Real-Time Image Content Assessment for Underwater Robot Manoeuvring Based on Structural Tensor Analysis

Jakub Nawała and Bogusław Cyganek

AGH University of Science and Technology,
al. Mickiewicza 30, 30-059 Krakow, Poland
jakub.tadeusz.nawala@gmail.com

Abstract. The paper presents an efficient method for real-time image analysis for manoeuvring of the underwater robot. Image analysis is done after computing the structural tensor components which unveil rich texture and texture-less areas. To allow a power efficient underwater operation in real-time the method is implemented on the Jetson TK1 self-standing graphics card using the CUDA compute architecture. The laboratory experimental results show that the system is capable of processing about 40 Full HD images per second while allowing orientation toward texture specific regions for obstacle avoidance.

Keywords: robot manoeuvring, structural tensor, underwater operation, CUDA, real-time programming

1 Introduction

Robot manoeuvring and more specifically, obstacle avoidance, is one of the most fundamental functions that must be implemented first for any self-standing exploratory machine to operate properly. Given the operating conditions, solution providing this functionality must perform efficiently both in terms of power- and computational throughput efficiency. Thus, no conventional central processing unit (CPU) may be used. What is more, as will be later shown, mobile counterparts of CPUs proof insufficient for more complex image processing operations like this one. This necessitates utilisation of hardware accelerated mobile platform. For this purpose, NVIDIA Jetson TK1 board was chosen as it provides sufficient computational power, at the same time, retaining low energy consumption figures. Advantageous features of Jetson come from the presence of the first mobile, CUDA capable Tegra K1 graphics processing unit (GPU) on-board.

Video camera based system for underwater manoeuvring is given as a solution to the problem of growing need for autonomous inspections of maritime facilities. Amongst others, appearance of this phenomenon was described in [6]. It is important to underline that usage of standard camera, instead of sonar system, constitutes much cheaper approach that additionally allows to process data of much higher spatial resolution. On the other hand, requiring sufficient water transparency.

In order to identify rich texture and texture-less regions, structural tensor (ST) calculus is used. Procedure is based on assumption that in underwater environment any textured area represents potential obstacle. Using ST of the input image, it is possible to define highly coherent patches providing information about the presence of nearby objects [4].

Solution based on GPU was chosen due to its remarkable performance characteristics and its potential applications for execution time reduction. Latter of those was proven to be true by many scientific publications. For example, work of Wetzl *et al.* [11] shows GPU accelerated super-resolution (SR). Utilisation of hardware acceleration allowed authors to use their solution in interactive applications like image guided surgery. Similar approach was proposed in [9], which is another solution to SR problem based on bilateral total variation filtration. Apart from super resolving low resolution images, GPU may be used to enhance object detection algorithms, like the one presented by Chikkerur [2]. In this application, it allowed to achieve up to 10 fold performance boost.

Selection of NVIDIA Jetson TK1 as the main processing platform for self-standing application is natural due to its low power consumption. When measured under normal workloads, it oscillates around 4 W. For more computationally demanding operations, it reaches its maximum at 11.5 W.

Combining NVIDIA Jetson TK1 properties, allowing power efficient real-time image processing, and capability of ST algorithm to extract potential obstacles in underwater environs, it was decided to design, develop and test GPU-based solution for real-time image content assessment for underwater robot manoeuvring based on structural tensor analysis.

Reminder of this document consists of 4 sections. Section 2 provides more details on image content analysis based on structural tensor calculus. Section 3 includes overview of system architecture used and section 4 presents experimental results achieved. Section 5 concludes the paper giving both qualitative and quantitative analysis of the work done.

2 Image Analysis with the Structural Tensor

Most general execution flow of an algorithm used to perform image analysis with the structural tensor may be found in Fig. 1. Input gray-scale image is first used to calculate 3 components of its structural tensor. Those are later used to find corresponding coherence image. Then, coherence image is summed in square blocks of the given size and resultant lower resolution frame becomes an input for thresholding operation. At last, binary image is generated with white pixels denoting rich texture regions that represent potential obstacles in underwater environment.

2.1 Computing the Structural Tensor

A structural tensor $\mathbf{T}$ can be computed in each local and compact neighbourhood $R(\mathbf{x}_0)$ around a point $\mathbf{x}_0$ of a 2D image I, as follows [1][7][3]

$$\mathbf{T}\left(\mathbf{x}_0\right) = B_{R(\mathbf{x}_0)}\left(\mathbf{G}\mathbf{G}^T\right), \tag{1}$$

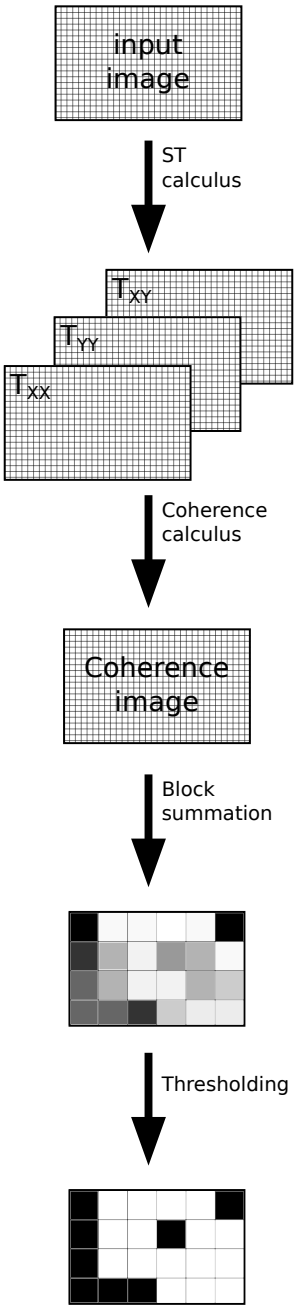


Fig. 1: Diagram of execution flow of image analysis algorithm

where $B_{R(\mathbf{x}_0)}$ denotes an averaging operator in a region R, with the centre at the point $\mathbf{x}_0$, and $\mathbf{G}$ is a gradient vector at *each* point $\mathbf{x}$ of R, that is, $\mathbf{x}\in R(\mathbf{x}_0)$. The gradient $\mathbf{G}$ at a point $\mathbf{x}$ of I, is defined as follows

$$\mathbf{G}(\mathbf{x}) = \begin{bmatrix} \frac{\hat{\partial}}{\hat{\partial}x} I(\mathbf{x}) \\ \frac{\hat{\partial}}{\hat{\partial}y} I(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} I_x(\mathbf{x}) \\ I_y(\mathbf{x}) \end{bmatrix}, \tag{2}$$

where $I_x(\mathbf{x})$ and $I_y(\mathbf{x})$ denote discrete spatial derivatives of I at the point $\mathbf{x}$, in the horizontal (x) and vertical (y) directions, respectively. In the presented system $G_{R(\mathbf{x}_0)}$ is a simplest discrete binomial filter [3].

It can be easily noticed that $\mathbf{T}(\mathbf{x}_0)$ constitutes a symmetric positive 2D matrix with elements describing averaged values of the gradient components in a certain neighbourhood around a point $\mathbf{x}_0$. Thus, the structural tensor $\mathbf{T}$ brings information on intensity changes not only at a point $\mathbf{x}_0$ but, more interestingly, also in its nearest neighbourhood.

It is worth noticing, that when $\mathbf{T}$ is computed at each point $\mathbf{x}$ of I, then each $\mathbf{T}(\mathbf{x})$ conveys information on overlapping regions around the point $\mathbf{x}$. Hence, it contains information on image texture and local curvature.

Inserting (2) into (1), the following is obtained

$$\mathbf{T} = B_R\left(\begin{bmatrix} I_x \\ I_y \end{bmatrix}\begin{bmatrix} I_x & I_y \end{bmatrix}\right) = B_R\left(\begin{bmatrix} I_xI_x & I_xI_y \\ I_yI_x & I_yI_y \end{bmatrix}\right) = \begin{bmatrix} T_{xx} & T_{xy} \\ T_{yx} & T_{yy} \end{bmatrix}. \tag{3}$$

To simplify the equations, in the aforementioned derivations, the point $\mathbf{x}$ was omitted since averaging by the filter B_R is over a set of points in R, as already described.

Concluding, the structural tensor $\mathbf{T}$ is built from four components: T_{xx}, T_{yy}, T_{xy} and T_{yx}, each of those being averaged multiplication of partial derivatives of input image's intensity signal with respect to horizontal and vertical spatial coordinate.

2.2 Texture Analysis for Robot Manoeuvring

In order to practically verify the concept of robot manoeuvring based on texture analysis, consideration done in this subsection will be accompanied by the presentation of execution steps outcomes for real underwater image acquired (see Fig. 2).

As it was already shown, first step is calculation of structural tensor of the input image. Result of performing this operation may be found in Fig. 3. It should be noted that rich texture regions can be roughly estimated this early in the processing procedure, using simple visual inspection. 3 components of ST are now used to calculate *coherence* parameter for each pixel of the input image. Formula used to do it is given in equation 4.

$$coherence = \frac{(T_{xx} - T_{yy})^2 + (2T_{xy})^2}{(T_{xx} + T_{yy})^2} \tag{4}$$

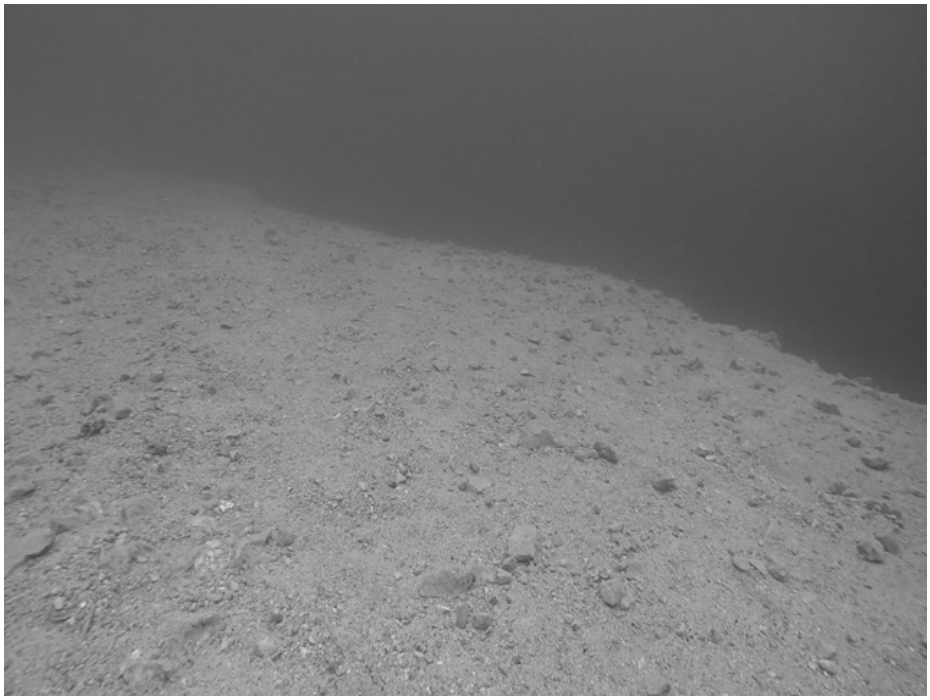
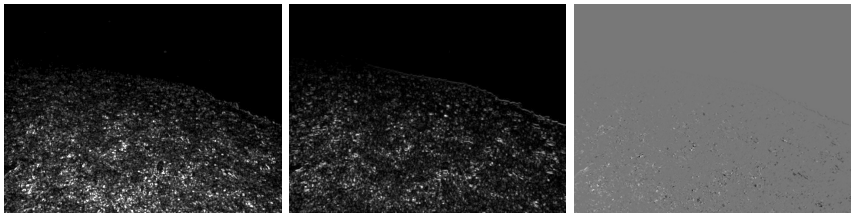


Fig. 2: Input gray-scale image captured in Zakrzówek lake at depth of 15 meters



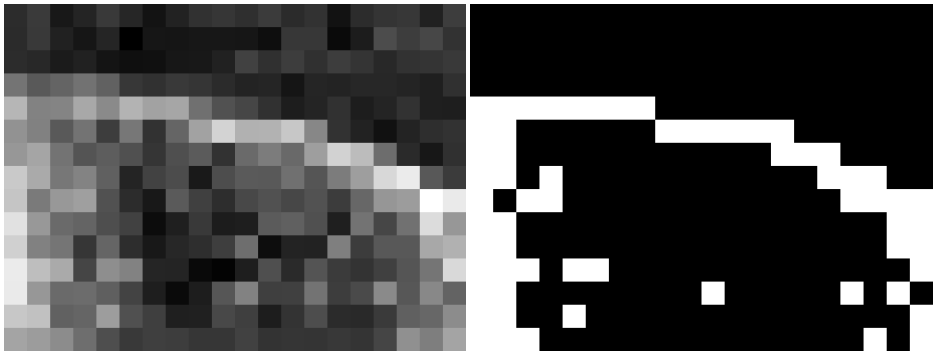
(a) T_{xx} (b) T_{yy} (c) T_{xy}

Fig. 3: Structural tensor components calculated

Calculated values are now summed in square blocks of arbitrary size. For images shown in this paper, block size of 32 was chosen, as it provides most satisfactory end results. As a last step, block-summed coherence image is filtered with thresholding filter, which sets to maximum all the pixels being larger than the middle point between largest and smallest value of the block. This operation may be expressed by the following equation:

$$b(x,y) = \begin{cases} 1 & \text{, when } C_{block}(x,y) > \frac{max(C_{block}) - min(C_{block})}{2} \\ 0 & \text{, otherwise,} \end{cases} \tag{5}$$

where $b(x,y)$ represents pixel in output binary image at horizontal coordinate x and vertical coordinate y, and C_{block} represents block-summed coherence image. Final binary image juxtaposed with block-summed coherence frame may be seen in Fig. 4.



(a) Summed in blocks (b) After thresholding

Fig. 4: Coherence image summed in blocks and after thresholding

Willing to further proof proper operation of the procedure proposed, second underwater input image and its final binary image generated is given in Fig. 5.

As it can be judged by visual inspection, solution given shows strong potential for application in underwater robot manoeuvring. Resultant binary images yield good approximation of obstacles appearing on the path of the exploratory machine and as such, may be used to avoid unwanted collisions.

3 System Architecture

Proposed image content analysis for underwater robot manoeuvring was implemented as a software application. Its general operation flow was given in Fig. 1. Most of the components were implemented on CPU and hardware acceleration was utilised only for computationally intensive structural tensor computation.

(a) Input image (b) Binary map

Fig. 5: Second input image from Zakrzówek lake and resultant binary map

As for any GPU accelerated image processing task, CUDA threads were mapped onto input image's pixels. Architecture of this mapping may be found in Fig. 6. It should be pointed out that each block of threads is dedicated for a single image row. Number of 384 threads per block was chosen due to its best performance characteristics for tested use-cases. In case of images wider than 384 pixels, single thread from a given block is used to serve pixels at rows indexes being multiplication of its original position. However, not every thread must process equal number of pixels and hence, images of all possible widths may be used.

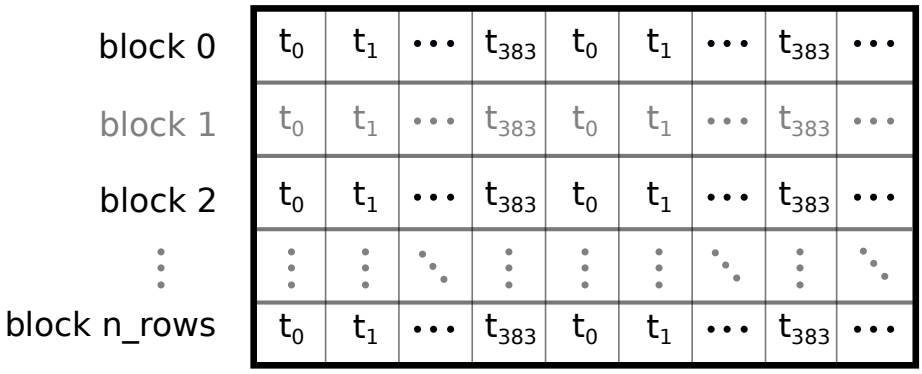


Fig. 6: Mapping of CUDA threads over input image's pixels (t_n represents n-th thread and n_rows means number of input image's rows)

In addition to proper CUDA threads utilisation, texture memory of GPU was used to assure best possible computational throughput. Its presence is justified

as almost any image processing operation shows strong spatial locality and this type of memory is optimised just for those kinds of tasks.

4 Experimental Results

As execution time constitutes the crucial factor in real-time applications, it was the main scope of the experimental tests conducted. More specifically, execution time of the most computationally intensive step of the algorithm proposed was measured. In this case, it was the structural tensor computation that was consuming major part of processing resources.

For better visualisation of GPU predominance over standard CPU implementations, tests were done on 3 platforms, namely: Jetosn's Tegra K1 GPU, Jetson's ARM Cortex A-15 mobile CPU and standard Intel Xeon X5650 CPU. Comparison with Cortex A-15 is justified as it posses mobile properties necessary for self-standing applications. On the other hand, Xeon X5650 is given to show how mobile GPU is capable of outperforming more standard processing platforms, when used in image processing tasks.

In order to verify what is the influence of input image size, measurements were performed for 3 standard resolutions: VGA (640x480), HD (1280x720) and Full HD (1920x1080). Additionally, results were averaged over 5 iterations to assure measurements stability.

Figure 7 shows execution times achieved. Detailed information about the results may also be found in Table 1. It is readily visible that Tegra K1 GPU outperforms both CPUs. What is more, its superiority becomes more relevant for higher resolutions. At Full HD, GPU executes almost 20 times faster, as compared to ARM Cortex A-15, and more than 11 times for Xeon X5650. On top of that, Tegra K1 is capable of processing up to 40 Full HD frames per second, being the figure sufficient to treat this application the real-time solution.

Table 1: ST computation frames per second throughput comparison for selected input image resolutions (MP stands for *megapixel* meaning 1 million pixels); additionally compared with average power consumption

Image resolution	Frames per second		
	Tegra K1 GPU	Xeon X5650	ARM Cortex A-15
VGA (0.31 MP)	65	27	16
HD (0.92 MP)	44	7	5
Full HD (2.07 MP)	39	3	2
Avg. power cons. [W]	5 [8]	95 [5]	0.6 [10]

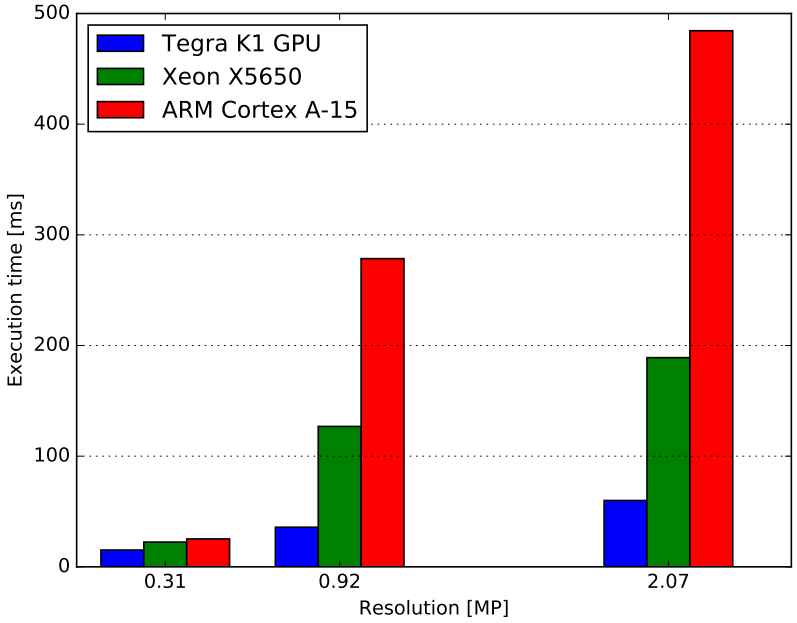


Fig. 7: Comparison of ST computation execution time when ran on Tegra K1 GPU, ARM Cortex-A15 mobile CPU or Xeon X5650 standard CPU

It is worth mentioning that good performance scaling versus input image size for GPU, comes from its massively parallel architecture. It provides sufficient number of processing resources dedicated for each pixel, even for resolutions as high as 2 MP. Following this consideration, it can be assumed that there is some theoretical boundary for input image size, which makes GPU execution time rise faster than for 3 presented cases. Nevertheless, good performance figures for resolutions up to 2 MP proofs to be sufficient for most practical applications.

5 Conclusions

In this paper, usability of GPU-based platform, combined with structural tensor calculus, was investigated as the mean to perform real-time image content analysis for underwater robot manoeuvring. Given the laboratory execution time results, it was proven that NVIDIA Jetson TK1, with Tegra K1 GPU on-board, does constitute efficient platform for real-time image processing. Additionally, taking into account its mobile properties, it was shown to be a good solution for computer vision provision in self-standing exploratory machines. On top of that, structural tensor calculus was presented as a tool for extraction of information about nearby obstacles in underwater environs.

It is important to point out that lower power consumption figures of on-board computer vision provider means not only better utilisation of resources, but also possibility to use the for other application-critical tasks.

Having verified Jetson TK1 serviceableness for real-time image analysis and given structural tensor calculus functionality for obstacle avoidance, it is concluded that solution proposed, may be used in potential submarine exploratory application. It is thus planned to practically verify this assumption.

Acknowledgement

This work was done as a realisation of the research project financed by The Polish National Science Centre (NCN) and has the following project number UMO-2014/15/B/ST6/00609.

References

1. Bigün, J., Granlund, G. H. & Wiklund, J. Multidimensional Orientation Estimation with Applications to Texture Analysis and Optical Flow. *IEEE Trans. Pattern Anal. Mach. Intell.* **13,** 775–790 (1991).
2. Chikkerur, S. *CUDA Implementation of a Biologically Inspired Object Recognition System* <http://code.google.com/p/cbcl-model-cuda/> (2008).
3. Cyganek, B. *An Introduction to 3D Computer Vision Techniques and Algorithms* ISBN: 047001704X (John Wiley & Sons, 2009).

4. Cyganek, B. *Object detection and recognition in digital images: theory and practice* (Wiley, Hoboken, NJ, 2013).
5. *Intel Xeon X5650 CPU: Review, Benchmarks, and Specs* <http://processors.specout.com/l/230/Intel-Xeon-X5650> (2010).
6. Jacobi, M. Autonomous Inspection of Underwater Structures. *Robot. Auton. Syst.* **67,** 80–86. ISSN: 0921-8890 (May 2015).
7. Jahne, B. *Digital Image Processing: Concepts, Algorithms, and Scientific Applications* 4th. ISBN: 3540627243 (Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1997).
8. Kelion, L. *CES 2014: Nvidia Tegra K1 offers leap in graphics power* <http://www.bbc.com/news/technology-25618498> (2014).
9. Mitzel, D., Pock, T., Schoenemann, T. & Cremers, D. in *Pattern Recognition: 31st DAGM Symposium, Jena, Germany, September 9-11, 2009. Proceedings* (eds Denzler, J., Notni, G. & Süße, H.) 432–441 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2009). ISBN: 978-3-642-03798-6. doi:10.1007/978-3-642-03798-6_44. <http://dx.doi.org/10.1007/978-3-642-03798-6_44>.
10. Shimpi, A. L. *The ARM vs x86 Wars Have Begun: In-Depth Power Analysis of Atom, Krait & Cortex A15* <http://www.anandtech.com/show/6536/arm-vs-x86-the-real-showdown> (2013).
11. Wetzl, J. *et al. GPU Accelerated Time-of-Flight Super-Resolution for Image-Guided Surgery* in *Bildverarbeitung für die Medizin* (eds Tolxdorff, T. & Deserno, T. M.) (Heidelberg, 2013), 21–26. <http://www5.informatik.uni-erlangen.de/Forschung/Publikationen/2013/Wetzl13-GAT.pdf>.