Scientific Visualization, 2024, volume 16, number 1, pages 38 - 51, DOI: 10.26583/sv.16.1.04

Building Depth Maps Using an Active-Pulse Television Measuring System in Real Time Domain

I.D. Musikhin[1], V.V. Kapustin[2], A.A. Tislenko[3], A. Movchan[4], S.A. Zabuga[5]

Tomsk State University of Control Systems and Radioelectronics

[1] ORCID: 0009-0001-7776-1698, musihin.i@tu.tusur.ru
[2] ORCID: 0000-0002-2293-0511, vk@tu.tusur.ru
[3] ORCID: 0009-0009-8082-6753, tislenko_1999-2012@mail.ru
[4] ORCID: 0000-0002-0020-6354, ltaak@tu.tusur.ru
[5] ORCID: 0009-0004-2402-2885, sergeizabuga@gmail.com

Abstract

The paper presents the results of software development for building depth maps based on video data from a television camera of an active-pulse television measuring system (AP TMS) in real time domain. The development of such software is required to conduct various scientific studies, as well as to improve the methods and techniques for building depth maps and remote measurement of the characteristics of objects of interest. The software was implemented using the Python programming language with additional libraries installed. According to the results of testing the implemented algorithm, it was found that the calculation speed using the graphics processing unit (GPU) is on average 3.5 times higher than the speed of the algorithm using only the central processing unit (CPU). It has been established that with the help of CUDA cores it is possible to build depth maps in real time domain at the maximum possible resolution of video frames of the system (1544x2064 pixels), while when using the central processor, real-time operation is possible only at a reduced resolution of video frames (772x1032 pixels).

Keywords: Depth maps, active-pulse television measurement system, image processing, CPU, CUDA cores, GPU, performance.

1. Introduction

In today's world, remote distance measuring is a very important and challenging task. This problem arises in geodesy, in radar, in video analysis, in the design of various infrastructure systems, unmanned vehicles, etc. There are many methods for distance measuring. Distance measurement can be carried out directly by the operator, using the simplest methods, or with the help of specialized devices, the operation principles of which are based on the optical laws of physics. Active-pulse television measuring system (AP TMS) can also be attributed to such devices.

A television system with a pulsed method of observing space and the possibility of measuring any parameters of the observed objects is called an active-pulse television measuring system.

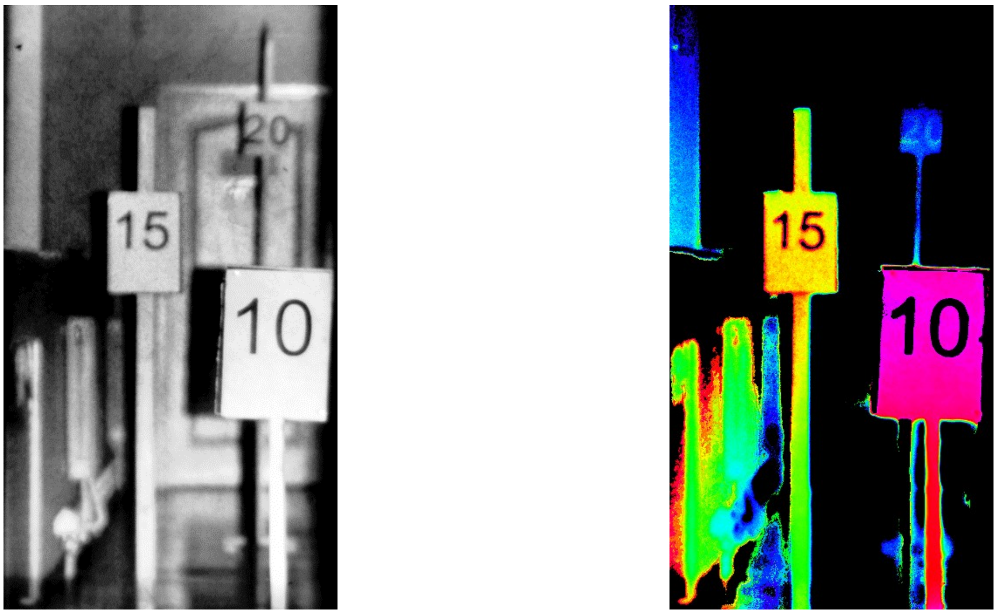
This paper discusses the development of the software which is able to build depth maps based on video data from an AP TMS television camera. The scientific novelty of this work is in the development of new high-speed software for building depth maps both directly from the AP TMS model in real time and from pre-recorded video material, which would provide a high-level interface for various types of measurements. It should be noted that using AP TMS it is possible to carry out distance measurements in adverse weather conditions (fog, snowfall, rain and other meteorological phenomena), therefore, the development of such software and its use in conjunction with AP TMS is an urgent task [1–3].

The developed software is designed for use in conjunction with the AP TMS model, mainly for research work in which it is necessary to measure distances to objects of interest under conditions of normal and reduced transparency of the propagation medium of optical radiation. Such an application could be the navigation of unmanned mobile robots, remote measurement of distances and various parameters of observed objects. The software is designed for a trained user familiar with the operating principles of AP TMS. Please note that the software, after final testing, will be distributed with the prior consent of TUSUR management and its developers.

The video sequence from the AP TMS television camera goes at a frequency of 50 frames per second, it is required to divide each 2 frames, and substitute the result of the division into a polynomial function to get the distance to the objects. It follows from this that the time taken by the software to build a depth map should be no more than 40 ms with a 7th order polynomial. Also, in order to obtain various kinds of scientific data, it is required to put various functionalities into the software, for example, depth map pseudo-coloring, recording the input video stream, setting up a polynomial function, etc.

Based on this, it can be noted that the development of such software is very important task, that would allow to configure lots of various parameters of the AP TMS and would have extensive image processing capabilities.

A depth map is an image where for each pixel, instead of brightness, the distance from the camera to the object is stored [4]. For a better visual perception of the distance, the possibility of presenting the depth map in color was implemented. An example of a depth map is shown below in Figure 1.



a **b**

Figure 1. Original frame (a) and the depth map (b) obtained using AP TMS

In Figure 1 b, the numbers on the shields are visible, but it is logical that they should not be visible. This is due to the fact that the depth map is built based on the ratio of the intensity of the reflected light. Due to the different reflectivity of surfaces, dark objects reflect less light, this leads to the fact that the ratio (and therefore the distance) to a dark surface is calculated with greater distortions than to a light one (fewer number of brightness quantization levels for dark surfaces and, as a result, a larger error rounding, as well as greater influence of black level in frames). This is another scientific problem that is currently being solved. Also, unreliable values in Figure 1 b were filtered by a threshold, which led to the zeroing of the values for the area of boards containing numbers.

2. Active-pulse television measuring system

2.1. General information about AP TMS

The dependence of the energy value of the light flux reflected from a separate object during one video frame obtained from the AP TMS photodetector is determined by the active vision area (AVA) [5]. AVA is created by emitting a space illumination pulse (SIP) from the pulsed laser semiconductor emitter (PLSE) and receiving the reflected light flux by a photodetector during the open state of the shutter. The control pulse for opening the shutter of the photodetector is called the photodetector gating pulse (PGP).

The use of multi-area distance measuring methods (MADM) in AP TMS is based on the formation of two groups of frames. An individual frame formed by an AP TMS television camera during its exposure is the sum of a group of frames captured by the system [6].

Moreover, the first group is the sum of frames with local AVA, i.e. both the duration of the SIP and PGP, and the duration of the gate delay remain constant. The second group of frames is formed by summing frames containing such AVAs, whose location in space is changed by changing the length of the shutter delay by some discrete step. The first group of frames is exposed in the odd frame of the AP TMS television camera, and the second group of frames is exposed in the even frame. Next, to obtain a linear dependence of brightness on range, it is necessary to divide the even frame into an odd one.

2.2. Structure of AP TMS

The block diagram of the AP TMS is shown in Figure 2.

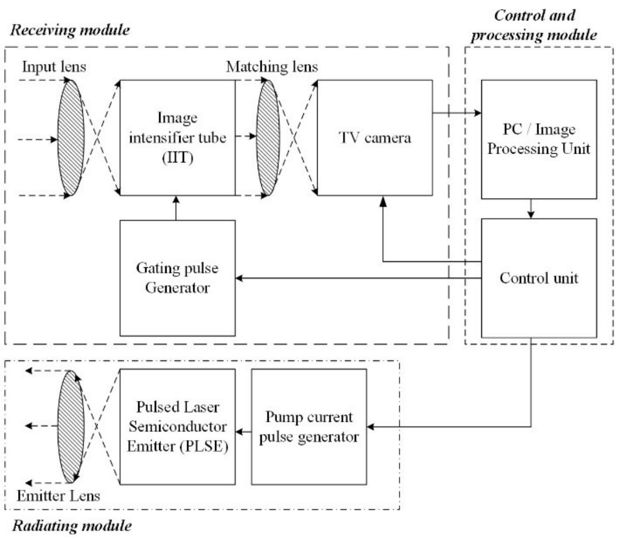

Receiving module
Control and processing module
Input lens
Matching lens
Image intensifier tube (IIT)
TV camera
PC / Image Processing Unit
Gating pulse Generator
Control unit
Pulsed Laser Semiconductor Emitter (PLSE)
Pump current pulse generator
Emitter Lens
Radiating module

Figure 2. The block diagram of the AP TMS

The block diagram of AP TMS (Figure 2) can be divided into three modules: radiating module, receiving module, control and processing module.

The emitting module consists of a pump current pulse generator, PLSE, and a lens. The pump current pulse generator controls the PLSE to ensure the strength of the luminous flux during a given radiation time. Light passes through the lens of the emitter, where it acquires the necessary spatial angle to illuminate the space. The receiving module consists of an image intensifier tube (IIT), in front of which there is an input lens that focuses the reflected optical flow. The image intensifier tube in this system primarily acts as a fast shutter, and it also converts the reverse light flux emitted in the frequency range invisible to the eye into visible light, increasing its intensity. The light flux leaving the image intensifier tube is focused on the matrix of the television camera due to the matching lens.

The TV camera is synchronized through the control unit with the image intensifier. The frame frequency of the TV sensor is 50 Hz, the operating frequency of the image intensifier is 5 kHz. Thus, for 1 frame, up to 100 frames formed on the image intensifier screen can be integrated in the TV sensor.

The image control and processing module allows receiving a video stream from an AP TMS television camera and perform the necessary processing of video frames to build depth maps.

2.3. Approximation of the measurement function

The use of polynomial approximation allows increasing the accuracy of building a depth map relative to a linear approximation, thereby increasing the accuracy of measuring the distance to objects of interest using AP TMS.

Approximation is a scientific method that involves finding a simplified mathematical model that is as close as possible to the original mathematical model. When approximating, the values at the points of the original function do not perfectly match the values at the corresponding points of the approximating function. To apply this method, the criterion of root-mean-square approximation (method of least squares) was chosen.

For AP TMS, linear and polynomial types of approximation were considered (in particular, polynomial functions of the 3rd, 5th and 7th degrees).

To calculate the coefficients of the approximating functions in accordance with the root mean square (RMS) approximation criterion, it is necessary to calculate the partial derivatives with respect to the general form of each of the approximating functions. Obtaining expressions for partial derivatives will lead to a system of linear algebraic equations that can be solved using the Cramer method.

The main parameter for assessing the proper approximation accuracy between approximating function and original function, presented in tabular form, was the root-mean-square error (RMSE). According to [7], an experiment was conducted and the following results were obtained (Table 1).

Table 1. RMS values based on the results of the approximation

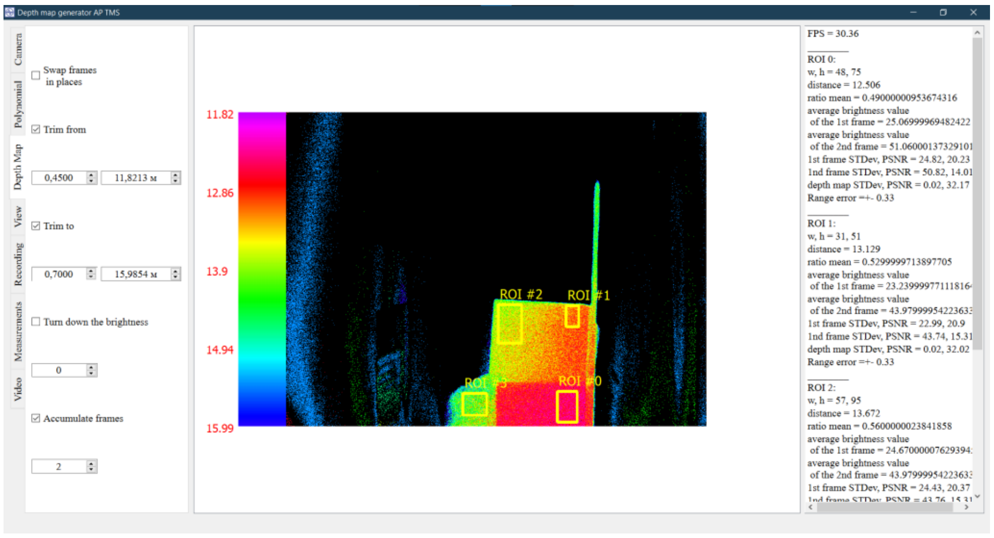
Approximation function	Linear	3rd degree polynomial	5th degree polynomial	7th degree polynomial
RMSE, m	0.1568	0.0573	0.0180	0.0125

The best result in terms of approximation accuracy is achieved when using an approximating polynomial function of the 7th degree compared to others.

3. Software implementation of the algorithm for building depth maps

3.1. Software implementation

The software was developed using the Python programming language [8] with the PyCharm IDE [9] using the libraries PyQt6, SciPy, SymPy, Ximea, Numba, NumPy, OpenCV, glob, time, sys, os, math, psutil, pyqtgraph [10–23]. The interface created using the PyQt6 library is shown in Figure 3.



☐ Swap frames in places

☑ Trim from

0,4500 | 11,8213 м

☑ Trim to

0,7000 | 15,9854 м

☐ Turn down the brightness

0

☑ Accumulate frames

2

```
FPS = 30.36
________
ROI 0:
w, h = 48, 75
distance = 12.506
ratio mean = 0.49000000953674316
average brightness value
 of the 1st frame = 25.06999969482422
average brightness value
 of the 2nd frame = 51.0600013732910
1st frame STDev, PSNR = 24.82, 20.23
1nd frame STDev, PSNR = 50.82, 14.01
depth map STDev, PSNR = 0.02, 32.17
Range error =+- 0.33
________
ROI 1:
w, h = 31, 51
distance = 13.129
ratio mean = 0.5299999713897705
average brightness value
 of the 1st frame = 23.239999771118164
average brightness value
 of the 2nd frame = 43.9799995422363
1st frame STDev, PSNR = 22.99, 20.9
1nd frame STDev, PSNR = 43.74, 15.31
depth map STDev, PSNR = 0.02, 32.02
Range error =+- 0.33
________
ROI 2:
w, h = 57, 95
distance = 13.672
ratio mean = 0.5600000023841858
average brightness value
 of the 1st frame = 24.6700000762939
average brightness value
 of the 2nd frame = 43.9799995422363
1st frame STDev, PSNR = 24.43, 20.37
1nd frame STDev, PSNR = 43.76, 15.31
```

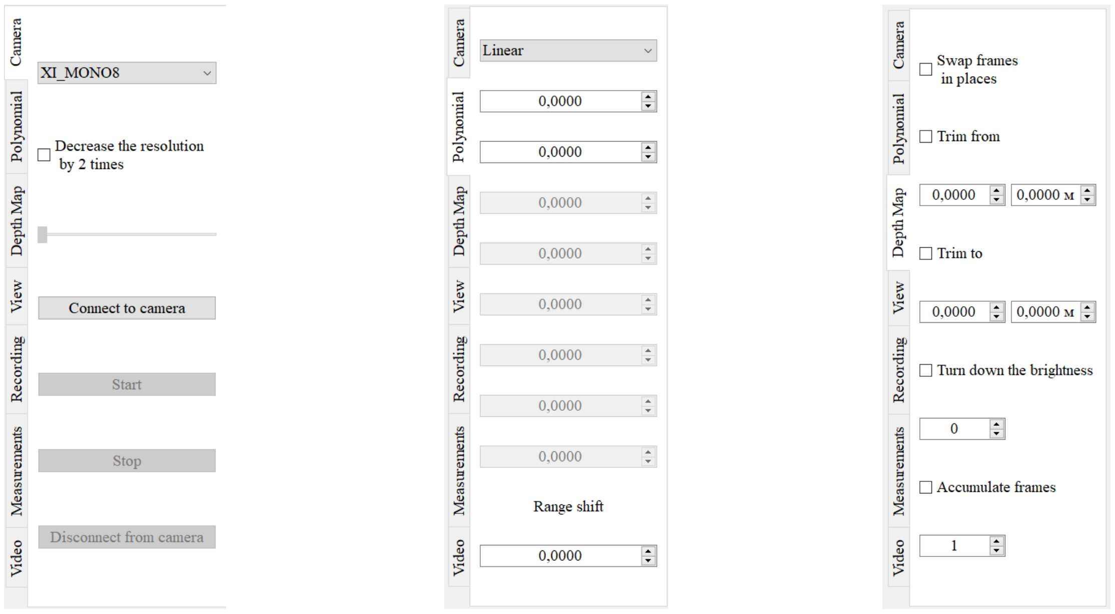
Figure 3. Interface of the developed program

The program interface contains such tabs as "Camera", "Polynomial", "Depth Map", "View", "Recording", "Measurements" and "Video".

The program interface provides a "Camera" tab (Figure 4, a), where you can connect and disconnect from the AP TMS television camera, adjust the image bit depth, resolution, and its amplification.

In the “Polynomial” tab (Figure 4, b), the required type of approximation of the measuring function is selected (from a linear function to a polynomial of the 7th degree), its coefficients are adjusted.

The “Depth Map” tab (Figure 4, c) contains the settings for building a depth map. In this tab, you can replace even and odd frames in places when calculating the depth map, filter the depth map by range (both using its normalized values, and directly specifying the ranges in meters). To increase the signal-to-noise ratio of frames, their accumulation and averaging is implemented.



☐ Decrease the resolution by 2 times

0,0000
0,0000
0,0000
0,0000
0,0000
0,0000
0,0000
0,0000
Range shift
0,0000

☐ Swap frames in places

☐ Trim from

0,0000 | 0,0000 м

☐ Trim to

0,0000 | 0,0000 м

☐ Turn down the brightness

0

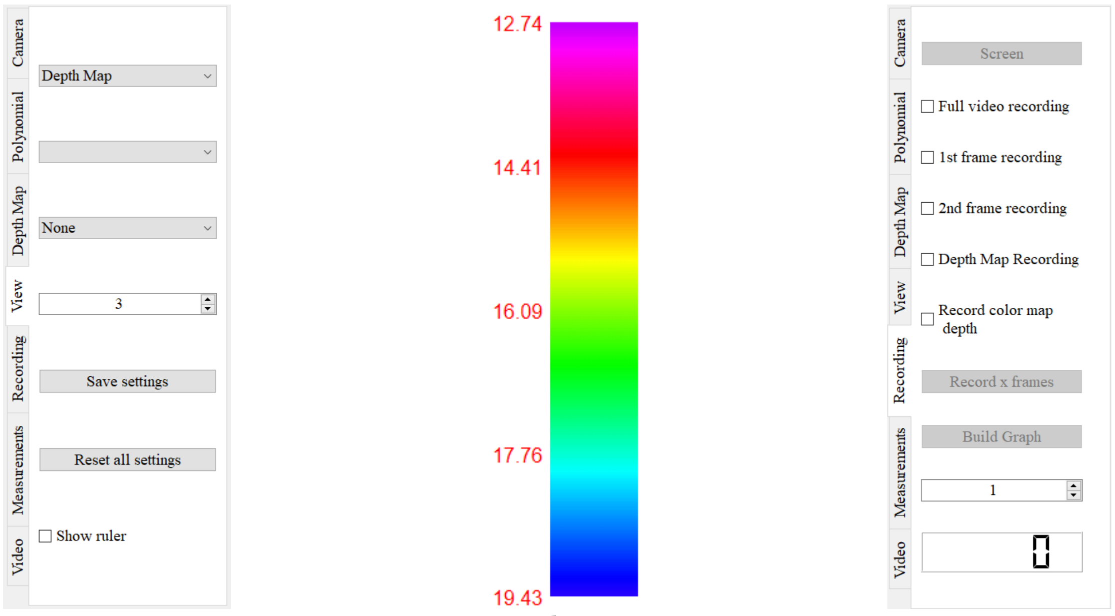
☐ Accumulate frames

1

a **b** **c**

Figure 4. Camera (a), Polynomial (b), and Depth Map (c) tabs

In the "View" tab (Figure 5, a) display mode on the screen can be selected (even frame, odd frame, depth map or colorized depth map). In the second field from the top, the type of coloring of the depth map can be selected, and the field following it allows you to select the type of filtering of the depth map and set the size of the filter mask. So, for the convenience and acceleration of the user's work, the possibility of saving all the program settings and resetting them to the default values is available. For the convenience of perception of range as a color on the colored depth map, it is possible to display a line of colors. An example of such line, with reference to ranges, is shown in Figure 5, b.



a b c

Figure 5. "View" tab (a), Ruler example (b), and "Record" tab (c)

Region of interest (Figure 6) can be configured directly in the frame output window. Pressing the left mouse button, holding it down and moving the mouse creates and adjusts the size of the region of interest.

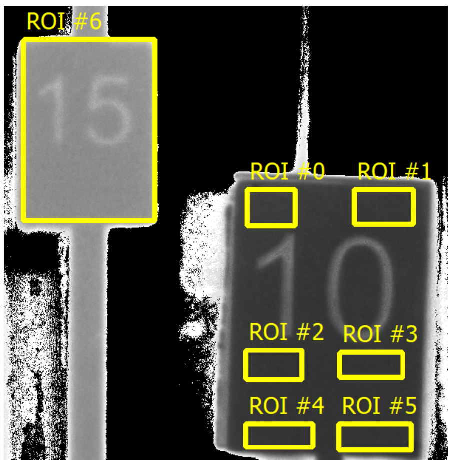


Figure 6. Region of interest

On the “Recording” tab (Figure 5, c), the possibility of saving frames from the program, as well as recording to a video file, is implemented. Also, in this tab, you can plot the amplitudes (Figure 7) for all selected regions of interest (ROI), this is used for the scanning mode.

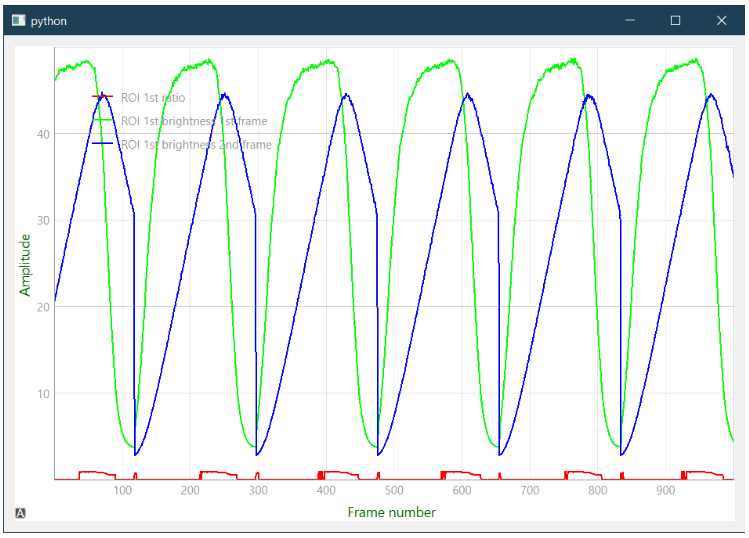


Figure 7. An example of a plot of frame ratio amplitudes plotted from ROI

Data from regions of interest are displayed in the interface (Figure 3) in the right part of the window.

In the "Measurements" tab (Figure 8, a), a reference frame is created by averaging the input frames. The number of frames in the average is configured in the field below. After the reference frames are formed, you can turn on the “Measurement” mode, in this mode, for each region of interest, the root-mean-square error (RMSE) and the peak signal-to-noise ratio (PSNR) for an even frame, an odd frame, and a depth map are calculated. Mean squared error (MSE), RMSE and PSNR are calculated according to the formulas below.

$$MSE = \frac{\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\left|I(i,j)-K(i,j)\right|^2}{mn}$$

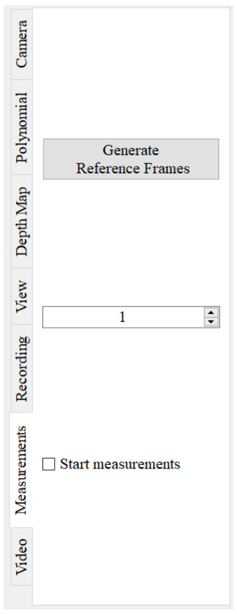
$$RMSE = \sqrt{MSE}$$

$$PSNR = 10\lg\left(\frac{MAX_I^2}{MSE}\right) = 20\lg\left(\frac{MAX_I}{RMSE}\right)$$

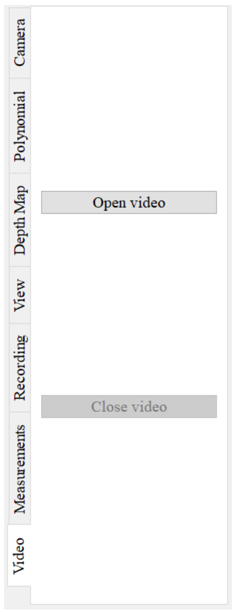
I and *K* – are two images of size $m \times n$, one of which is considered a noisy approximation of the other,

MAX_I is the maximum value accepted by the image pixel. When pixels are 8 bits wide, $MAX_I = 255$.

In the “Video” tab (Figure 8, b), opening a video file in *.avi format is implemented, and all the functionality described above can also work with a video file.



a



b

Figure 8. "Measurements" (a) and "Video" (b) tabs

The task of the software described above was to complete in 40 ms: counted two video frames, divided the even frame (contains the sum of frames from the 2nd group) into an odd frame (contained the sum of frames from the 1st group), filtered by the range of the resulting ratio, colored the resulting depth map with a selected color map, calculated and display range data from areas of interest using the selected approximation, and when necessary, recorded the result in a video file. In order to achieve the required performance of the program, for its operation in real time, the program is parallelized into threads, also CUDA cores of the computer's video card were used. In the first (main) thread, the interface is displayed and interactions with it are processed; in the second thread, frames are read from the video camera and a depth map is built. The third and fourth threads do not always exist, the third thread is created when creating regions of interest and is used to calculate the distance to the object and metrics (RMS and SNR), and the fourth thread is created to record the video stream. It is also worth noting that frames and video files are recorded without compression in order to exclude the loss of their information content.

As mentioned above, in the second thread, the depth map is built using calculations on the video card. Using the Numba library, a core function was written to perform parallel computing on a video card, with the help of which the required speed of building and coloring a depth map is achieved. Even and odd frames are moved to the video card memory, then parallel pixel-by-pixel division (into 32 threads) takes place, after this filtering of unreliable values (unreliability is caused by noise and/or low brightness) and colorization of the resulting depth map occurs.

The general algorithm for the functioning of the AP TMS depth map generator is shown in Figure 9.

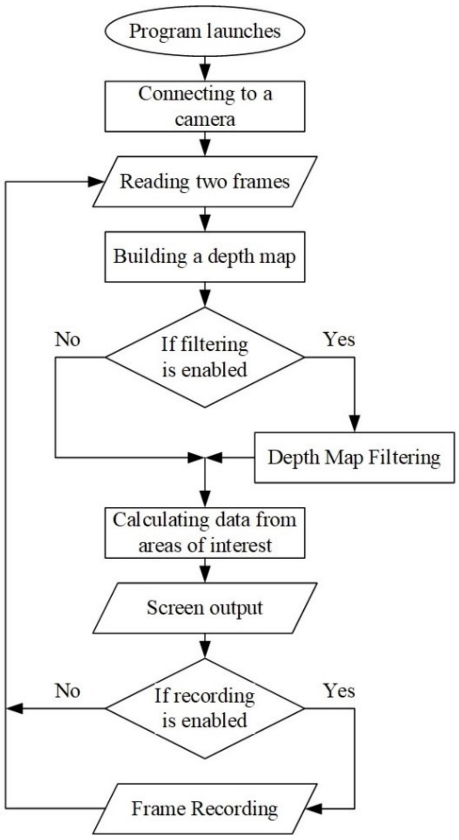


Figure 9. Block diagram of software operation

3.2. Performance evaluation

The data were obtained on a laptop, the parameters of which are given in Table 2.

Table 2. The main characteristics of the laptop

Parameter	Value
Processor model	Intel Core i7-12650H
Number of processor cores	10 (6 productive, 4 energy efficient)
Maximum processor frequency	4.7 ГГц
Discrete graphics card model	GeForce RTX 3060
CUDA kernel number	3840
The maximum clock frequency of the video card	1475 MHz
RAM	16 GB

The CPU is designed to execute multiple threads at the same time, or split one thread of instructions into several and, after executing them separately, combine them back into one thread in the correct order. There are few execution units in the processor, and the entire emphasis is on execution speed and reduction of idle time, which is achieved using cache memory and a pipeline [24].

The main function of the GPU is to work on a huge number of tasks that are independent of each other, which is why the GPU contains a huge number of execution units [25].

CUDA is a parallel computing platform and programming model developed by NVIDIA for general purpose GPU computing. With the help of CUDA, developers can significantly speed up computing applications using the performance of modern GPUs [26]. The CUDA core is different from the usual processor architecture. They strive to work effectively in parallel. CUDA cores are simpler in design, but there are a lot of them in a video card. If a central processor can have 2-16 cores, then there are thousands of them in a video card [27].

The original frames were read in the original resolution of 1544x2064 pixels and in the reduced resolution of 772x1032 pixels. The distance was calculated using a polynomial function of the 7th degree. Figure 10 shows a performance comparison graph for reading frames at native resolution. The processing time was counted from the moment two frames were received in the original resolution until the moment the depth map was displayed on the screen.

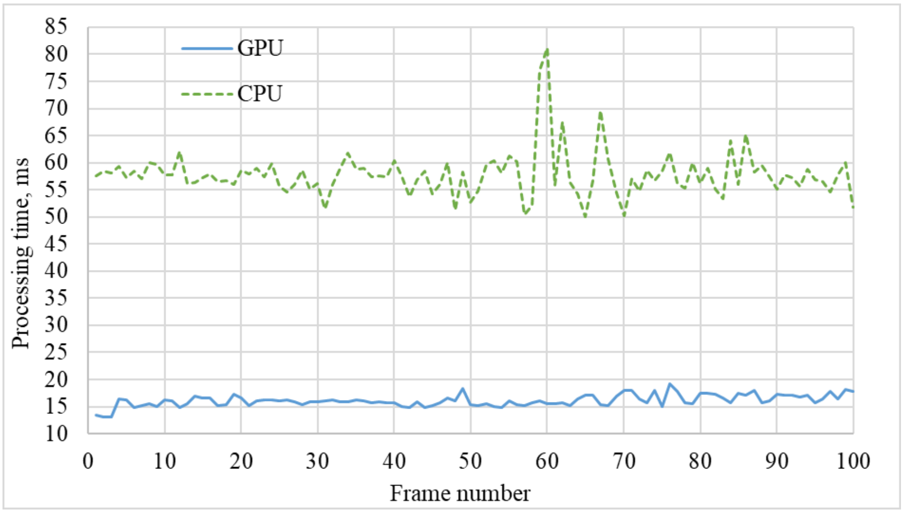


Figure 10. Graph comparing performance when reading frames in the original format

From the graph shown in Figure 10, it follows that image processing using CUDA cores is on average 3.5 times faster than processing on the processor. It can also be concluded that when using the central processor, it will not be possible to achieve real-time operation. Figure 11 shows a graph comparing performance when reading frames at reduced resolution.

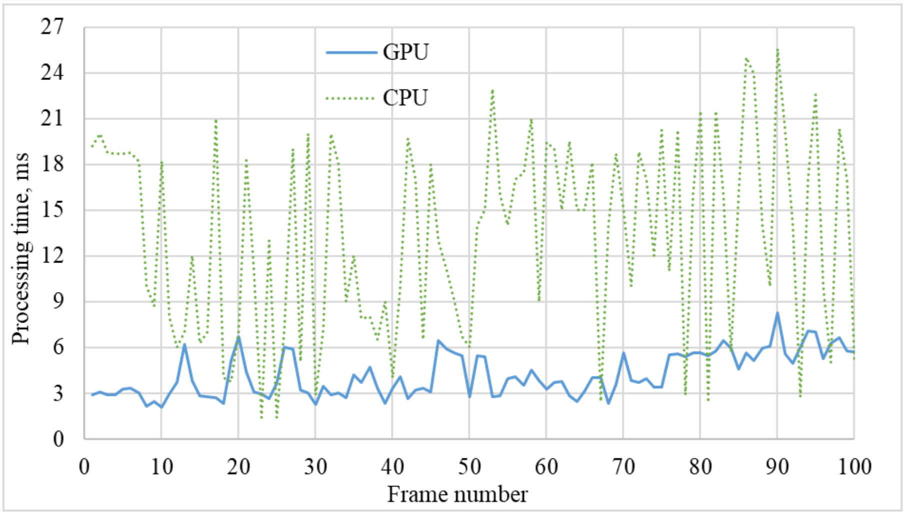


Figure 11. Graph comparing performance when reading frames at reduced resolution

From the graph shown in Figure 11, we can conclude that the CPU can work with small images (772x1032 pixels) in real time.

When conducting an experimental study of software performance, averaged values were found: to form one frame of a depth map with source frames in the uint8 format, 19 ms is spent, with source frames in the uint16 format - 31 ms. An image produced in uint8 format has 256 levels per pixel, while a 12-bit image (which the AP TMS television camera is capable of) can only be recorded in uint16 and has 4096 levels per pixel. 12-bit image is more accurate, but depth map building is slower.

Related works

Research for similar software was conducted. No complete analogues of the developed software have been found, which confirms the relevance of this development. It is also worth noting that there are many software solutions with similar functions - building spatial depth maps in real time, but at the expense of other principles for obtaining information about the distance to objects.

One such solution is the ZED SDK for the ZED line of stereo cameras from Stereolabs [28]. ZED SDK is optimized for real-time operation as it uses CUDA for computing. This software for a stereo camera has performance from 15 FPS with a stereo pair resolution of 4416×1242 pixels to 100 FPS with a stereo pair resolution of 1344×376 pixels [29]. With ZED it is possible to measure distances in the range from 0.3 to 20 meters. A distinctive feature of this software is the ability to use a neural network to improve the accuracy of construction and visual quality of depth maps.

Another example is the RealSense SDK from Intel [30]. RealSense SDK is developed for the Intel RealSense line of stereo cameras and is used to build spatial depth maps in real time. RealSense has a performance of 30 FPS at a stereo resolution of 3840x1080 and can measure distances in the range from 0.2 to 10 meters. The distinguishing feature of this software is the many ready-made and built-in software solutions, for example, building three-dimensional models, facial recognition or eye tracking.

Arena SDK [31] from LUCID Vision Labs is a software solution that supports all LUCID cameras. A special feature of the Arena SDK is its support for JupyterLab, which provides a built-in and pre-configured interactive development environment that can be used to test SDK features and camera performance. Software performance is determined by camera performance. For example, when using the Time of Flight camera Helios2+ [32], the frame resolution of which is 640 × 480 pixels and the measured range of which is from 0.3 to 8.3 meters, in High-Speed mode, the Arena SDK performance is 103 FPS.

Conclusions

As a result, software with a graphical interface was developed to build a colored depth map in real time based on frames obtained from the AP TMS television camera.

The performance of the AP TMS frame processing algorithm using the central processor and the video card was compared. If we talk about working in real time, then building a depth map with a maximum resolution is possible only using GPU CUDA cores. The processor can only work in real time with reduced video frame resolution. In addition, from the performance graphs, you can see how the frame processing time on the central processor changes dramatically. This is because the laptop system uses the CPU for its tasks. It was found that using the Numba library, which transfers calculations to the video card, it is possible to speed up the software by an average of 3.5 times. As a result of the work, it can be concluded that it was possible to achieve sufficient functionality and speed of the algorithm for building depth maps in real-time at the maximum resolution of video frames and calculating the distance using a polynomial of the 7th degree.

It should be noted that the developed software has a wide functionality. So, you can adjust the resolution and bit depth of the input video stream, adjust the degree of the polynomial and its coefficients, filter the depth map, adjust the filtering by distance, record the input video stream and the depth map, measure RMS and SNR, and much more.

Acknowledgements

The research was carried out at the expense of the grant of the Russian Science Foundation No. 21-79-10200 in TUSUR.

References

1. V.G. Volkov, “Active-pulse surveylance devices,” in Defense Technology Issues, 1994, no. 3(142), pp. 18–25.
2. V.E. Karasik, V.M. Orlov, “Laser vision systems: a tutorial”. Moscow, RF: MSTU im. N.E. Bauman, 2001, 352 p.

3. Sun, H. Y., Guo, H. C., & Li, Y. C. (2009, August). Per-formance analysis of range-gated active imaging system. In International Symposium on Photoelectronic Detection and Imaging 2009: Laser Sensing and Imaging (Vol. 7382, p. 73822E). International Society for Optics and Photonics. DOI: 10.1117/12.835576

4. Hartley R., Zisserman A. Multiple view geometry in computer vision. – Cambridge university press, 2003.

5. Kapustin V.V., Movchan A.K., Kuryachiy M.I. Vision area parameters analysis for active-pulse television-computing systems // Int. Siberian Conf. Control and Communications (SIBCON), 2017. - P. 1–4. doi:10.1109/sibcon.2017.7998432

6. E. S. Chaldina, A. K. Movchan, V. V. Kapustin and M. I. Kuryachiy, "Multi-Area Range Measurement Method Using Active-Pulse Television Measuring Systems," 2020 21st International Conference of Young Specialists on Micro/Nanotechnologies and Electron Devices (EDM), Chemal , Russia, 2020, pp. 293-297, doi: 10.1109/EDM49804.2020.9153500.

7. A. A. Tislenko, A. K. Movchan and V. V. Kapustin, "Improving the Distance Measurement Accuracy of Active-Pulse Television Measuring Systems Using Polynomial Approximation," 2022 International Siberian Conference on Control and Communications (SIBCON), Tomsk, Russian Federation, 2022, pp. 1-6, doi: 10.1109/SIBCON56144.2022.10002953.

8. Nagpal A., Gabrani G. Python for data analytics, scientific and technical applications // 2019 Amity international conference on artificial intelligence (AICAI). – IEEE, 2019. – P. 140-145.

9. PyCharm: The Python IDE for Professional Developers by JetBrains. Accessed: November, 29st, 2023. [Online]. Available: https://www.jetbrains.com/pycharm/.

10. Qt 6 – The latest version of Qt. Accessed: November, 29st, 2023. [Online]. Available: https://www.qt.io/product/qt6.

11. SciPy. Accessed: November, 29st, 2023. [Online]. Available: https://scipy.org/.

12. SymPy. Accessed: November, 29st, 2023. [Online]. Available: https://www.sympy.org/en/index.html.

13. XiAPI – ximea support. Accessed: November, 29st, 2023. [Online]. Available: https://www.ximea.com/support/wiki/apis/xiAPI.

14. Numba: A High-Performance Python Compiler. Accessed: November, 29st, 2023. [Online]. Available: https://numba.pydata.org/.

15. NumPy. Accessed: November, 29st, 2023. [Online]. Available: https://numpy.org/.

16. OpenCV. Accessed: November, 29st, 2023. [Online]. Available: https://opencv.org/.

17. glob - Unix style pathname pattern expansion. Accessed: November, 29st, 2023. [Online]. Available: https://docs.python.org/3/library/glob.html.

18. time - Unix style pathname pattern expansion. Accessed: November, 29st, 2023. [Online]. Available: https://docs.python.org/3/library/time.html.

19. sys – System-specific parameters and functions. Accessed: November, 29st, 2023. [Online]. Available: https://docs.python.org/3/library/sys.html.

20. os - Miscellaneous operating system interfaces. Accessed: November, 29st, 2023. [Online]. Available: https://docs.python.org/3/library/os.html.

21. math - Mathematical functions. Accessed: November, 29st, 2023. [Online]. Available: https://docs.python.org/3/library/math.html.

22. GitHub - giampaolo/psutil: Cross-platform lib for process and system monitoring in Python. Accessed: November, 29st, 2023. [Online]. Available: https://github.com/giampaolo/psutil.

23. PyQtGraph - Scientific Graphics and GUI Library for Python. Accessed: November, 29st, 2023. [Online]. Available: https://www.pyqtgraph.org/.

24. Kuck, David L., “The Structure of Computers and Computations.” (1978).

25. N. Stojanovic and D. Stojanovic, “Parallelizing Multiple Flow Accumulation Algorithm using CUDA and OpenACC,” ISPRS International Journal of Geo-Information, vol. 8, no. 9, p. 386, Sep. 2019, doi: 10.3390/ijgi8090386.

26. Nvidia, "CUDA Zone". Accessed: February, 10st, 2023. [Online]. Available: https://developer.nvidia.com/cuda-zone.
27. J. Gómez-Luna, J. M. González-Linares, J. I. Benavides, and N. Guil, “Parallelization of a Video Segmentation Algorithm on CUDA–Enabled Graphics Processing Units,” Lecture Notes in Computer Science, pp. 924–935, 2009, doi: https://doi.org/10.1007/978-3-642-03869-3_85.
28. Stereolabs Docs. Accessed: November, 29st, 2023. [Online]. Available: https://www.stereolabs.com/docs/.
29. ZED 2 Camera and SDK Overview. Accessed: November, 29st, 2023. [Online]. Available: https://cdn.stereolabs.com/assets/datasheets/zed2-camera-datasheet.pdf.
30. Intel RealSense Documentation. Accessed: November, 29st, 2023. [Online]. Available: https://dev.intelrealsense.com/docs/.
31. Arena SDK. Accessed: November, 29st, 2023. [Online]. Available: https://thinklucid.com/arena-software-development-kit/.
32. Helios2+ Time of Flight (ToF) IP67 3D Camera. Accessed: November, 29st, 2023. [Online]. Available: https://thinklucid.com/product/helios2plus-time-of-flight-tof-ip67-3d-camera/.