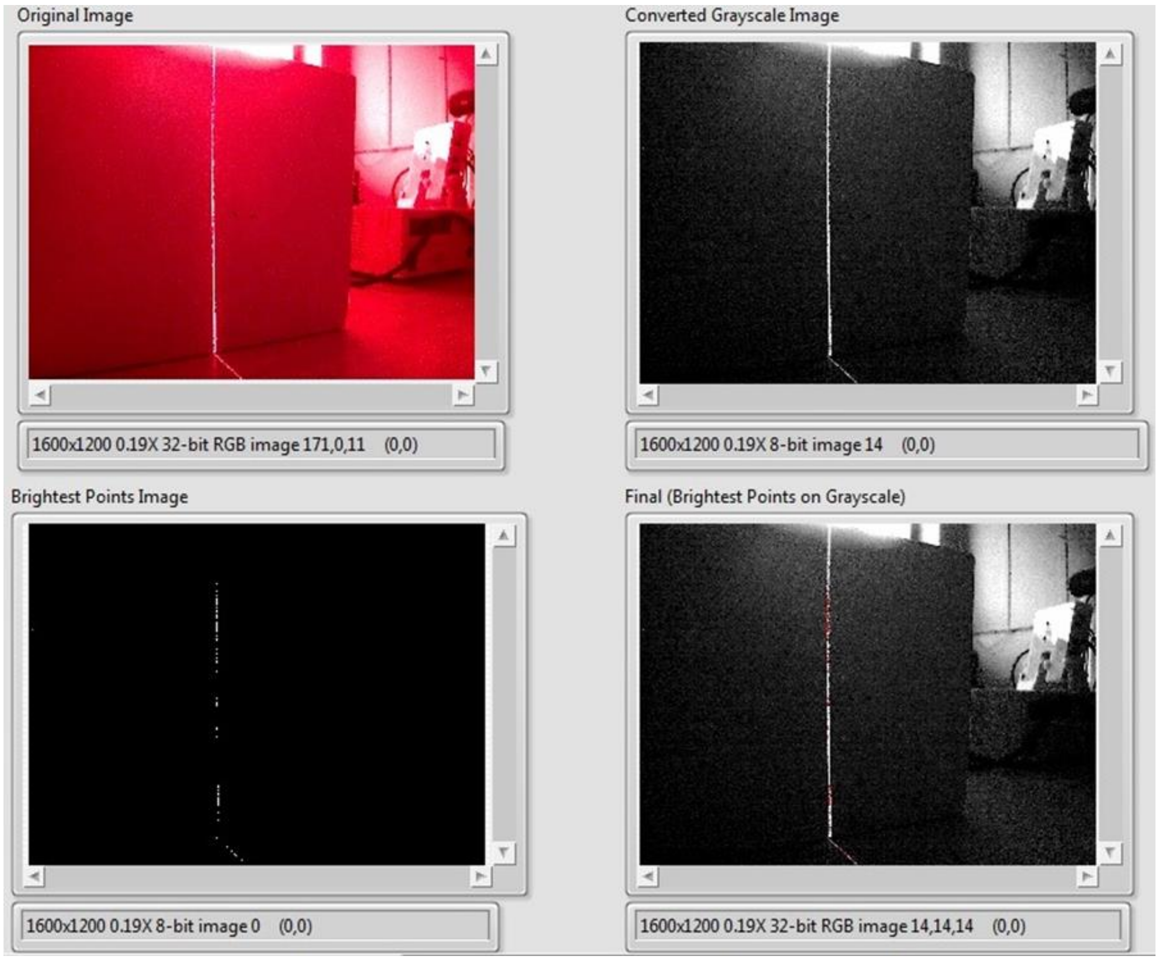

Original Image
1600x1200 0.19X 32-bit RGB image 171,0,11 (0,0)
Converted Grayscale Image
1600x1200 0.19X 8-bit image 14 (0,0)
Brightest Points Image
1600x1200 0.19X 8-bit image 0 (0,0)
Final (Brightest Points on Grayscale)
1600x1200 0.19X 32-bit RGB image 14,14,14 (0,0)

Fig. 8 - All images obtained during image processing algorithm

5. Algorithms of Determination the Distance with Laser and Web-camera

5.1 System Overview

The systems for distance evaluation, which uses camera, provide fast contactless measurement for many points and edges simultaneously in real time. Obtained results can be used for further image processing, calculations or algorithms.

Measurement systems, which don't require laser beam: single camera or systems with two cameras can evaluate distance, but have limitations and disadvantages.

Solution with single camera can be applied to measure the distance only with knowledge of possible detected object or obstacle. It is more appropriate to use it for recognition purposes.

Distance measurements with two cameras can create stereoscopic system requires multiple cameras with same characteristics and constant system parameters. Measurements realized through angle detection.

Algorithms for measuring distance based on laser emitter and web camera are more precise. These methods have several advantages in comparison with other camera-based systems: accuracy, focusing on laser beam to mark the obstacle. There are some possible math solutions to obtain distance in result: triangulation, calculating of phase shift, measuring of laser beam propagation.

For our conditions and purposes, algorithms based on triangulation method are the most suitable. Further described solutions uses data from image processing of laser beam emitted into line or another shape, depending on size and form of obstacle. These calculation algorithms are able to measure the distance at any point along the laser line in surrounding area.

After detection the laser points on the screen obtained from camera image, it is needed to provide a mathematical model to determine the relation between positons of laser brightest points on the screen and the real distance between mounted device with camera and obstacle.

Because of vertical spreading of laser beam, it is possible to calculate distance to obstacle at any point of vertical axis of image. Our image processing algorithms determines as a result an array of horizontal coordinates of brightest point in every row. This conditions allow to apply mathematical algorithms for the mapping the environment space in front of web camera.

As a main goal of work – to determine distance to obstacle, we use the data from one central horizontal row of image to perform the calculations. In result, we obtain accurate single number, which indicates distance to obstacle from laser emitter. This value changing in real time if object or frame with laser and camera moves, showing actual distance.

For mapping the ambient space or for data calibration of obstacle distance it is possible to apply in mathematical algorithms the coordinates of brightest point of any row from image.

5.2 Mathematical Algorithm 1

First mathematical algorithm for measuring distances between obstacle and device based on horizontal opening angle of web camera and pixel ratio of laser point position in obtaining image.

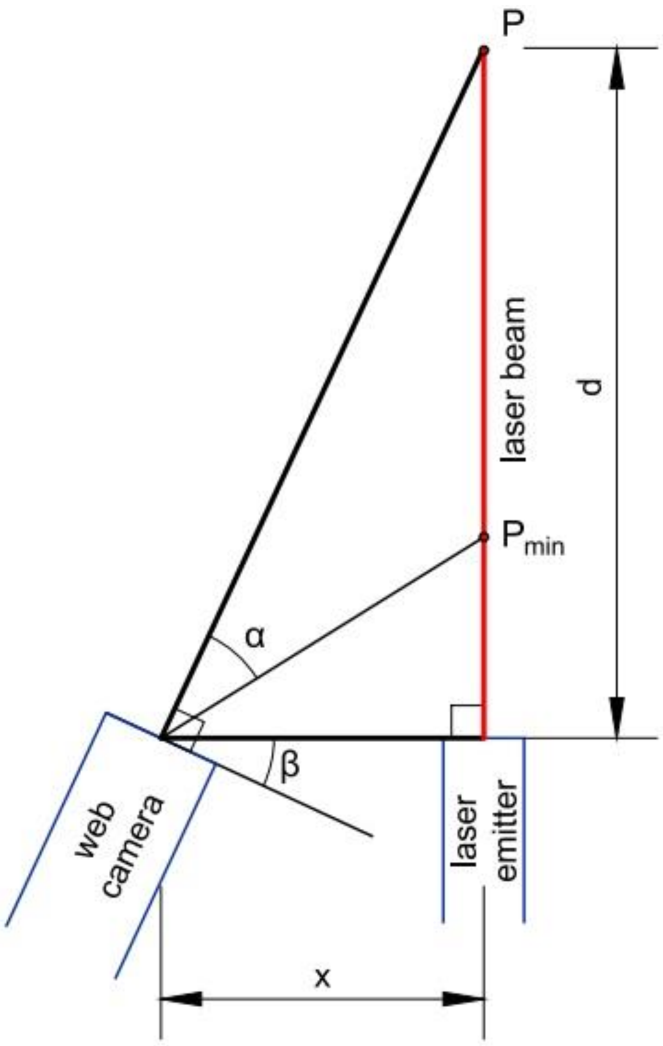


Fig. 9 - Mathematical model for Algorithm 1

α – horizontal opening angle, °

β – tilt angle, °

x – distance between center of web-camera lens and laser eye emitter, m

d – distance between front surface of laser emitter and obstacle, m

P – position of the detected obstacle

P_{min} – closest possible position for obstacle to be detected by camera

As web camera and laser emitter are mounted on the solid rotatable frame, we can define constant parameters. The laser pointing forward at 90° to the frame. The web camera is tilted inward. Values x and β are known by conceptual design and assembling of the frame (x=0.08m, β=25°) . Value of angle α was defined experimentally (α=32°).

$$d_{min} = \frac{x}{tan(\alpha) + tan(\beta)} \tag{1}$$

According to the geometrical parameters of this model, the minimum distance to obstacle to detected by camera can be calculated by (1). For our values d_{min} = 0.071 (m).

Equation (1) does not contain any parameter related to pixel position of laser spot on the image. Implementing pixel coefficient to this triangulation model it is possible to calculate obstacle distance with respect of laser dot index on the obtaining image from camera.

$$d = \frac{x}{(tan(\alpha) + tan(\beta)) - 2 * p * \ tan(\alpha)} \tag{2}$$

Determination of distance between the obstacle and laser emitter [8]. The distance can by calculated by (2).

p – is an auxiliary ratio variable that represents the proportional value that the brightest pixel to one image border compared to entire horizontal pixel length of image.

$$p = \frac{\Delta l}{l} = \frac{l - ppx}{l} \tag{3}$$

ppx – pixel index of brightest laser point in x-axis row in camera image (0…1599).

l – entire number of pixels in row.

Value of l defined by camera frame parameters. In our case l = 1600 pixels.

For the proposed math model should be respected:

$$0 < p \ll 1 \tag{4}$$

Due the conditions of mounting frame with laser pointer and web camera – as close object to laser emitter in real area, as laser point on the image closer to right border of image frame. So, as bigger value of pixel index (ppx) of laser point as less real distance (d). Either, as smaller value of p as larger distance to obstacle.

After defining coefficient p it is possible to check the value of d_{min} by equation (2) – applying in formula the largest possible value of ppx = 1599, the closest possible obstacle position on the screen. In this case d_{min} = 0.07.

Comparing this result with value of d_{min} obtained from equation (1), it can be seen that they are equal.

By series of experiments and calculation based of equation (2) it is possible to calculate more precise values of horizontal opening angle (α=32.3°) and tilt angle (β=25.7°). Which is more complicate to measure by protractor with such accuracy.

Fig. 10 shows overall non-linearity of the system, which most affects the measuring process on the large distances.

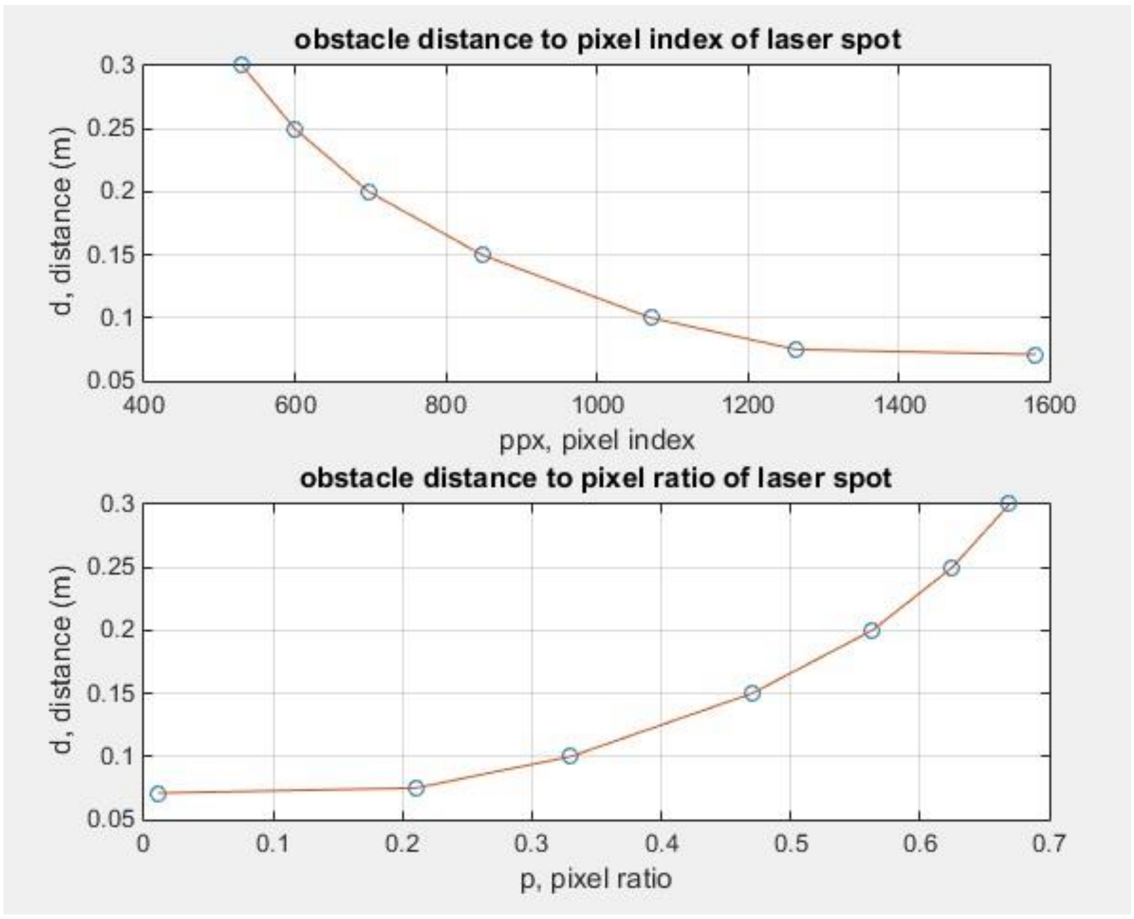


Fig. 10 - Distance to obstacle (m) to pixel index and pixel ratio of laser point

On Fig. 10 the x-axis represents ppx - the horizontal pixel number laser point and p – pixel ratio. The ordinate axis corresponds to the calculated real distance in meters between obstacle and laser emitter.

Disadvantage of this algorithm is that it is complicate to measure accurate value of horizontal opening angle (α) without series of experiments and data calibration. Also this mathematical model does not allow to determine values on small distances – less than d_{min}, because of geometrical limitations.

Actual and calculated distance to obstacle

ppx, index of laser spot on the screen	Calculated d, (m)	Actual d, (m)	% of error
530	0.301	0.3	0.3
600	0.248	0.25	0.8
698	0.199	0.2	0.5
848	0.154	0.15	2.6
1072	0.105	0.1	5
1264	0.079	0.075	5.3
1582	below limit	0.048	below limit

Table 1 – Results of Algorithm 1

Table 1 shows the comparison of real distance to obstacle with calculated by first mathematical model. It can be seen that this algorithm evaluates more accurate result with distances larger than 0.15 m – error is less than 1%. So the model calculations are precise.

5.3 Mathematical Algorithm 2

Second mathematical model for calculating distances between obstacle and device based on pixel angle.

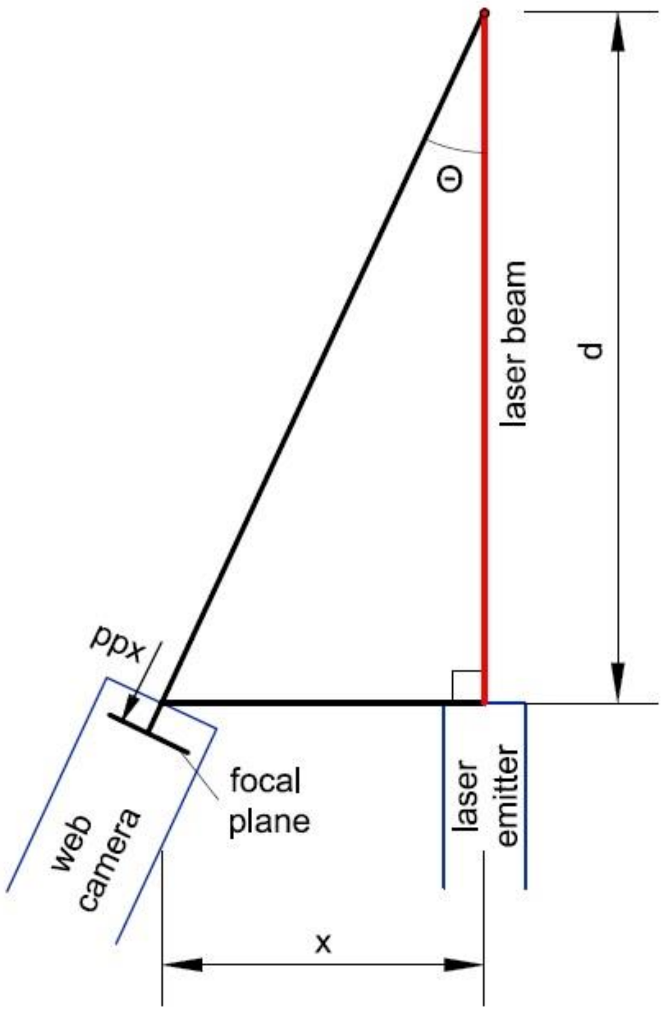


Fig. 11- Mathematical model for Algorithm 2

x – distance between center of web-camera lens and laser eye emitter, m

d – distance between front surface of laser emitter and obstacle, m

ppx – pixel index of brightest laser point in x-axis row in camera image, (0…1599)

θ – angle between laser beam and focal plane, °

Fig. 11 presents main triangle determined by laser emitter, laser spot on the obstacle and the camera focal plane. The laser beam is spread with 90° angle to the axis of front surface of laser emitter. Value of x is constant and known from mounting of frame (x=0.08m).

Fig. 12 shows triangle that formed on the focal plane. It is determined by the surface of camera sensor chip, line from the right border of chip, which is ideally parallel to laser beam emitted from laser pointer and line from ppx – pixel index of brightest spot, where the laser beam falls on camera chip.

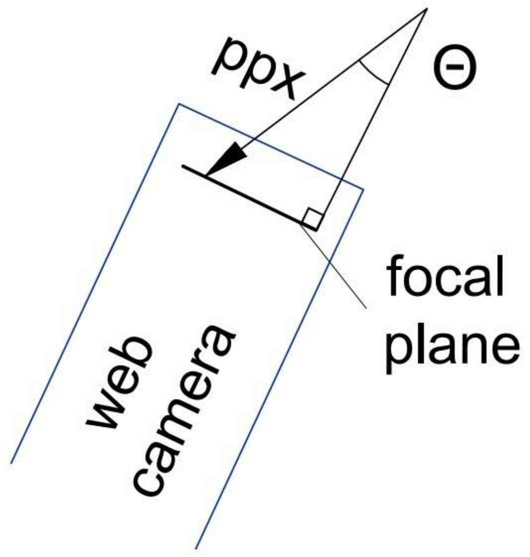


Fig. 12 - Triangulation principle on camera chip

In fact, the main triangle can be described from image on focal plane. Angle θ is equal to angle between pixel index and right border of image frame.

$$d = \frac{x}{tan(\theta)} \tag{5}$$

$$tan(\theta) = ppx * rpc + ro \tag{6}$$

Equations (5) and (6) determine the calculation of distance to obstacle using angle θ [4].

Equation (5) described like simple trigonometrical definition of tangent – ratio between opposite leg to adjacent leg. In our case it is ratio between x – distance between center of web camera lens and laser eye of emitter and d – distance between front surface of laser pointer and obstacle.

Equation (6) shows how it is possible to evaluate angle θ through parameters of camera sensor chip. ppx – pixel index of brightest laser point in x-axis row in camera image. This parameter are changing if either frame with laser or obstacle moves. Parameters rpc – radians per pixel pitch and ro – radian offset, are constant and unique for each camera. The values of these parameters for our web camera obtained in experimental way (rpc = 0.044, ro = -8.768).

$$d = \frac{x}{tan(ppx * rpc + ro)} \tag{7}$$

Equation (7) determines the final solution for calculation of distance from laser emitter to obstacle.

Because of construction of frame with web camera and laser pointer, we know that as close object to the laser emitter in real environment, as laser point on the image closer to right border of image frame. So, as bigger pixel index (ppx) of laser point as less real distance (d). Experimental results and calculations according to formula (7) proves it.

Obstacle distance dependence on pixel index (ppx) and angle θ is described by Fig. 13.

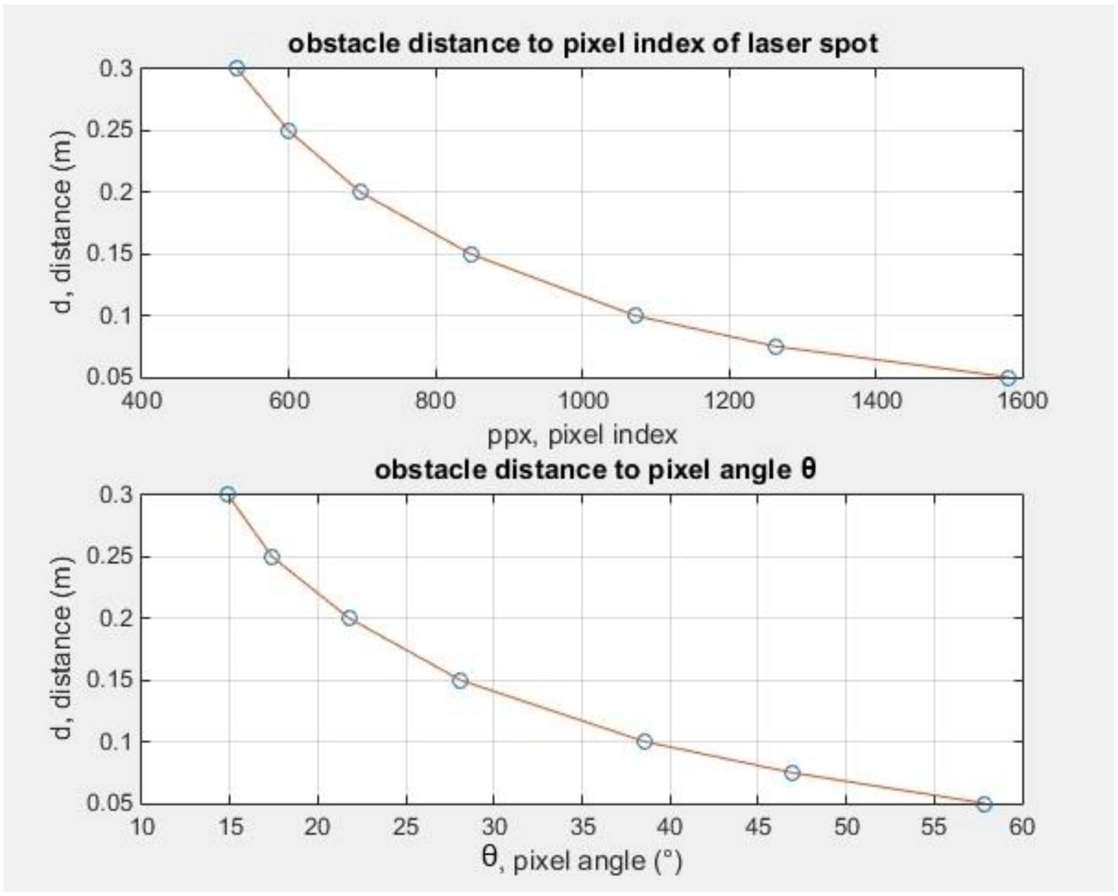


Fig. 13 - Distance to obstacle (m) to pixel index of laser point and pixel angle θ

As it can be seen from Fig. 13, as less pixel angle θ, as large distance to obstacle. It also shows non-linear dependence of obstacle distance on pixel index of laser point and pixel angel parameters.

Because of geometrical conditions of assembled frame with laser emitter and pixel resolution – 1600 pixels per row we should respect:

$$\theta < 60^{\circ} \qquad (8)$$

So it is possible to define the minimal distance to obstacle which system based on second algorithm can measure: $d_{\mathbf{min}}$ = 0.047 (m).

For checking experimental results or data calibration, it is possible to calculate actual angle θ, according to main triangle from Fig. 11 and equation (5) :

$$\theta_{actual} = arctan\left(\frac{x}{d_{actual}}\right) \qquad (9)$$

Equation (9), based on values x – distance between center of web camera lens and laser eye emitter and d – actual measured distance to obstacle for each data point.

The next step is find out how angle θ_{actual} from equation (9) relates to the pixel index (ppx) in the web camera image from equation (7). It is necessary to define the values of radians per pixel pitch of the web camera (rpc) and the radian offset (ro). As it was mentioned, these parameters constants for a particular camera and derived in experimental way from calibration data from equation (9).

Radians per pixel pitch (rpc) = 0.0439

Radian offset (ro) = -8.7683

Fig. 14 shows the linear relationship between θ_{actual} and pixel index (ppx).

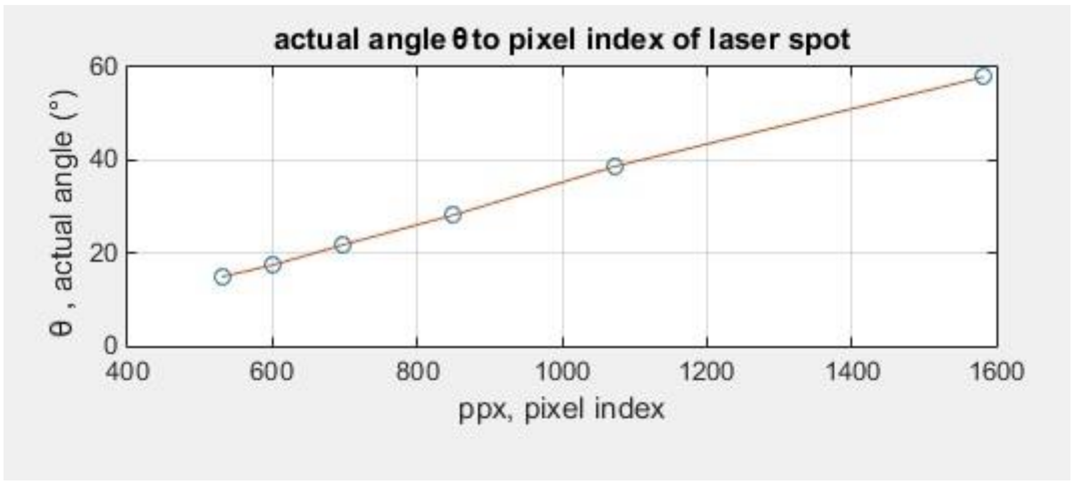


Fig. 14 - Actual pixel angle to pixel index

If we put the trendline through the points in Fig. 14, it is possible to define the linear equation of this line:

$$f(ppx) = 0.0439 * (ppx) - 8.7683 \qquad (10)$$

Comparing the coefficients from linear equation (10) with the values of radians per pixel pitch (rpc) and radian offset (ro) we can find them equal.

So, we can implement calculation based on equation (7) to obtain precise value of obstacle distance.

Table 2 shows the comparison of results of calculating obstacle distance (d) obtained from main equation (7) with actual measured distance to obstacle.

Actual and calculated distance to obstacle

ppx, index of laser spot on the screen	Calculated d, (m)	Actual d, (m)	% of error
530	0.308	0.3	2.6
600	0.251	0.25	0.4
698	0.198	0.2	1
848	0.147	0.15	2
1072	0.101	0.1	1
1264	0.075	0.075	0
1582	0.047	0.048	2.1

Table 2 – Results of Algorithm 2

It can be seen from Table 2 that result from mathematical model evaluates the accurate values of distance to obstacle. The error in comparison is less than 3%.

Advantage of second algorithm is that it allows to define more accurate result on small distances to obstacle – less than 0.01 m. This system also has larger range to evaluate distance, as d_{min} less than in first mathematical model (0.047 m < 0.071 m). For other distances – results calculating by two algorithms for same actual distance are differs no more than 0.007 m.

6. Software Design

6.1 Software Overview

Programming application, which performs image processing and mathematical algorithms for calculation the distance from laser emitter to obstacle, was designed in LabView National Instruments software.

LabView (Laboratory Virtual Instrument Engineering Workbench) is a system-design platform and development environment for a visual programming language from National Instruments. In comparison with other possible solutions, for example – MatLab, Python or C++, LabView provides more simple user interface and debugging opportunities, allowing development to focus on processing data.

The application was written using National Instruments LabView version 14.0, which creates programs using the G - dataflow programming language. LabView was initially designed to allow engineers with little programming experience to interface with data acquisition systems and perform analysis. This ability led to the creation of a programming environment that simplifies communication with external devices and allows easy creation of graphical user interfaces with familiar mechanical shapes and knobs.

Applications, known in LabView as “Virtual Instruments” or VIs, are written by laying elements on a block diagram and connecting inputs and outputs rather than writing lines of code. The inputs and outputs of the block diagram are linked to the controls and indicators on the user interface, called the front panel.

An example of LabView VI block diagram is shown in - LabView Block Diagram Fig. 15 with corresponding front panel shown in Fig. 16.

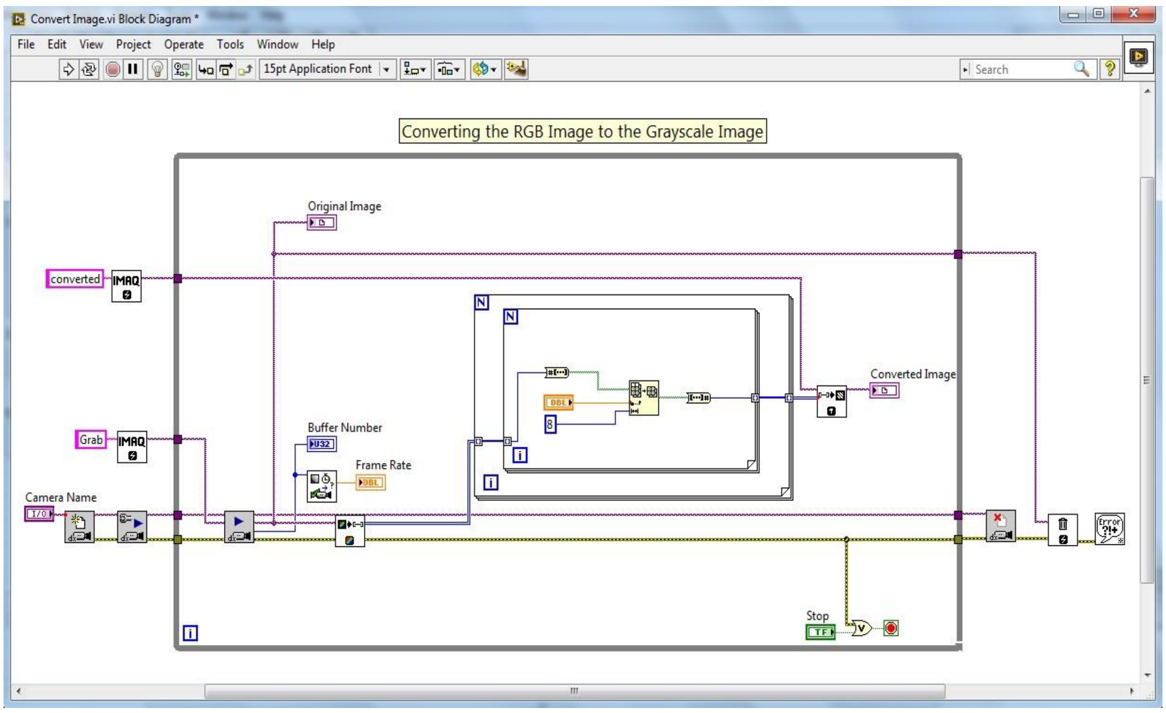


Fig. 15 - LabView Block Diagram

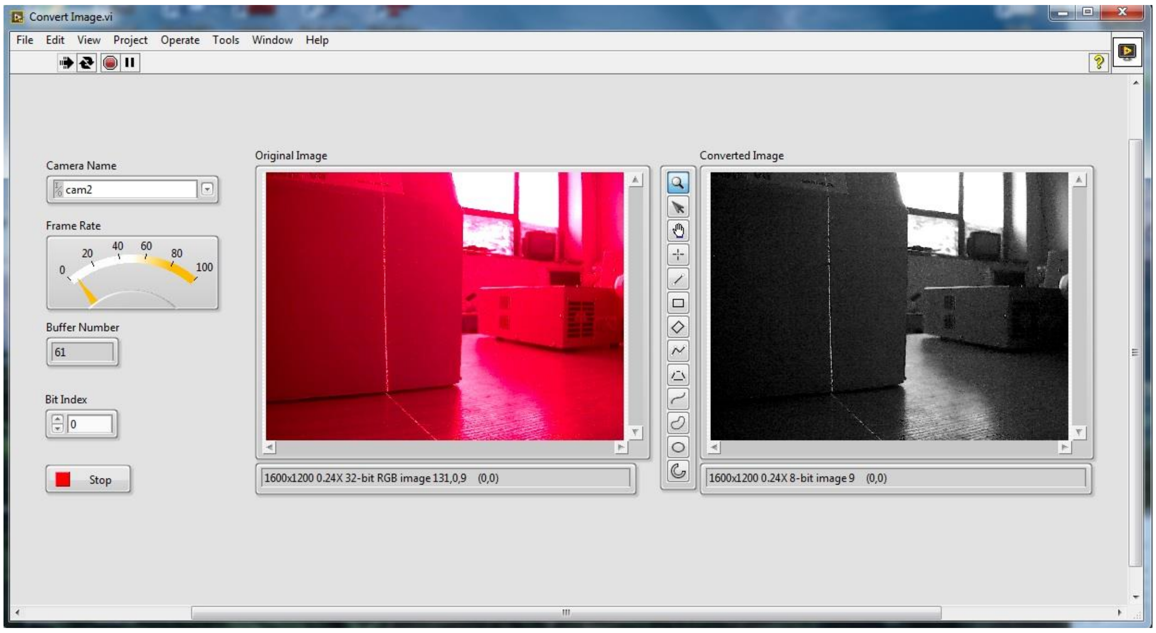

Convert Image.vi
File Edit View Project Operate Tools Window Help
Camera Name
cam2
Frame Rate
Buffer Number
61
Bit Index
0
Stop
Original Image
Converted Image
1600x1200 0.24X 32-bit RGB image 131,0,9 (0,0)
1600x1200 0.24X 8-bit image 9 (0,0)

Fig. 16 – LabView Front Panel

From its simple beginnings, LabView has evolved into a fully functional programming language while still retaining an easy user interface. Programming with a block diagram approach allows the programmer to focus on dataflow, rather than syntax. This way also makes it simple to create multithreaded parallel tasks since LabView will execute all subroutines that have all necessary inputs in parallel.

LabView was created for data acquisition, making it easy to read sensors connected to the computer by serial or firewire. The biggest advantage to using LabView is its graphical nature, allowing users to see the state of any variables while the program is running. A rich library of indicators allows quick implementation of charts and visualization of images or arrays. Using these powerful indicators, debugging programs becomes easier, reducing development time.

LabView is well suited for image processing and machine vision applications, using an image processing libraries called IMAQ (IMage AcQuisition) [10] and Vision Development Module [11]. LabView simplifies computer vision development by being able to display the image at any stage of processing with a single click of the mouse, and overlay information on images without affecting the image data.

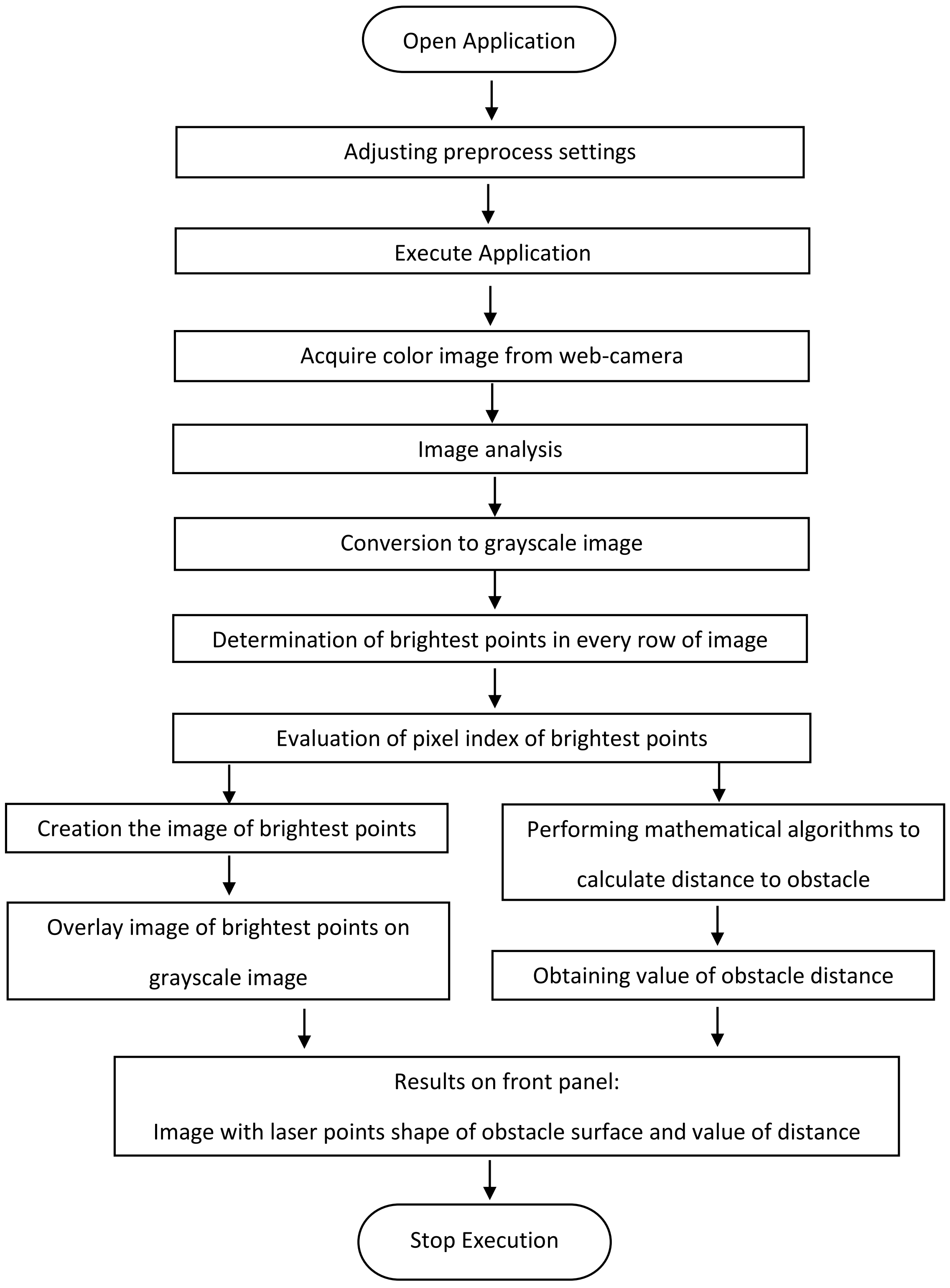
The IMAQ library contains simple image manipulation functions, such as resizing or extracting parts of images and several high-level vision tools, such as pattern recognition and shape analysis.

The Vision Development Module is designed to develop and deploy machine vision applications. It includes hundreds of functions to acquire images from a camera and to process images by enhancing them, checking for presence, locating features, identifying objects, and measuring parts.

Mathematical operations, which required in our algorithms, are performed using graphical blocks in Numeric palette from Programming Functions, which included in base package and does not require any additional structure library.

Numeric functions allow to create and perform arithmetic and complex mathematical operations on numbers and to convert numbers from one data type to another. Using the VIs and functions on the Elementary and Special Functions and VIs palette is possible to perform trigonometric and logarithmic functions.

6.2 Programming Application Structure



Open Application
Adjusting preprocess settings
Execute Application
Acquire color image from web-camera
Image analysis
Conversion to grayscale image
Determination of brightest points in every row of image
Evaluation of pixel index of brightest points
Creation the image of brightest points
Overlay image of brightest points on grayscale image
Performing mathematical algorithms to calculate distance to obstacle
Obtaining value of obstacle distance
Results on front panel:
Image with laser points shape of obstacle surface and value of distance
Stop Execution

6.3 User Interface of Application

Before running the application, it was designed some options for user interface to increment directly on the front panel.

Fig. 17 shows steps to implement and settings, which is possible to adjust in program on front panel before executing the code:

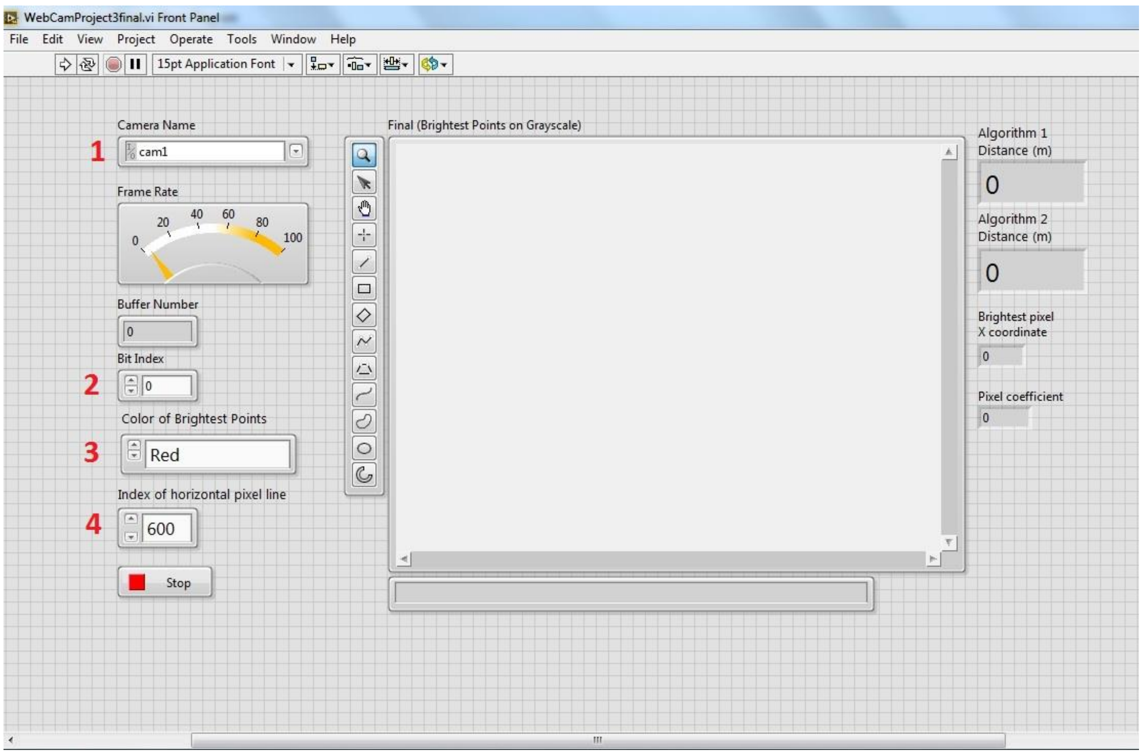
• Select the camera (option 1). In situation when personal computer or robot has several connected cameras it is required to choose the right one.

• Select the desired Bit Index (option 2). The user can select which of the eight bits to retain in grayscale from the original thirty-two bits of the image by changing the Bit Index control. This configuration can be easily modified to change the bit depth of any type of image.

• Choose the color of brightest points to overlay on web-camera grayscale image (option 3), evaluating position and shape of laser beam on obstacle. The color can be set to red, green or blue.

• Choose the number of horizontal line for data mathematical operations (option 4). This option can be useful for data calibration, or in situation when bright light appears in the center of image causing wrong condition for evaluation the brightest points in row as laser point. User can switch to other rows in the image to capture right data to perform in algorithm. In our assembling device the center of camera lens and laser eye of emitter are located on one line, so it is best option to use pixel line from the middle of camera image.

• Run the application (run button). Even while program is running - it is possible to change mentioned configurations.


WebCamProject3final.vi Front Panel

1 Camera Name: cam1

Buffer Number: 0

2 Bit Index: 0

3 Color of Brightest Points: Red

4 Index of horizontal pixel line: 600

Algorithm 1 Distance (m): 0

Algorithm 2 Distance (m): 0

Brightest pixel X coordinate: 0

Pixel coefficient: 0

Fig. 17 - Steps to complete before running the application on front panel

6.4 Execution Code Structure Type

As application works with continuously captured frames from the web-camera and not only with one image it is necessary to apply While Loop execution structure for main part of program. The While Loop repeats the code within its subdiagram until a specific condition occurs (Fig. 18).

In our case the While Loop will execute its code until the stop button on front panel (Fig. 17) will be pushed.

So code will execute with data of every frame from captured image in real time.

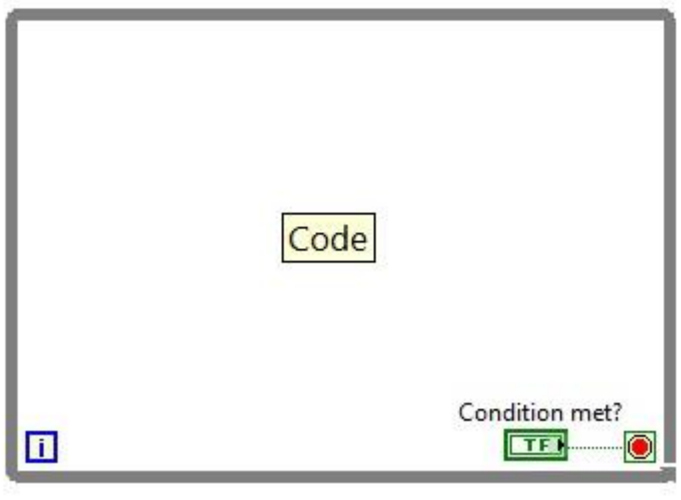
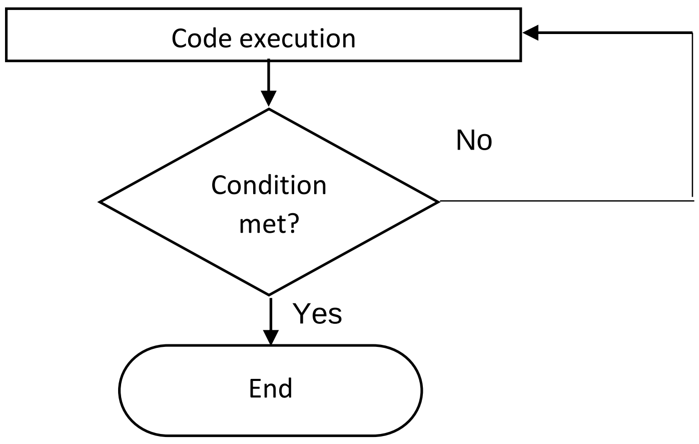


Fig. 18 - LabView While Loop execution structure

Flowchart of While Loop functionality



However, some part of code which related to connection and opening of the web-camera for acquiring captured image is located before While Loop structure. And last part of program linked to closure of the web-camera is disposed after While Loop.

6.5 Application Code. Acquiring the image

Some steps of software application is realized in form what is called a subVI - Labview's term for a function or subprogram. As it can be seen from Fig. 19 – each element in red circle has its own structure and functionality.

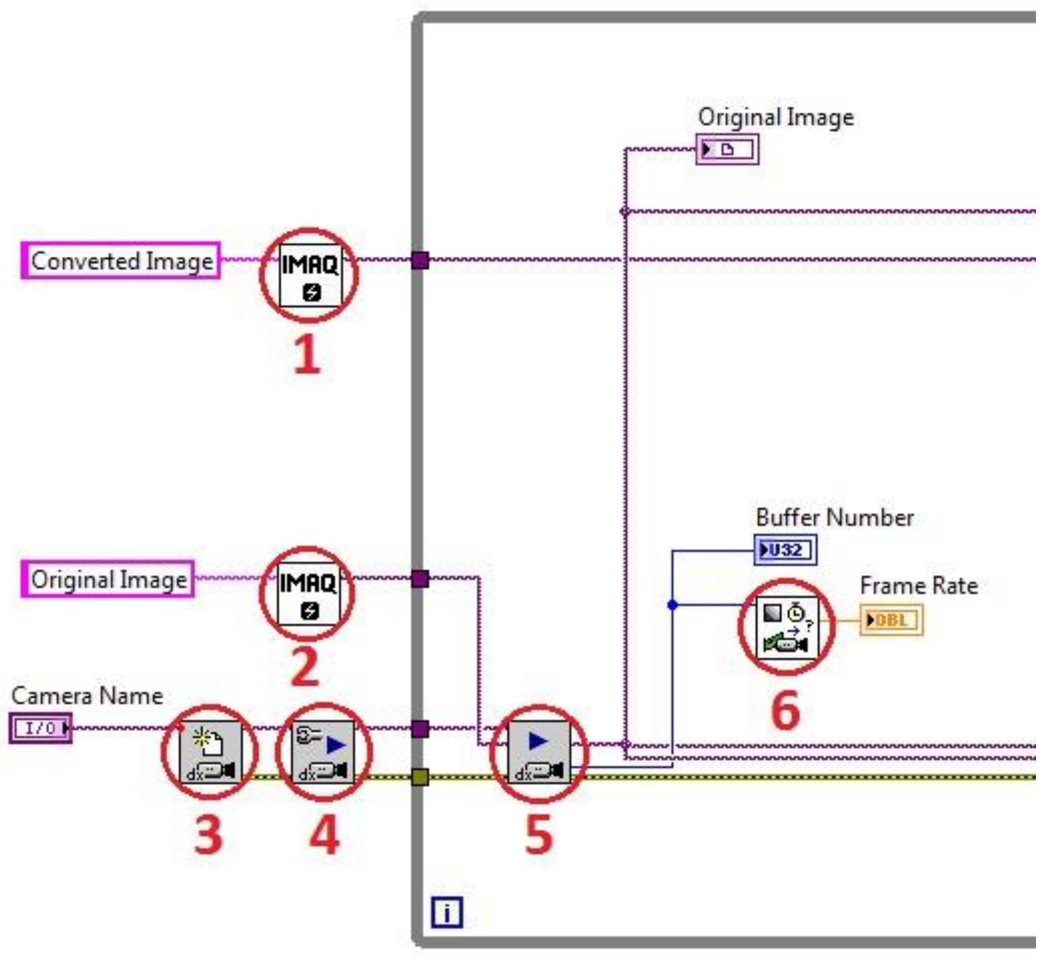


Fig. 19 - SubVI structure of first part of application on block diagram

Fig. 19 shows the block diagram of first phase of software, where the newly acquired camera image is prepared for image analysis.

The application starts by acquiring color image from web-camera in thirty-two bit RGB (Red, Green, and Blue) format.

"Camera Name" control element creates link to chosen web-camera and corresponds to front panel option 1 on from Fig. 17. Configuration before capturing of image by web-camera is performed using subVI 3 and subVI 4 (Fig. 19).

SubVI 3 – "Open Camera VI". Opens the web-camera, queries the camera for its capabilities, loads a camera configuration file, and creates a unique reference to the camera.

SubVI 4 – "Configure Grab VI". Configures and starts a grab acquisition. A grab performs an acquisition that loops continually on a ring of buffers. Applied for high-speed image acquisition. Also used to copy an image out of the buffer.

To perform further image analysis and processing operations with images and its data it is applied two "IMAQ Create VI" (subVI 1 and subVI 2 on Fig. 19), which creates a temporary memory location for an image. SubVI 1 is used for converted image and subVI 2 for original captured image.

SubVIs 1,2,3,4 are located before the While Loop structure because it relates to necessary starting conditions of the application – connection, opening and configure of the web-camera to capture the image.

After these elements introduced While Loop structure for further code of program.

The process of capturing original image from the web-camera is starts by subVI 5 “Grab VI” (Fig. 19). It acquires the most current frame in real time.

The output of this VI is “Original Image” – screen on the front panel with grabbed image from the web-camera.

There are also some additional tools are implemented as outputs of subVI 5. “Buffer number” indicator which shows number of current buffer on front panel. SubVI 6 (Fig. 19) is used to calculate frames per second. “Frame rate” is indicator of result of this subVI on front panel in knob form.

As a result of first part of code from Fig. 19 it is possible to obtain acquired thirty-two bit RGB original image from the web-camera and indicators on front panel of application (Fig. 20). Acquired camera image and its data is used for further image processing operations.

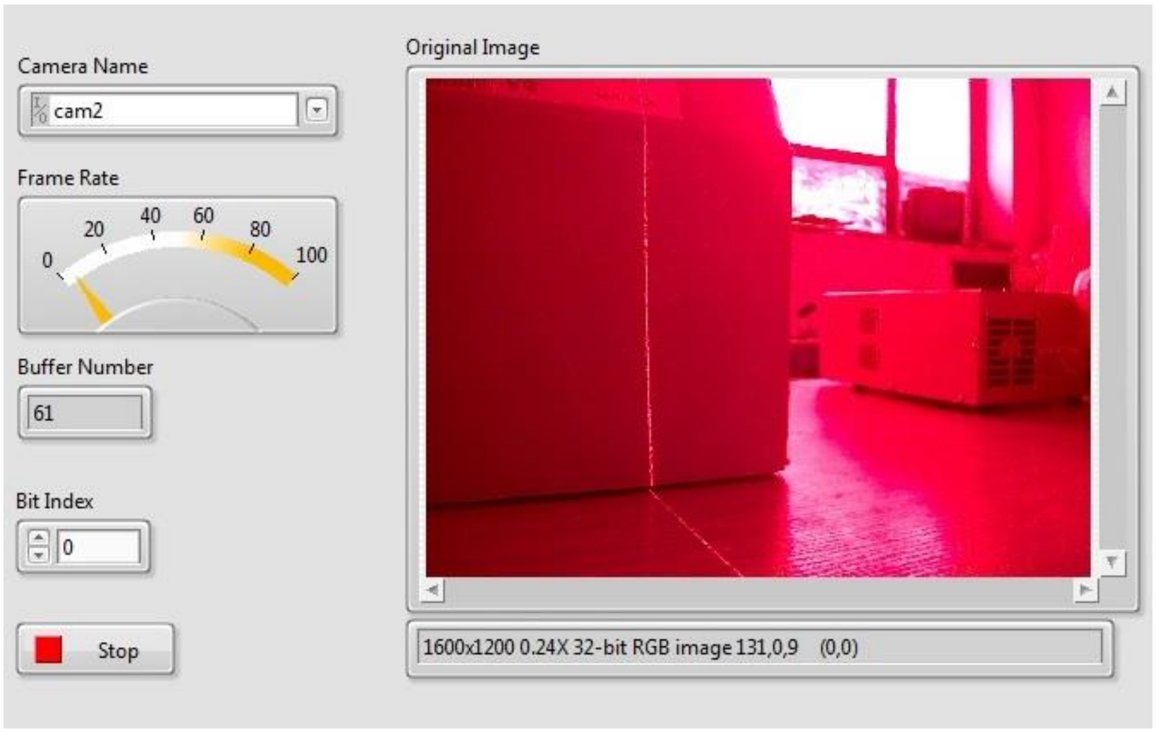


Fig. 20 - Results of first part of the code on front panel

6.6 Application Code. Conversion to grayscale

Next step of the application processes the original image to grayscale by converting the image to an array and then performing array operations to retain eight of the original thirty-two bits of the image.

According to image processing algorithm the main goal of this part is to provide operations with data of acquired color image and obtain converted grayscale image which will based on new data array.

As result of previous part of code was color image it is necessary firstly to extract the data from image. It is realized by conversion from image to array of data. To obtain new image in grayscale format it is required to implement data array to image conversion after completion of data manipulations.

Fig. 21 shows the part of LabView program where the original image acquired from the web-camera in first part of program is converted to grayscale image.

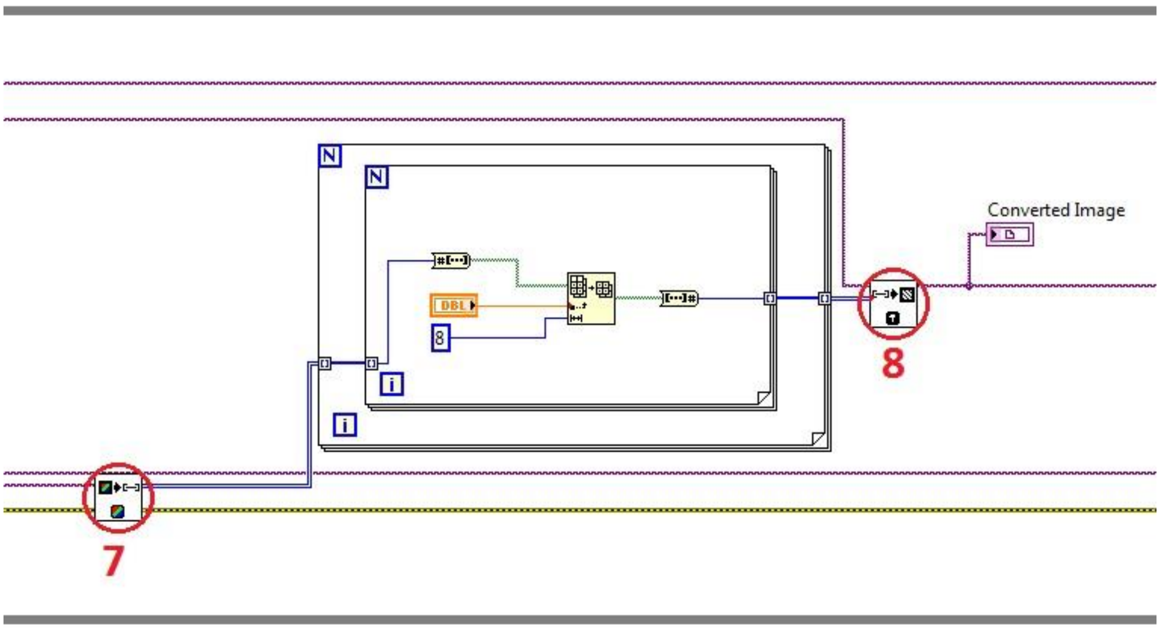


Fig. 21 - Conversion to grayscale block diagram

Firstly, is used block subVI 7 (Fig. 21) – “Color Image To Array VI”. This VI extracts the pixels from a color image or from part of a color image into a 2-dimensional (2D) array. In result it returns the values as a 2D array of either unsigned 32-bit integers or clusters of four unsigned 16-bit integers, depending on the bit depth of image. The input of block is a reference to the source image (“Original Image” on Fig. 19). The output is a 2D array of unsigned 32-bit integers of the pixel RGB values.

After conversion of the color image in to array of data it is necessary to perform data operations with number values. As 2D array format obtained it is required to convert it in first step in to 1D array and as second step – extract single numbers from 1-dimensional (1D) array. After data operations execution it is needed to have again 2D array format for further conversion. So it is necessary to form 1D array with new set of single number values obtained after data operations and compose new 2D array which will contain pixel values and its indexes.

The solution of data operations and array extractions (2D array to 1D array, 1D array to single number values) and forming (single number values to 1D array, 1D array to 2D array) is an applying of For Loop execution structures.

For Loop executes a subdiagram a set number of times. Fig. 22 shows a For Loop LabVIEW structure. In our case For Loop is execute for all data extraction values and array forming.

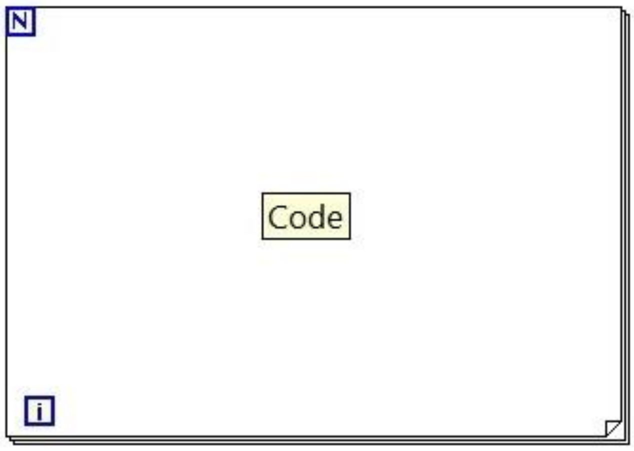
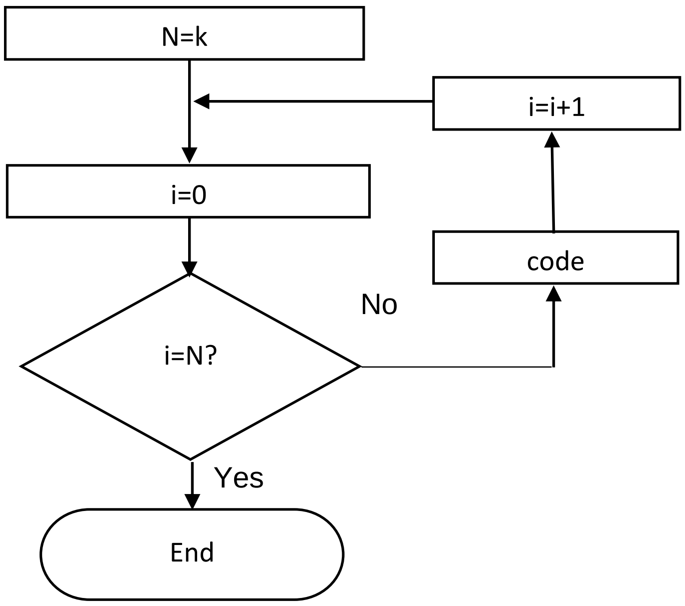


Fig. 22 - LabView For Loop structure

The count terminal (N) is an input terminal whose value indicates how many times to repeat the code. The iteration terminal is an output terminal that contains the number of completed iterations. The iteration count (i) for the For Loop always starts at zero.

The For Loop differs from the While Loop in that the For Loop executes a set number of times. A While Loop stops executing only if the value at the conditional terminal exists.

Flowchart of For Loop functionality



In the application is applied two For Loops (Fig. 22) for each type of procedure and its opposite operation. First For Loop is for 2D array to 1D array and 1D array to 2D array conversions. Inside it located second For Loop which used for 1D array to single number values extraction and forming of 1D array with new single number values.

Inside second For Loop data operations with single number values are performed for further steps of grayscale conversion.

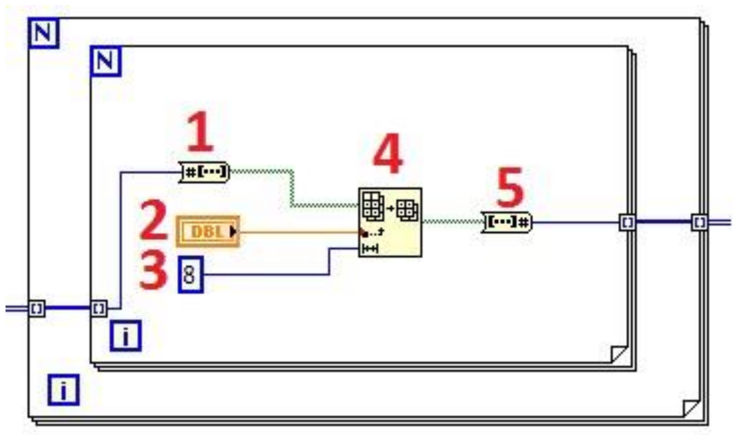


Fig. 23 - Data operations for grayscale conversion

The main element of code of data operations with single number values is Array Subset function (position 4 on Fig. 23). It returns a portion of array starting at index and containing length elements.

The inputs are: boolean array obtained after single number conversion (position 1), length (position 3) specifies the elements portion of array to return (according to principle of grayscale conversion it is required to retain eight of the original thirty-two bits of the image, so index is 8). As index of first elements to return it was implemented Bit Index control (position 2) which is related to front panel option (position 2 on Fig. 17). Last operation is conversion of Boolean array in to number (position 5).

Final step of this application part is array of data to image conversion. It is performed using SubVI 8 (Fig. 21) called "Array To Image VI". This VI creates an image from a 2D array. The inputs are: reference for source image ("Converted Image" on Fig. 19) and 2D array of unsigned 8-bit integers containing the pixel values that comprise the image. The first index of array corresponds to the vertical axis and the second index corresponds to the horizontal axis (ppx – pixel index in mathematical algorithms). The VI resizes converted image to be the same size as image pixel values in array. The output is reference to the destination image – "Converted Image" (Fig. 21).

The result of second part of application is converted grayscale image. The data of this image is used for further implementation of mathematical algorithms of distance determination.

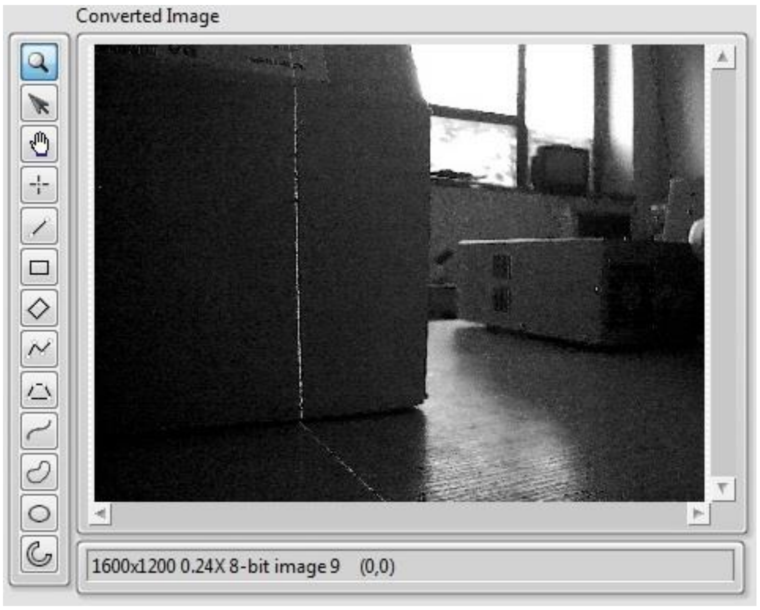


Fig. 24 – Result of second part of code

6.7 Application Code. Brightest Pixel Determination

Third part of application is determination of laser line footprint on the converted grayscale image. In program code is implemented image processing algorithm “brightest pixel per row”. The goal of this approach is extraction of points with brightest pixel value in every pixel row of grayscale image.

Fig. 25 shows the application code of evaluation of brightest points and determination of its data for creating the image of extracted laser points and for mathematical algorithm of laser distance calculation.

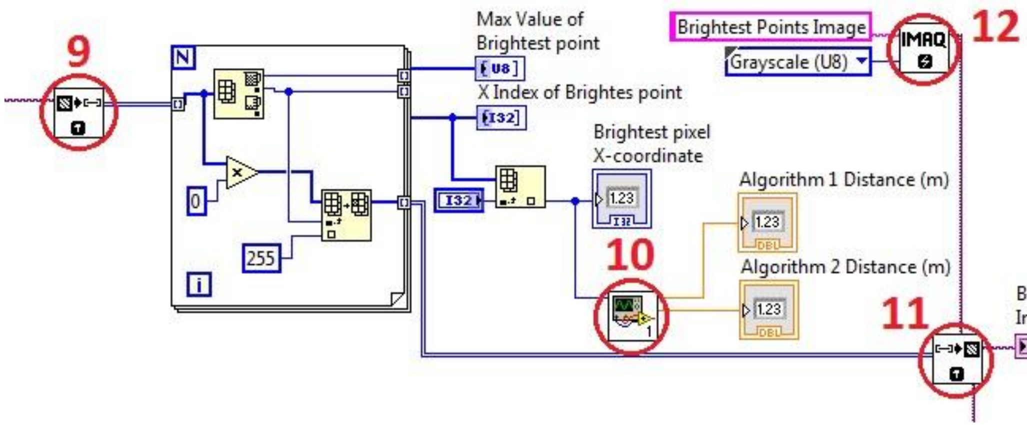


Fig. 25 - Brightest points algorithm block diagram

The result of previous part of application is grayscale converted image. Using data of this image it is possible to evaluate pixel values and indexes of horizontal and vertical position of brightest points.

For this purposes SubVI 9 (Fig. 25) “Image To Array VI” is used. This VI extracts the pixels from an image, or part of an image, into a 2D array. This array is encoded in 8 bits as determined by the type of input image – 8-bits grayscale image. The input of this VI is reference to the source image (“Converted Image” on Fig. 21). The output - image pixels, returns the extracted pixel values into a 2D array. The first index corresponds to the vertical axis and the second index corresponds to the horizontal axis. Using data of pixel values and its horizontal index it is possible to brightest points in every row by seeking pixel with highest pixel value (biggest value – 255) in every pixel row (1200 rows due the web-camera resolution). So in result it will be data array of 1200 numbers of the brightest points with pixel values and its vertical and horizontal indexes.

For evaluation of pixel values of brightest points and its x-axis index (horizontal axis) in every pixel row it is applied For Loop execution structure (Fig. 25).

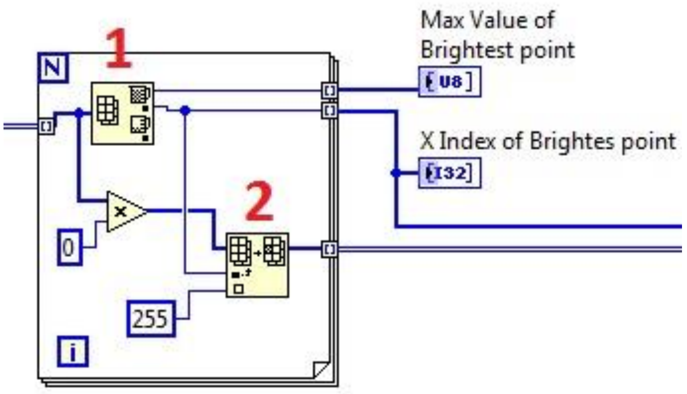
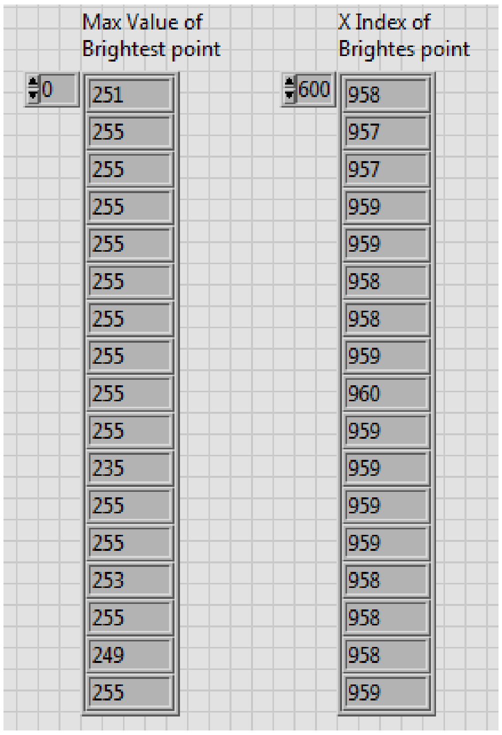


Fig. 26 - Evaluation of pixel value and its x-axis index

Main element inside For Loop structure is Array Max & Min function (position 1 on Fig. 26). It returns the maximum and minimum values found in array, along with the indexes for each value. As it has 2-dimensional input and goal of the function is determining only biggest value in row – as output it has two 1-dimensional arrays (Fig. 27) with biggest maximum pixel value and its horizontal index, which determines position of brightest pixel in x-axis (ppx – pixel index in mathematical algorithms).



0	Max Value of Brightest point	600	X Index of Brightes point
	251		958
	255		957
	255		957
	255		959
	255		959
	255		958
	255		958
	255		959
	255		960
	255		959
	235		959
	255		959
	255		959
	253		958
	255		958
	249		958
	255		959

Fig. 27 - Data arrays with pixel values an x-indexes on front panel

On Fig. 27 it can be seen data evaluation of “brightest points per row” algorithm. It is mathematical representation of brightest points which represents position of laser points on the image which has 3 parameters: pixel value (0...255 – RGB values), x-index – position of brightest pixel on horizontal axis (0…1599) and y-index – index of vertical axis or row (0…1199). Maximum values of x and y indexes are determined by camera resolution.

Values of these data arrays can be used in further mathematical algorithms for laser distance determination and to create image with extracted laser line points as part of laser line detection image processing algorithm.

To distinguish the laser points from other possible sources with same brightness – light or white color, all other pixel values except brightest points in row (which all set to pixel value – 255) are set to 0. In this condition is possible to obtain in fact binary image. After this operation, array will contain data with only two possible values – 255 (brightest points represents laser) or 0 (black color which represents other environment).

This data operations done by Replace Array Subset function (position 2 on Fig. 26). It replaces an element or subarray in an array at the point specified in index. The output of this function is array with the replaced elements.

Based on obtained 2D array and using SubVI 11 “Array to Image VI” (Fig. 25) is possible to create image of extracted brightest points. The inputs of SubVI 11 are 2D array and reference to the source image which temporary memory location created by SubVI 12 “IMAQ Create VI” (Fig. 25). The output of VI is reference to the destination image – “Brightest Points Image”.

In result image with extracted laser line points is obtained. Evaluated brightest points represents footprint of laser in real environment and has same shape like laser spreads on object surface.

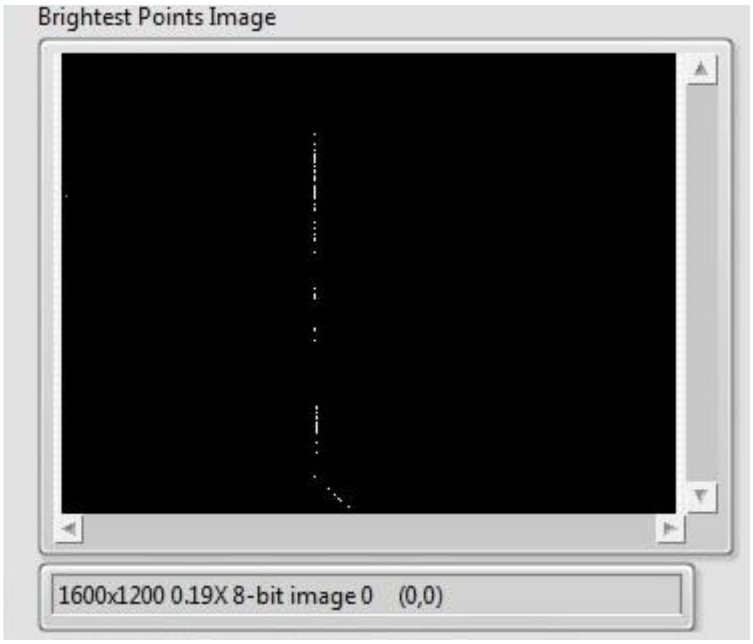


Fig. 28 - Extracted laser line after third part of code

Value of x-index of brightest points is necessary input parameter of SubVI 10 (Fig. 25) “Mathematical Calculations VI”. As goal of this work is distance determination to obstacle – it is required to apply the values of pixel index of brightest point from one pixel row. In our device conditions the center of camera lens and laser eye of emitter are located on one line, so it is best option to use pixel line from the middle of camera image. So by default input setting it is applied row with y-index equals 600. User can change this parameter from front panel of application using control of horizontal pixel line (option 4 from Fig. 17). It is realized by Index Return function. This element returns the element or subarray of n-dimension array at controlled index.

For mapping of environment purposes it is possible to use data of brightest points from every pixel row of the image. It allows to develop algorithm of distance determination to every object on captured image where laser beam falls. For example – table, room walls and doors.

In our specified case for movement of mobile robot we are interested in detection and laser distance determination to the closest obstacle in front of robot. So main output of “brightest points per row” algorithm will be the values of x-index of central pixel row of image. X-index, called in mathematical algorithms as pixel index (ppx=0…1599) – certain number of pixels from left border of image (0 – left border of the image, 1599 – right border).

6.8 Application Code. Mathematical Algorithms

After laser line extraction and determined values of pixel index of brightest point it is necessary to perform mathematical calculations to evaluate the laser distance to detected obstacle.

SubVI 10 (Fig. 25) “Mathematical Calculations VI” is code implementation of mathematical algorithms 1 & 2. All data operations in this SubVI are based on equations (11) and (13) of mentioned algorithms.

Algorithm 1:

$$d_1 = \frac{x}{(tan(\alpha) + tan(\beta)) - 2 * p * \ tan(\alpha)} \tag{11}$$

$$p = \frac{\Delta l}{l} = \frac{l - ppx}{l} \tag{12}$$

Algorithm 2:

$$d_2 = \frac{x}{tan(ppx * rpc + ro)} \tag{13}$$

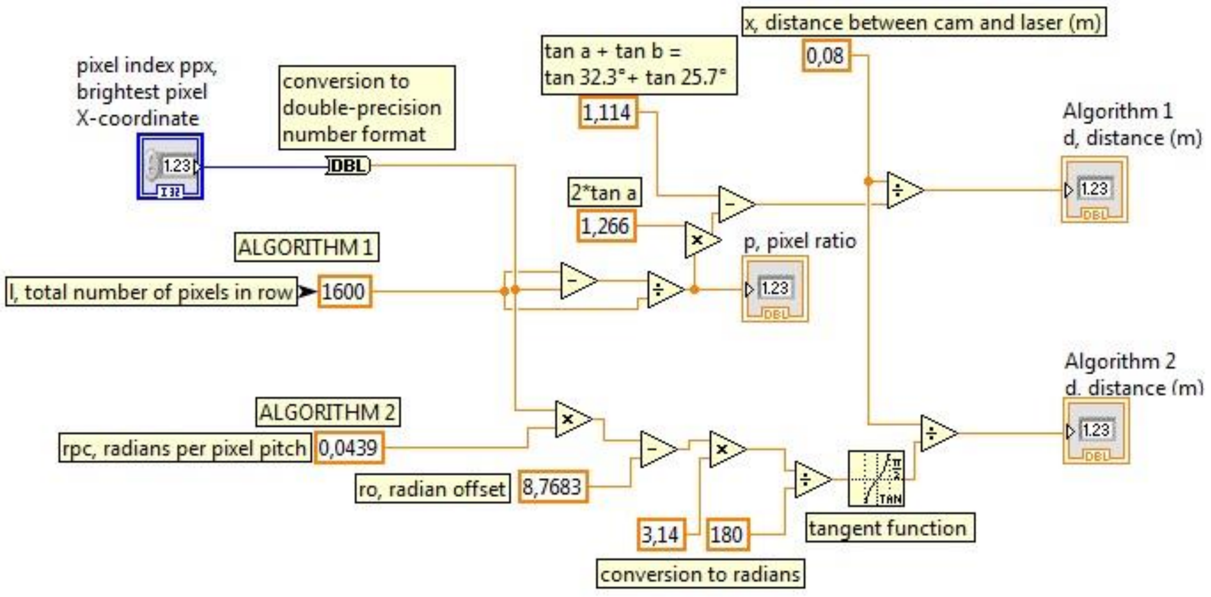

pixel index ppx, brightest pixel X-coordinate
1.23
I32
conversion to double-precision number format
DBL
tan a + tan b = tan 32.3°+ tan 25.7°
1,114
x, distance between cam and laser (m)
0,08
Algorithm 1 d, distance (m)
1.23
DBL
2*tan a
1,266
p, pixel ratio
1.23
ALGORITHM 1
l, total number of pixels in row
1600
Algorithm 2 d. distance (m)
1.23
ALGORITHM 2
rpc, radians per pixel pitch
0,0439
ro, radian offset
8,7683
3,14
180
conversion to radians
tangent function
TAN

Fig. 29 - Code of Mathematical Calculation SubVI

Fig. 29 shows the code of mathematical operations for laser distance to obstacle evaluation.

Main input parameter to the mathematical model is value of pixel index (ppx=0…1599), which evaluates x-coordinate position of brightest points. This value is result of image processing algorithm and not constant, as either mobile robot or obstacle moves. Pixel index value is obtained in long signed integer format after image processing algorithm. So it is required to convert it to double-precision number format for further mathematical calculations.

All other input parameters are constant and caused by geometrical conditions of assembling device and camera lens capability.

Along with pixel index the common input parameter for Algorithm 1 and Algorithm 2 is:

- x = 0.08 (m) – distance between center of web-camera lens and laser eye emitter

Input parameters for Algorithm 1 according to the equation (11) are:

- l = 1600 – entire number of pixels in row
- α = 32.3° – horizontal opening angle
- β = 25.7° – tilt angle

Parts of the equation (11) which contains only constant input parameters are applied as known variables:

$$tan(\alpha) + tan(\beta) = tan(32.3°) + tan(25.7°) = 1.114 \qquad (14)$$

$$2 * tan(\alpha) = 2 * tan(32.3°) = 1.266 \qquad (15)$$

For Algorithm 1 is introduced numeric indicator of value of pixel ratio – p. It allows to obtain the result of pixel ratio from equation (3) on front panel of SubVI and validate the principle of mathematical model 1: as smaller value of p, as larger distance to obstacle.

Input parameters for Algorithm 2 according to the equation (13) are:

- rpc = 0.0439 - radians per pixel pitch
- ro = -8.7683 - radian offset

As it is applied LabView tangent function which computes the tangent of variable, where variable is in radians – it is necessary to add conversion to radians. To convert degrees to radians is needed to calculate the number of half circles in the answer by dividing by 180º. But each half circle equals π radians, so the number of half circles multiplied by π.

$$radians = degrees * \frac{\pi}{180} \qquad (16)$$

All mathematical operations are performed using elements from numeric LabView palette.

Results of implemented mathematical algorithms 1 & 2 are obtained in double-precision number format and visible on front panel of main application in form of numeric indicators.

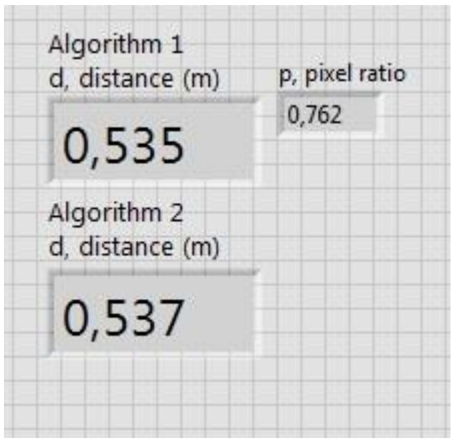

Algorithm 1
d, distance (m)
0,535
p, pixel ratio
0,762
Algorithm 2
d, distance (m)
0,537

Fig. 30 - Front panel of Mathematical Calculations SubVI with results

6.9 Application Code. Overlaying of Images

Next part of application code is implementation of image post processing algorithm. It was decided to have in result on front panel the image with extracted laser line and environment at one frame. The solution is to apply overlay technique in image.

As first step is performed a threshold on the brightest points image obtained after image processing part. Second step is to color areas of the threshold brightest point binary image. Final step - to overlay it on the grayscale image of acquired image.

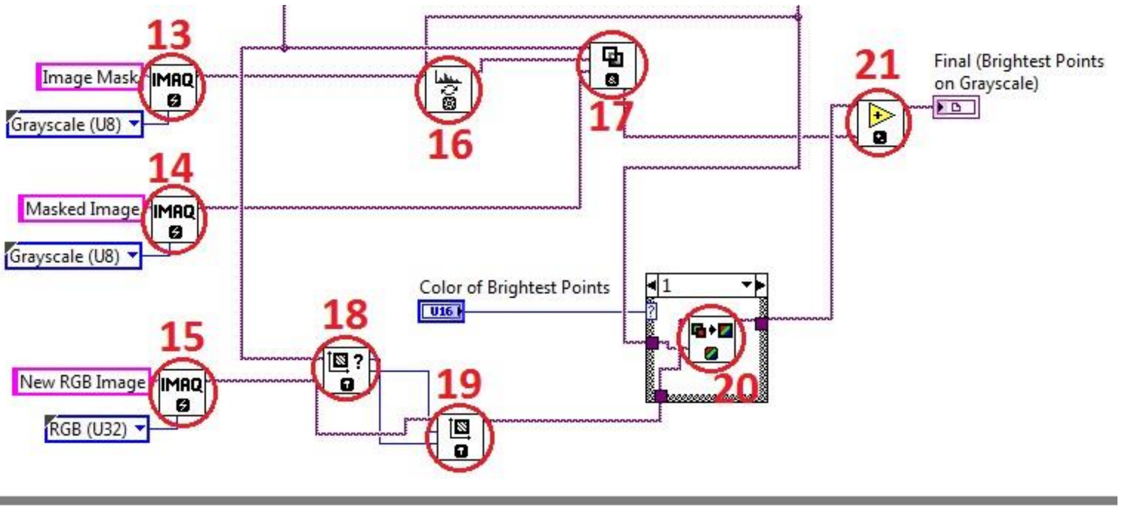


Fig. 31 - Block diagram of image post processing overlay technique

Most of elements used in this part of application is taken from Vision Development Module library.

To perform image post processing operations with images it is applied three “IMAQ Create VI” (subVI 13, subVI14 and subVI 15 on Fig. 31), which creates a temporary memory location for an image at every step of overlaying process.

SubVI 16 “IMAQ Inverse VI” inverts the pixel intensities of an image to compute the negative of an image. First input of this subVI is a reference to the source image – Brightest Points image (Fig. 25). Second input is reference to Image Mask - an 8-bit image that specifies the region of the small image that will be copied. Only pixels in the source image that correspond to a non-zero pixel in the mask image are copied. All other pixels keep their original values. The output is a reference to the destination image for next step.

SubVI 17 “IMAQ Mask VI” recopies the source image into the destination image. If a pixel value is 0 in the Image Mask, the corresponding pixel in destination image is set to 0. SubVI 17 has three inputs. First input is reference to the source image – Converted Grayscale Image (Fig. 21). Second input is Image Mask which is the output of SubVI 16. Third input is reference to Masked Image. The output of SubVI 17 is reference to base image for overlaying.

SubVI 18 “Get Image Size VI” gives information regarding the size (resolution) of the image. It specifies the horizontal (X) and vertical (Y) resolution of source image – Converted Grayscale image. The data of resolution passes to SubVI 19 “Set Image Size VI” which modifies the resolution of a new created RGB image. These elements allow to have final image with same size and resolution like acquired image from the web-camera.

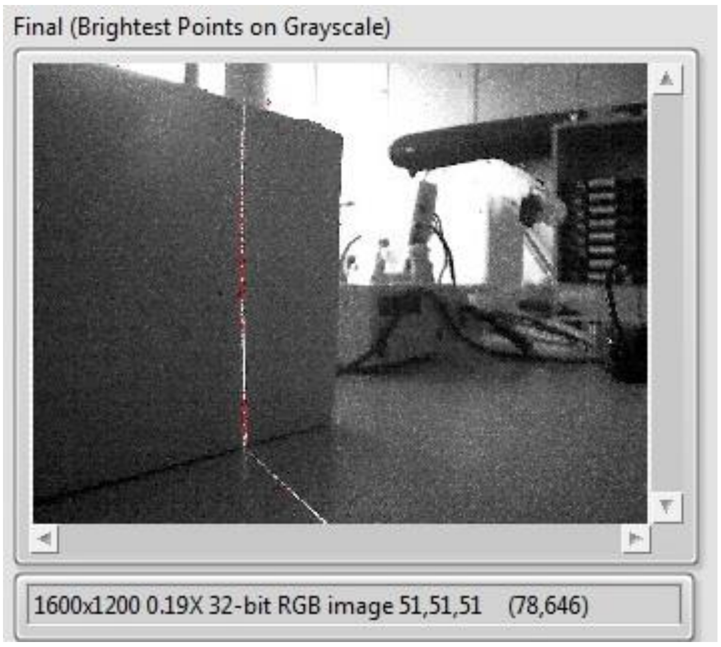
SubVI 20 “Replace Color Plane VI” replaces one or more image planes from a color RGB image. Only the planes connected at the input are replaced. The image is resized to the dimensions of the planes passed on input.

SubVI 20 is located in Case LabView execution structure. The Case structure has three subdiagrams, or cases. Only one subdiagram is visible at a time, and the structure executes only one case at a time. An input value determines which case executes.

This structure creates option to the user to set the color of area of image which will overlay on the base image on final screen. The color of the threshold area can be set to red, green or blue from the front panel (option 3 on Fig. 17). In default settings it is set to red color. But in some conditions, for example if it is necessary to overlay brightest points image on the red filter image, other colors to distinguish threshold area will be useful.

SubVI 21 “IMAQ Add VI” (Fig. 31) adds two images or an image and a constant. In this code this SubVI adds two source images – Masked Image of Grayscale Converted Image and New RGB Image of Brightest Points Image. The output of SubVI is the reference to the destination image – Final (Brightest Points on Grayscale) that receives the processing results of the VI.

In result final image consists of grayscale image and red threshold image with laser line footprint (Fig. 32).



1600x1200 0.19X 32-bit RGB image 51,51,51   (78,646)

Fig. 32 - Grayscale image with overlay image of brightest points

6.10 Application Code. End of Execution

The last part of program code is related to the end of application execution and closure of the web-camera.

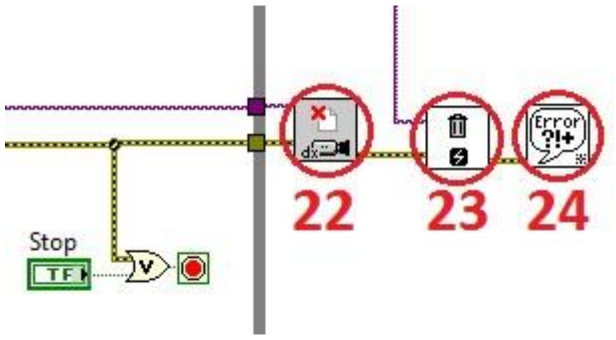


Fig. 33 - Stop execution part of code

According to the conditions of While Loop the application will stop execute its code when the stop button on front panel (Fig. 17) will be pushed.

There are also elements of code which located after While Loop to complete necessary conditions of camera and application memory.

SubVI 22 “Close Camera VI” (Fig. 33) stops an acquisition in progress, releases resources associated with an acquisition, and closes the specified camera session. The input is a unique reference to the camera opened with the SubVI 3 “Open Camera VI” (Fig. 19).

SubVI 23 “IMAQ Dispose VI” destroys an image and frees the space it occupied in memory. This VI is required for each image created in an application to free the memory allocated to the “IMAQ Create VI” (subVI 1 and subVI 2 on Fig. 19). When a LabVIEW application is aborted, allocated images remain in memory. The input specifies the reference to the image to destroy.

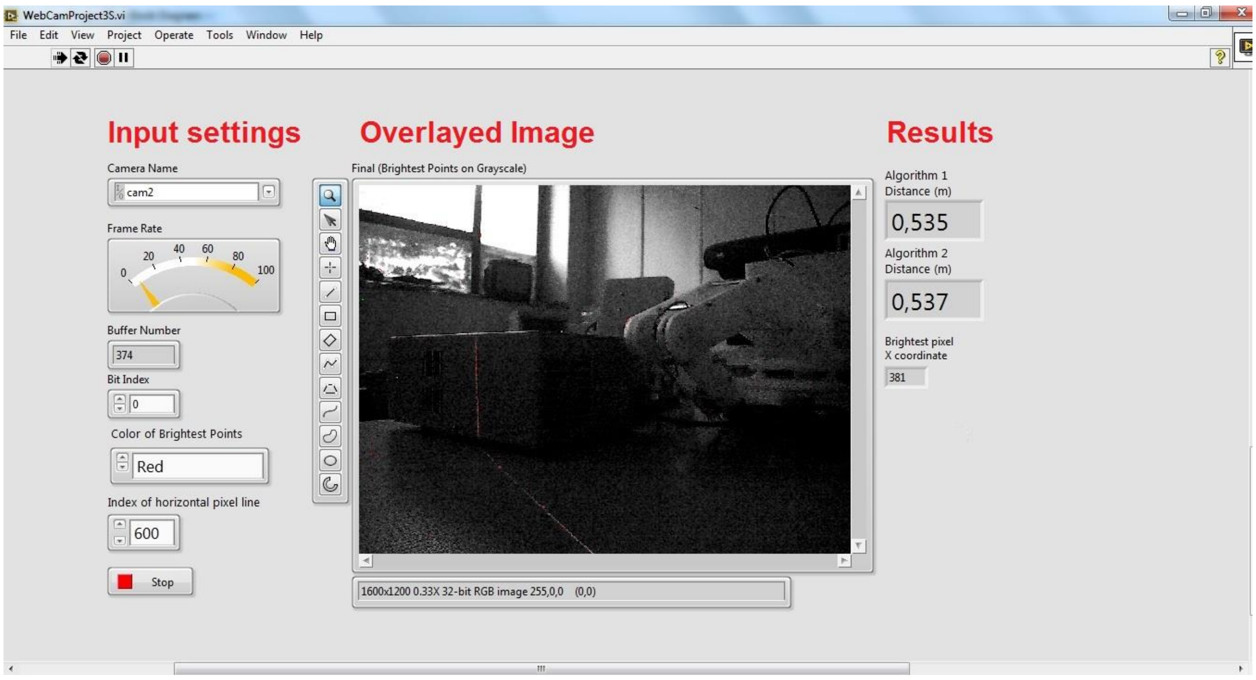
Last element – SubVI 24 “General Error Handler VI” Indicates whether an error occurred. If an error occurred, this VI returns a description of the error and optionally displays a dialog box.

In result after stop of execution of application the web-camera will be closed and image memory of program will be free.

6.11 Running Application

As a result of completed code of application is front panel of running application. It contains acquired by the web-camera image in grayscale format with overlay red color image of brightest points represented laser points shape of obstacle surface and values of laser distance to obstacle determined by two mathematical algorithms. All image processing and data evaluation are performed in real-time.

While program is running it is possible to modify some settings on the front panel – bit index, color of brightest points on the final image and change pixel row for distance determination algorithms.



Input settings

Camera Name: cam2

Frame Rate

Buffer Number: 374

Bit Index: 0

Color of Brightest Points: Red

Index of horizontal pixel line: 600

Stop

Overlayed Image

Final (Brightest Points on Grayscale)

1600x1200 0.33X 32-bit RGB image 255,0,0 (0,0)

Results

Algorithm 1 Distance (m): 0,535

Algorithm 2 Distance (m): 0,537

Brightest pixel X coordinate: 381

Fig. 34 - Front panel with result of running application

7. Experimental Results

After the software solution was designed it was necessary to perform experiments with device and running application to validate proposed image processing techniques and distance determination mathematical algorithms. All tests were performed in enclosed room in daylight time with suitable distance range for mobile robot navigation – up to 0.5 meter. As an obstacle was cardboard box on the table

Real distance (m)	Algorithm 1 (m)	Algorithm 2 (m)
0.05	0 (below limit)	0.048
0.075	0.079	0.075
0.1	0.105	0.101
0.15	0.154	0.147
0.2	0.199	0.198
0.25	0.248	0.251
0.3	0.301	0.308
0.4	0.402	0.404
0.5	0.504	0.506

Table 3 - Results of experiments of distance to obstacle determination

Based on Table 3 was built graph for graphical representation of obtained results.

On Fig. 35 it can be seen the result of experiments of distance determination performed by software application based on two mathematical algorithms.

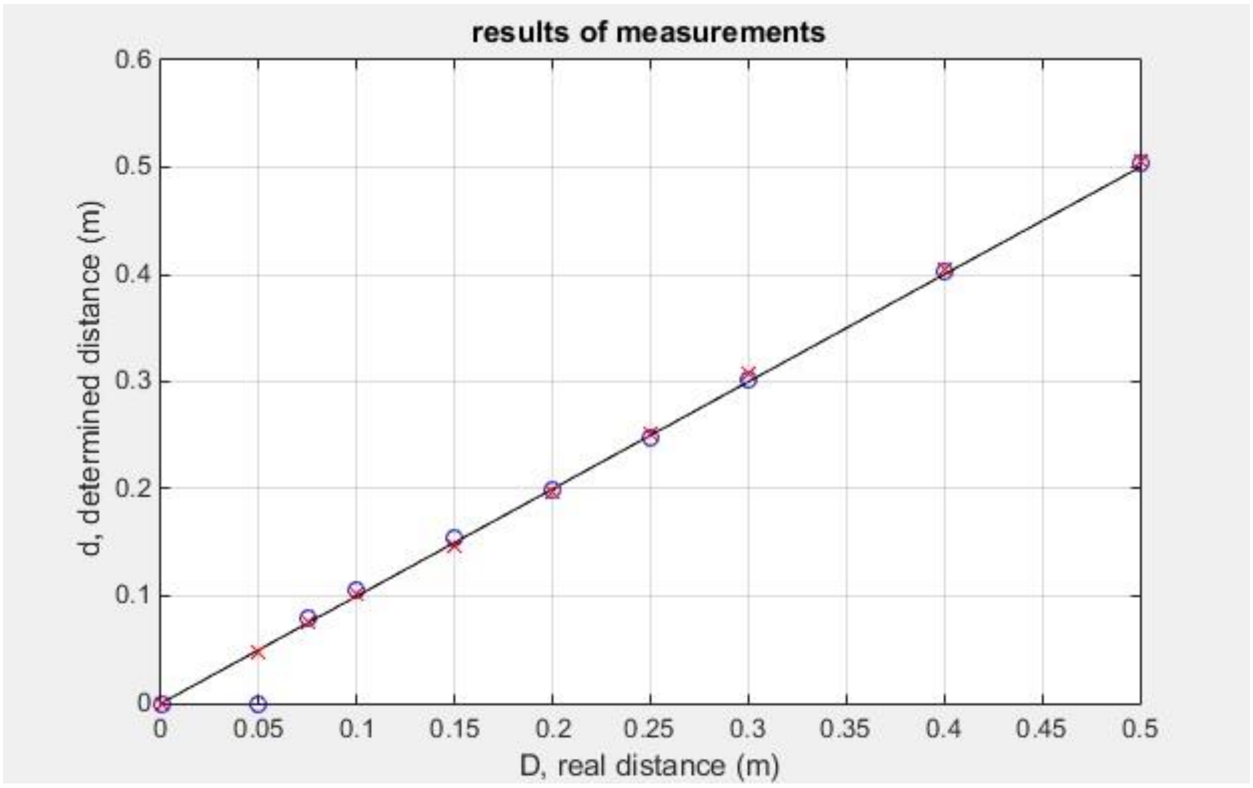


Fig. 35 - Results of distance measurements performed by software application

(–) – ideal result (determined distance equals real distance). Used for comparison.

(o) – result of distance determination by Mathematical Algorithm 1

(x) – result of distance determination by Mathematical Algorithm 2

Results of experiments shows precise values of distance to detected obstacle and confirms applied mathematical models. Highest error in comparison with ideal performance – nearly 5% obtained with small distances up to 0.1 meter.

As optical triangulation models are based on mechanical and optical components there are can be some sources of possible errors in measurements, which are outside of calculation system such as: bright light, weak laser signal, reflectivity of surface, circle shape of object, not precise assembling of device.

Image processing algorithm of laser line detection – “brightest pixel per row” also shows good performance. In most cases brightest points represented footprint of laser beam which was detected. The only hurdle was met during tests is the condition when bright light appears in left side of image before laser beam. As used algorithm extracts only one brightest point from each row it could led to extraction of points representing bright light points with same pixel index instead of laser points.

Result image obtained on the final screen (Fig. 34) consists of extracted layer with brightest points over acquired image in grayscale format for better visibility of red laser beam points. Laser footprint is represented in image by points which always moves in time and blinking on the screen. So there is impossible to catch the image with full points of laser footprint.

During tests were validated limitations to designed device:

• localization of laser beam in the image creates minimum and maximum distances for possible applying of mathematical algorithms. This distances proportional to distance between camera and laser at assembling device (value x on Fig. 9) and influenced on distance determination part of software. In our case it was limitation of minimal distance for Mathematical Algorithm 1 – 0.07 m.

• the accuracy of calculation will be lower with transparent, glass or mirror surfaces. In this cases, depending on reflectivity, the device may evaluate wrong distance or being unable to detect the laser.

• significant decrease of system performance appears in bright light environment, especially with distances to object more than 0.5 meter. This conditions are harder to image processing algorithm to distinguish laser spots from bright light. That’s why this device will show worse result outside in sunny weather.

• degree of turn of device is restricted because of USB-cable connection from the web-camera to computer. To avoid the winding of cable over the device it is better do not exceed 180° degree turn of device.

Designed system will show the best performance in indoor application with objects with smooth and opaque surfaces. In this case the accuracy of obtained result will be the highest and will satisfy conditions for robot navigation.

8. Conclusion

The result of work is measurement system which consists of hardware device and software application. Using this structure with mobile robot it is possible to detect obstacle and determine the distance to it with high accuracy.

Mounted device couples the required components – web-camera and laser emitter in one element by common detail, which can be driven by stepper motor. The device comprises low-cost parts and does not require additional equipment.

The obstacle identified using image processing algorithm of evaluation of brightest points from image captured by the web-camera and distance calculation performed by implemented mathematical algorithms in real time. Running application as a result provides the extracted image with laser footprint on the surface of obstacle overlaying on original acquired image in grayscale format and values of distance.

Measured results confirmed the expectations of applied triangulation mathematical models and shows the capability of precise distance determination in required range for mobile robot. Experimental results verified both proposed mathematical algorithms for distance calculation.

The success of the software solution in LabView can be attributed to reliably detecting laser line, using the detected points data array for all calculations fast enough for possible movement of mobile robot.

The designed device is well-suitable for indoor usage and can be easily connected to the front side of robot.

9. Future Work

The device and software application described in this work shows precise results without big errors in determined data. However, the possible improvement and development of the system are still not over.

Some steps which possible to perform to obtain better accuracy in mathematical algorithms:

• change the location of the web-camera in assembling device in parallel line relating to the laser emitter (make tilt angle $\beta = 0$ on Fig. 9 - Mathematical model for Algorithm 1). It creates condition when as close obstacle to the web-camera as it close to the center of image frame in both x and y coordinates. It gives more opportunities for measurement calibration.

• use YC_BC_R color space model instead of RGB model. It will enable to perform more accurate localization laser beam in unknown environment. In RGB chrominance and luminance components are mixed that is why RGB model is not best option for color analysis and color based segmentation algorithm.

There is an opportunity to modify principle of algorithm from distance to obstacle determination to mapping the environmental. As our system has vertical line spreading laser beam, designed software application can perform measurements at any point of vertical axis. So using array of data of brightest points from every row of the image which represents each point of detected laser beam it is possible to construct a vector map of the ambient environment and determine distances to any object in the captured image.

For automation of process of the rotation of device it is necessary to connect stepper motor to controller and implement the software which will coordinate movement of device with measurement operations.

The last part what can be done to complete whole process of obstacle detection and avoidance is development of software solution which will automatically change course of movement of the robot or will stop it immediately after obstacle detection in required distance range.

List of References

[1] D.C. Brown. Close-range camera calibration. Photogrammetric Engineering, 37(8): pp. 855–866, 1971.

[2] Y. Oike, M. Ikeda, and K. Asada. A 120x110 position sensor with the capability of sensitive and selective light detection in wide dynamic range for robust active range finding. Solid-State Circuits, IEEE Journal of, 39(1): pp. 246–251, 2004.

[3] D. Scaramuzza, A. Harati, and R. Siegwart. Extrinsic self calibration of a camera and a 3d laser range finder from natural scenes. In Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on, pp. 4164–4169, 2007.

[4] T. Danko. Webcam based DIY laser rangefinder. https://sites.google.com/site/todddanko/home/webcam_laser_ranger, accessed on 7.3.2016.

[5] C.E. Portugal-Zambrano, J.P. Mena-Chalco. Robust Range Finder Through a Laser Pointer and a Webcam. Electronic Notes in Theoretical Computer Science, 281, pp. 143-157, 2011.

[6] P. Chmelar, M. Dobrovolny, "The Optical Measuring Device for the Autonomous Exploration and Mapping of unknown Environments" Perner's Contacts, vol. VII, no. 4, pp. 41-50, 2012.

[7] P. Chmelar, M. Dobrovolny, The Laser Line Detection for Autonomous Mapping Based On Color Segmentation. International Journal of Mathematical, Computational, Physical, Electrical and Computer Engineering Vol:7, No:12, pp. 1654-1658, 2013.

[8] S. Barreto, R. Sant'Anna, M. Feitosa. A method for image processing and distance measuring based on laser distance triangulation. Electronics, Circuits, and Systems (ICECS), 2013 IEEE 20th International Conference on, pp. 695 – 698, 2013.

[9] S. Aji, T. Singh. Obstacle detection for navigation of robot using computer vision and laser rangefinder. International Journal of Electrical, Electronics and Computer Systems (IJEECS), vol. II, pp. 64-70, 2014.

[10] C.G. Relf. Image Acquisition and Processing with LabVIEW, ISBN 0-8493-1480-1, 268 p., 2004.

[11] T. Klinger. Image Processing with LabVIEW and IMAQ Vision, ISBN 0-13-047415-0, 368 p., 2003.