Vol. 23, No. 3/4, 2014

Machine GRAPHICS & VISION

International Journal

Published by
The Faculty of Applied Informatics and Mathematics
Warsaw University of Life Sciences – SGGW
Nowoursynowska 159, 02-776 Warsaw, Poland
in cooperation with
The Association for Image Processing, Poland – TPO

Machine GRAPHICS & VISION

Editor-in-Chief: Leszek J. Chmielewski, The Faculty of Applied Informatics and Mathematics – WZIM, Warsaw University of Life Sciences – SGGW

Past Editor and Founder: Wojciech S. Mokrzycki

Honorary Editor: Juliusz L. Kulikowski

Editorial Office Secretary: Luiza Ochnio, Warsaw University of Life Sciences – SGGW

TEXsetting by Urszula Kruś **Language verification by Beata Konikowska**

Vol. 23, No. 3/4, 2014

Machine GRAPHICS & VISION

International Journal

Published by
The Faculty of Applied Informatics and Mathematics
Warsaw University of Life Sciences – SGGW
Nowoursynowska 159, 02-776 Warsaw, Poland
in cooperation with
The Association for Image Processing, Poland – TPO

The goal of **Machine GRAPHICS & VISION** is to provide a medium for exchanging information on image processing and image communication between the computer and its environment.

The journal concentrates on computational methods of synthesis and analysis of the models underlying computer generated, analyzed or processed imagery. In particular, the emphasis is on the methods of:

- image synthesis,
- computational geometry,
- three-dimensional imaging,
- computer graphics art,
- diagrammatic representation and reasoning,
- graphical data presentation and knowledge-oriented visualization,
- animation,
- image models and transforms,
- mathematical approaches to image analysis and processing
  - analytical geometry
  - statistical methods
  - functional analysis and spectral methods
  - logical methods
  - formal linguistic methods
  - fractals
- visual pattern matching and recognition
- three-dimensional scene mapping and modeling
  - statistical methods
  - spectral methods
  - structural methods
- motion capture
- composite image understanding
- monocular and stereo vision
- modeling of human visual perception

The journal encourages papers on research and applications to processing of scientific, medical and technological images, engineering design, graphical arts, environment protection, etc. State-of-the-art, expository as well as tutorial papers in the above-mentioned areas are welcome.

Published Quarterly

Subscription. **MG&V** (ISSN 1230–0535) can be ordered directly through Foreign Trade Enterprise ARS POLONA Joint Stock Company, Obrońców 25, 03-933 Warszawa, Poland
Fax: +48 22/509 86 48, Tel. +48 22/509 86 63, e-mail: arspolona@arspolona.com.pl

Visualization and Interaction in the Atlas of the Human Brain, Head and Neck

Wieslaw L. Nowinski
Department of Radiology, University of Washington, Seattle, USA

Abstract. Our ultimate objective is to create a holistic and reference atlas of the whole adult human brain along with the head and neck. Several techniques have been employed to create atlases. Here we discuss the atlas design and use from a point of view of two key techniques, visualization and interaction. For visualization, surface rendering of a geometrical model of the brain, head and neck is employed. Geometrical model ensures anatomic parcellability, high (sub-pixel) resolution, editibility, extendibility, structure separability, structure-to-structure conflict detection, and integration a knowledge-based content with the atlas. Interaction allows the user to create and explore any region of interest along with its surroundings just with a few clicks, taking into account that the atlas provides a rich set of functions and the number of atlas components is about 3,000. There are seven types of interaction enabling to: select and deselect tissue classes/groups/individual structures, do real-time manipulation, do virtual dissections, select and scroll the original scans, query a structure to get its label or location, get stereotactic coordinates and measure distances, and support other functionality. This design of visualization and interaction provides a fast and easy to use solution, and allows the atlas to run on desktop and mobile iPad and Android-based platforms.

Key words: atlas, brain, head, neck, visualization, interaction, geometrical model, integration, knowledge base, desktop platform, mobile platform.

1. Introduction

As the XXIst century is considered the century of the brain, numerous projects address modeling, mapping, and atlasing of the human brain. Our contribution to these efforts is to create adult human brain atlases and develop atlas-based applications. So far, we have developed 35 brain atlases licensed to 63 companies and institutions, and made available to medical societies, organizations, medical schools, and individuals.

Our ultimate objective is to create a holistic atlas of the whole adult human brain along with the head and neck. The atlas is three-dimensional (3D), advanced, detailed, reference, interactive, accurate, realistic, high resolution, fully parcellated, completely labeled, spatially consistent, stereotactic, user friendly, extendable (scalable), composable, dissectible, explorable, and modular. To date, we have constructed a 3D atlas from multiple 3 and 7 Tesla MR (Magnetic Resonance) and high resolution CT (Computed Tomography) in vivo scans of a single brain specimen to ensure spatial consistency and extendibility. The virtual model in the atlas contains structure [3], intracranial vasculature [4], white matter tracts [5], cranial nerves with nuclei [6], head muscles and glands [7], extracranial vasculature [8], and a complete skull added recently reconstructed

from a CT scan (Nowinski et al., Three-dimensional stereotactic atlas of the adult human skull correlated with the brain, cranial nerves and intracranial vasculature, submitted to Journal of Neuroscience Methods 2014). The atlas design criteria and tools for atlas construction have been addressed earlier [10].

Several techniques are required to create atlases, including image processing, computer graphics, modeling, registration, and image and model editing. The key techniques employed for the atlas use are visualization and interaction. The goal of this work is to discuss the atlas design and use from a point of view of visualization and interaction.

2. Visualization

Two main visualization techniques are volume rendering and surface rendering. Volume rendering handles a volumetric representation of data and surface rending a polygonal data representation. An advantage of the volumetric representation is that the scans used for atlas creation were originally acquired as volumetric data (sometimes with prior resampling). However, the parcellability of volume rendered images is limited as image intensities are translated to (R,G,B) color and opacity by means of transfer functions. Despite the use of various extensions of the standard transfer functions, such as [11], [14], anatomic parcellation of a scan is practically not feasible. For instance, if each pixel in the scan would be labeled (colored) individually, and assuming that a single pixel is processed within 30 seconds (to perform pixel reading, making decision about what tissue it belongs to, assigning label or color, and saving it), the main structural MR scan with 20 million voxels would be handled for about 100 years.

Anatomic parcellability is feasible by means of geometric modeling. Prior to forming a geometric model, the acquired scan has to be accurately parcellated (for instance contoured) which is a tedious and time consuming process. From the parcellated scan, its 3D polygonal model can be created by applying, e.g., the Marching cubes algorithm [2]. When multiple scans are used (as is in our case), they must be spatially registered first and their segmented models merged. The polygonal representation enables a high (subpixel) parcellation of the scans and provides control over editing of data (i.e., images and models), resulting in a more accurate and realistic model. It also has additional advantages as listed and compared in Tab. 1. Therefore in our atlas, the brain, head and neck are represented as polygonal models visualized by surface rendering.

3. Interaction

The key interaction requirement was to allow the user to create and explore any region of interest along with its surroundings just with a few clicks, taking into account that the number of atlas components is about 3,000. Moreover, the atlas provides a rich

Tab. 1. Volumetric versus polygonal representation

Feature	**Volumetric representation**	**Polygonal representation**
Data representation	Originally acquired or resampled	Must be created
Spatial resolution	Voxel size bounded	Sub-voxel
Parcellation	Fast but of low resolution and anatomically inaccurate; anatomic parcellation infeasible	Time consuming and accurate; anatomic parcellation feasible
Integration of knowledge-based content	Very difficult, if possible at all	Possible; requires dedicated tools
Extendibility	Easy (by image fusion)	Easy (by object import)
Editibility	Easy but extremely time consuming (by editing individual voxels)	Easy (by editing polygonal objects); requires dedicated tools
Structure separability	Low (fuzzy borders)	Very high, as borders are defined, enhanced by structure color coding
Structure-to-structure conflict detection	Non existent	Requires manual or automatic handling

set of functions [10] enabling component selection, 3D model display and real-time manipulation, brain/head/neck compositing (assembly) and decompositing (disassembly), structure labeling, virtual dissections, scan scrolling, 3D stereotactic coordinate readout, 3D distance measure, and highlighting of individual components. Therefore, the main challenge was to design interaction to be fast and easy to use.

There are seven types of interaction in the atlas, which enable to: 1) select and deselect tissue classes, groups and individual structures (during 3D scene compositing and/or decompositing); 2) do real-time manipulation of the composed 3D scene (rotate, zoom, pan, and set views); 3) do virtual dissections by means of 3D cutting of the cerebrum, cerebellum, brainstem, spinal cord, white matter and skull in order to expose structures lying inside; 4) select and scroll the original scan in axial, coronal, and sagittal planes displayed in 3D; 5) query a structure to get either its label (and diameter for the vessels) or location in the composed 3D scene; 6) get stereotactic coordinates and measure distances; and 7) support other functionality (namely, image saving to an external file, starting/stopping 3D scene auto rotation, clearing labels, and getting information and help). Typically during brain exploration, the user may perform all seven types of interactions.

In order to facilitate interaction as well as to expedite structure selection and 3D scene compositing and/or decompositing, the atlas components have been grouped at multiple levels. The atlas content has been divided into tissue classes (modules), groups within each tissue class, and individual components within each group. A component may be single or composed. There are at present 17 modules containing the central nervous system (CNS), deep structures, ventricles, white matter, white matter tracts, intracranial arterial system, intracranial venous system, head muscles, glands, extracranial arteries, extracranial veins, skull, skin, neck, visual system, and auditory system. The CNS is available permanently and the remaining modules are selectable by the user from a (4×4) module matrix. Examples of groups include 12 pairs of the cranial nerves (CN I – CN XII) within the cranial nerve tissue class or ICA (internal carotid arteries), ACA (anterior cerebral arteries), MCA (middle cerebral arteries) and PCA (posterior cerebral arteries) within the intracranial arterial system tissue class. An example of an individual component is the basilar artery (BA) which can be handled as a single component or a composed component (i.e., the BA itself along with its tributaries). The de/selection operations can be performed at 5 levels supporting the paradigm "from blocks to brain"; namely, at the level of all tissue classes (components), tissue class cluster (brain, head, neck or systems), tissue class, group within a class, and individual component within a class (with or without its sub-components).

4. Results and discussion

The first edition of the atlas of the human brain, head and neck has been developed and made available [9]. Figure 1 shows surface rendered images of the brain (along with the intracranial vasculature) and the head (with the skull, muscles, extracranial vasculature, cranial nerves, and visual system). Figure 2 presents the user interface with the controls for interaction, including the module matrix for selection of all the tissue classes, tissue class cluster and/or individual classes; actually selected modules with their panels enabling group selection; anatomical indices for individual component selection; and functional buttons.

This design of visualization and interaction provides a fast and easy to use solution. Moreover, one of the requirements of the atlas design was affordability. At present, the atlas runs on a standard PC and MAC equipped with a graphics card supporting OpenGL 2.1 or higher library. The current design of visualization and interaction has allowed porting the atlas to mobile platforms, including iPad [12] and Android-based [13].

This work in a long-term effort aiming to create a holistic atlas of the human brain, head and neck. Visualization and interaction are two key techniques enabling atlas use. Future atlas development may require more advanced visualization and interaction. Developing a hybrid volume and surface renderer would facilitate to display any scans imported to the atlas by the user. Stereoscopic viewing (especially to visualize tubular

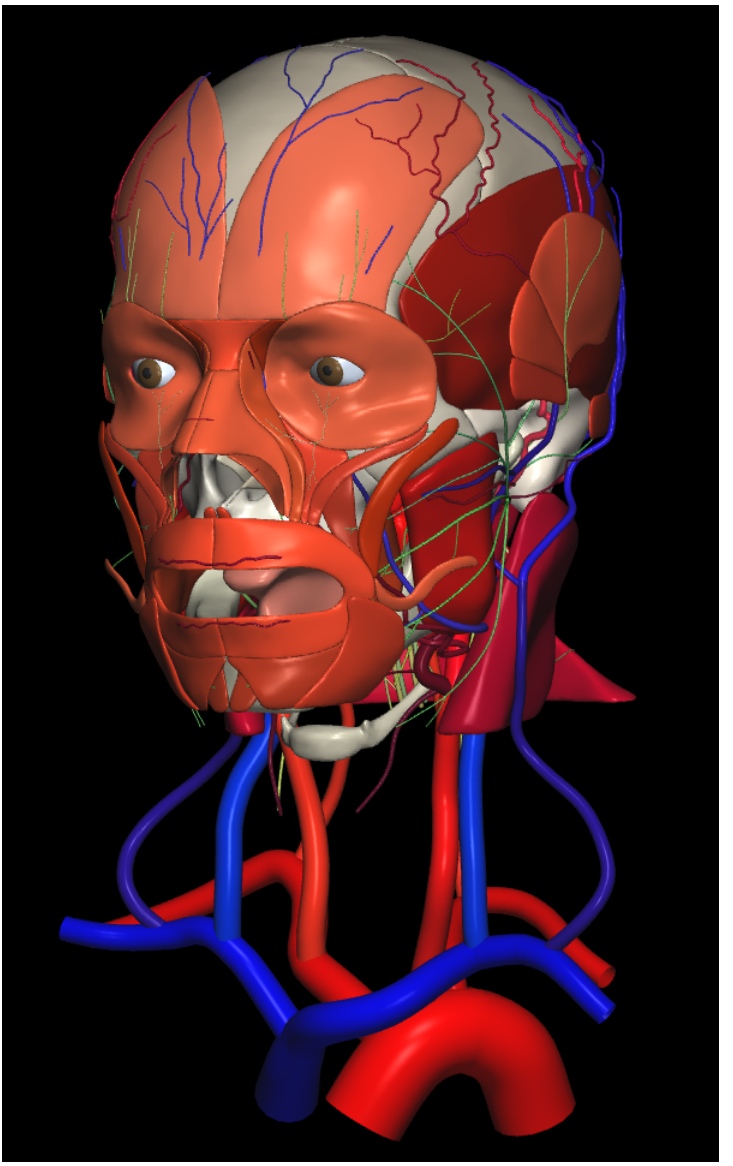


Fig. 1. Surface rendered images of the brain and the head.

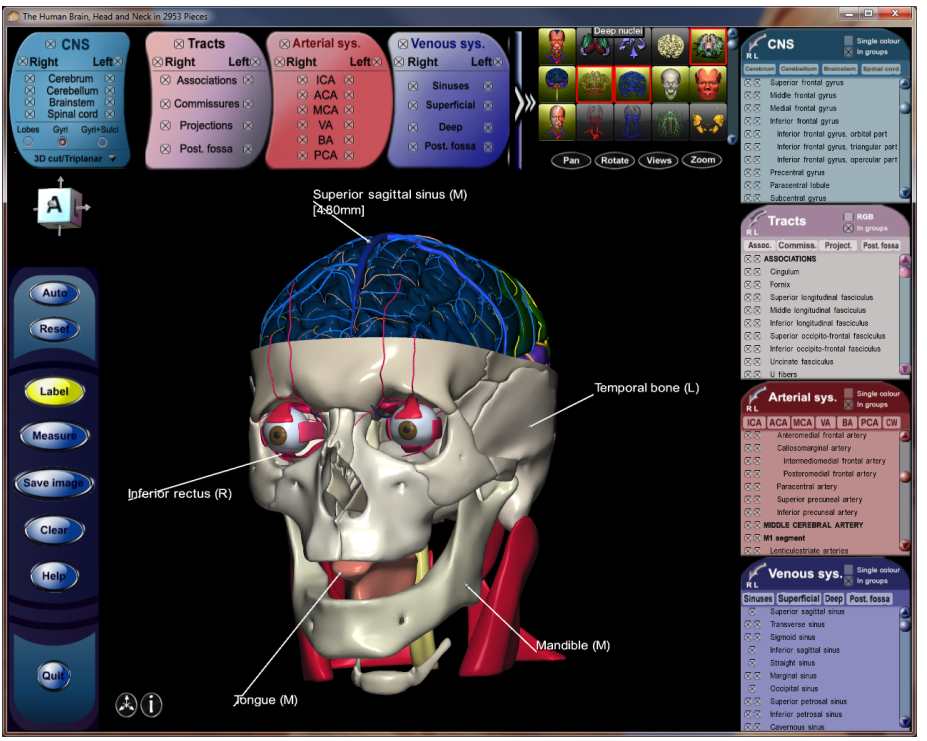


Fig. 2. User interface with the controls for interaction.

structures, such as vessels and cranial nerves) and 3D two-hand intuitive interaction, as applied earlier in our neurosurgery planning system [1], would further enhance visualization and interaction (though this approach may reduce affordability of the atlas).

5. Acknowledgement

The atlas development work was funded by ASTAR, Singapore. The author is very grateful to numerous individuals (listed as the co-authors in the references) who contributed to the atlas development.

References

1987

[1] Lorensen W., Cline H.: Marching cubes: a high resolution 3-D surface construction algorithm. Computer Graphics, 21(4),163-169.

1994

[2] Nowinski W. L., Ho W. C.: Generalized transfer functions and their use for three-dimensional neuroimaging. Proc. SPIE Medical Imaging: Image Capture, Formatting, and Display, Newport Beach, CA, USA, SPIE Vol. 2164:34-45.

2000

[3] Kockro R. A., Serra L., Yeo T. T., Chan C., Sitoh Y. Y., Chua G. G., Ng H., Lee E., Lee Y. H., Nowinski W. L.: Planning and simulation of neurosurgery in a virtual reality environment. Neurosurgery, 46(1):118-137.

2007

[4] Teistler M., Breiman R. S., Sauw M. L., Liang Yoong Ho, Atif Shahab, Nowinski W. L.: Interactive definition of transfer functions in volume rendering based on image markers. International Journal of Computer Assisted Radiology and Surgery, 2(1):55-64.

2011

[5] Nowinski W. L., Chua B. C., Puspitasari F., Volkau I., Marchenko Y., Knopp M. V.: Three-dimensional reference and stereotactic atlas of human cerebrovasculature from 7 Tesla. Neuroimage, 55(3):986-98.

2012

[6] Nowinski W. L., Chua B. C., Qian G. Y., Nowinska N. G.: The human brain in 1700 pieces: design and development 142 of a three-dimensional, interactive and reference atlas. Journal of Neuroscience Methods, 15;204(1):44-60.

[7] Nowinski W. L., Chua B. C., Yang G. L., Qian G. Y.: Three-dimensional interactive human brain atlas of white matter tracts. Neuroinformatics, 10(1):33-55.

[8] Nowinski W. L., Johnson A., Chua B. C., Nowinska N. G.: Three-dimensional interactive and stereotactic atlas of cranial nerves and nuclei correlated with surface neuroanatomy, vasculature and magnetic resonance imaging. Journal of Neuroscience Methods, 206(2):205-216.

2013

[9] Nowinski W. L., Chua B. C., Johnson A., Qian G., Poh L. E., Wut Yi S. H., Aminah B., Nowinska N. G.: Three-dimensional interactive and stereotactic atlas of head muscles and glands correlated with cranial nerves and surface and sectional neuroanatomy. Journal of Neuroscience Methods, 215(1):12-18.

[10] Nowinski W. L., Chua B. C.: The Complete Human Brain (version 1.0 for iPad). Thieme, New York, AppStore.

2014

[11] Nowinski W. L., Chua B. C., Ngai V.: The 3D Brain Atlas (version 1.0 for Android tablet). Thieme, New York.

2015

[12] Nowinski W. L., Thaung T. S. L., Chua B. C., Wut Yi S. H., Yang Y., Urbanik A.: Three-dimensional stereotactic atlas of the extracranial vasculature correlated with the intracranial vasculature, cranial nerves, skull and muscles. The Neuroradiology Journal, 28(2):190-197.

[13] Nowinski W. L., Chua B. C., Thaung T. S. L., Wut Yi S. H.: The Human Brain, Head and Neck in 2953 Pieces. Thieme, New York.

[14] Nowinski W. L.: Brain atlasing: design principles, methods, tools and applications. In: Mityushev V., Ruzhansky M. (eds) Analytic Methods in Interdisciplinary Applications, Springer Proceedings in Mathematics & Statistics 116, Springer, Cham, pp. 97-107. doi:10.1007/978-3-319-12148-2_6.

Robust Method for the Text Line Detection and Splitting of Overlapping Text in the Latin Manuscripts

Jakub Leszek Pach, Piotr Bilski
[1]*Institute of Radioelectronics, Warsaw University of Technology, Poland*
{*jakub.pach,pbilski*}*@ire.pw.edu.pl*

Abstract. The paper presents the modified method of the text lines separation in the handwritten manuscripts. Such an approach is required for the medieval text analysis, where multiple text lines overlap and are written at different angles. The proposed approach consists in dividing the bounding boxes into smaller components based on the points of the character curves intersection. The method considers the askew text lines, producing non-rectangular zones between the neighboring lines.

Key words: document analysis, unconstrained handwriting, Hough transform, text line detection, connected component analysis, histogram analysis.

1. Introduction

The manuscript author identification is the complex and time consuming task, requiring multiple steps, which include the scanned document preprocessing, extracting the handwriting features and, based on them, making decision. The first stage is crucial for the task, as the classification efficiency strongly depends on the ability to identify features maximizing the separation of manuscripts belonging to different authors. Before the classification can be executed, multiple image processing operations (such as the text line detection or word separation) must be commenced. The difficulty of these tasks depends on the character of the manuscript, its origin or the time it was created. For instance, letters in the medieval Latin manuscripts are extremely dense, with multiple overlapping letters from the neighboring text lines. Also, the text lines may be straight (parallel to each other) or aslant at various angles. This abundance of problems allows for classifying the manuscripts regarding their difficulty for the image processing operations. Existing approaches for each stage are grouped according to this taxonomy. In most papers, simpler texts are processed, leaving open field for the analysis of the complex documents.

In the following paper, novel methods for two operations performed in the first stage of the image processing are presented. They include the text line detection and division of overlapping text blocks into separate sections. Both are important for the subsequent segmentation, i.e. extracting separate words from text fragments. Although advanced mathematical transformation methods are used for both tasks, the more complex documents require the new approach characterized by a higher accuracy than before. The

proposed algorithms are verified against selected Latin documents collected from Polish National Library.

The paper structure is as follows. In Section 2 the existing approaches are categorized and discussed. In section 3 the generic word segmentation system is presented, including subsequent stages. Section 4 introduces the novel algorithms implemented to solve the presented problems. In Section 5 results of the manuscripts' processing are presented. Section 6 contains conclusions and future prospects for the proposed approaches.

2. State of the art

To solve the considered tasks, i.e. the text line detection and the word fragments separation, various algorithms were used in the past. Characteristics of the most popular approaches are presented below to show advantages and drawbacks of the existing methods.

2.1. Text line detection

The text line detection is well established in the field of historical documents processing. In general, the developed algorithms fall into the following groups, which will be briefly explained below:

- Profile projection
- Hough transform-based
- Exploiting fuzzy run length.

The profile projection approach uses the histogram analysis to detect rows of densely grouped pixels [4]. Each row is one stripe of the histogram with the value proportional to the number of black pixels (composing words and letters). This way the pattern consisting of peaks or valleys is generated, where the former indicate the text lines. In [2] the initial image was additionally partitioned into vertical segments. For each segment, the horizontal histogram was calculated, leading to the higher line detection accuracy. Histogram-based methods have difficulties in detecting askew lines, therefore their modifications are required. It is assumed that the subsequent text lines in the original image are almost parallel to each other, which is not true in general.

Other approaches ([1, 9]) use the Hough transform which is able to locate aslant lines of text. The original pixel (in the Cartesian space) is represented as the trajectory in the polar space. The intersections of trajectories identify the most probable text lines. The probability of the correct line identification in the particular point is proportional to the number of intersecting trajectories in this location. The original approach was optimized to decrease the computational complexity [9]. Modifications are proposed to increase the accuracy of the line detection [14].

The fuzzy run length method exploits the effect of smudging the image along the horizontal axis [6]. This way the original words and sentences become "fuzzy" (with smudged contours proportional to the number of black pixels in the continuous block constructing the letter or word). The line detection is then executed locating three lines, i.e. the top and bottom border, and the middle between them, which identifies the actual text line. The approach is flexible, computationally saving and is able to detect leaning lines at the angle from the range of $\pm 45°$. It fails to detect curved (parabolic shaped) lines. The algorithm was tested on the official historical manuscripts.

Besides the groups of presented approaches, less popular solutions exist. For instance, the shortest spanning tree was used to analyze the Arabic text [7]. The clustering of CCs may also be useful, as was presented in [3]. In [8] the Adaptive Local Connectivity Map was used for this purpose. These methods are not yet fully explored, therefore their computational demands and detection accuracy should be optimized.

2.2. Extraction of overlapping text blocks from detected lines

The second stage is executed for all CCs, intersected by at least two lines. To extract the particular words (for instance, having the common part), the skeleton must be generated, i.e. the thin shape being the center of the monochrome area. As the original text is binarized, only black and white colors remain. Depending on which color is processed, the foreground or background skeleton is generated. As the result, character or background contours are constructed, respectively. The ideas behind these approaches are as follows [5, 12]:

- Foreground processing, i.e. working on the black pixels over the white background (where black color is represented by the value 0, while white is represented by 1). They are applied to the text with overlapping characters from different blocks and generate contours of characters.
- Background processing, i.e. working on the white pixels from the binary text image. The background skeleton of empty spaces surrounding characters is generated here.
- Recognition of single characters, separating them from the text line.

Existing approaches fail in distinguishing between overlapping characters in the aslant text lines (where letters overlap under different angles), which is the typical in Latin manuscripts. The method proposed in the paper solves the presented problems ensuring higher accuracies.

3. Text processing architecture

This section presents the generic architecture for the image processing during the word separation procedure. Operations executed here prepare the separate words or their fragments to the features extraction (such as the letter thickness or convexity factor),

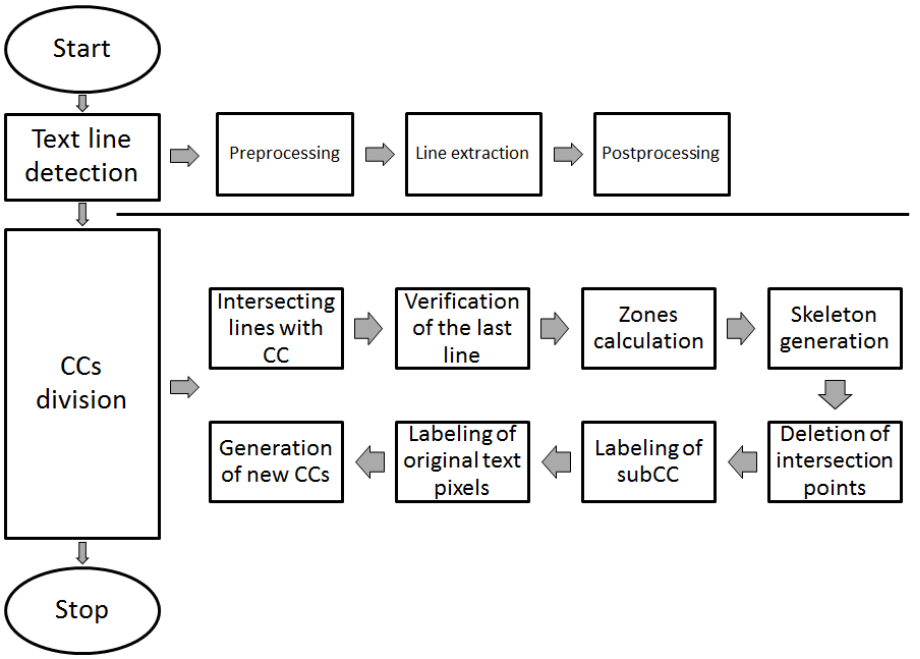


Fig. 1. The generic architecture of the word segmentation system

which further will be used to identify the author. In Fig. 1 two main operations (i.e. the text line detection and division of CCs into separate fragments) are presented. Each contains some smaller steps, executed sequentially. The text line detection consists in extracting fragments of the binarized image, which belong to the same line. This is the first step to extract dense fragments of the text, further used to identify the author. This operation was considered previously, therefore it will be presented shortly, while the CCs division requires more attention. Its main steps are extracting zones (areas belonging to the particular text line), creating skeletons of the writing curves and dividing CCs containing the text from two intersecting lines into separate areas (subCCs). In each subCC, pixels of the handwriting are added to the nearest skeleton and labeled as the dense fragment. Details of these operations are in Section 4.2.

The fundamental structure for the text processing is the Connected Component (CC). It is the dense set of the pixels marked by ink (handwriting) which borders are defined by the bounding box. Before CCs can be extracted, a list of operations must be performed in the preprocessing stage. Every document is first scaled down to the minimum acceptable resolution. The historical manuscripts are usually scanned with very high

resolutions, exceeding the algorithm's requirements. To shorten the processing duration, their resolution is decreased. Next, the binarization is performed to obtain only black and white image, where the white color represents the text, while the black is the background. The CCs are then identified and separated into three groups:

- S_1, containing the "short" characters, spanning over the single text line. They meet the following requirement:

$$cc_i \in S1 \Leftrightarrow \left\{ \left(\frac{1}{2} \cdot \bar{h} < h_i < 3 \cdot \bar{h} \right) \wedge \left(w_i > \frac{3}{2} \cdot \bar{w} \right) \right\} , \tag{1}$$

  where h_i is the height of the i-th CC, and w_i its width, and $\bar{h}$ and $\bar{w}$ are their mean values, respectively.

- S_2, containing all characters spanning over multiple text lines. This includes capital letters and the ones with long tails, overlapping the neighboring text lines:

$$cc_i \in S2 \Leftrightarrow \left(h_i >= 3 \cdot \bar{h} \right) . \tag{2}$$

- S_3, containing remaining elements of the text, such as accents, dots, etc.:

$$cc_i \in S3 \Leftrightarrow \left(\frac{1}{2} \cdot \bar{h} > h_i \right) . \tag{3}$$

The extracted CCs are processed by the Hough transform to find candidates for the text lines. Every CC is divided into the selected number of blocks. In each block the central point is calculated with the coordinates of the ink pixel being the closest one to the geometric center of the image fragment. For each point the accumulator array is generated, illustrating the position of the particular text lines intersecting the subsequent central pixels. The text line is detected if pixels from at least nine blocks intersect (which means the same straight line goes through all of them). From the remaining elements of the accumulator array the additional text lines are identified if they are parallel to the ones detected previously. In the postprocessing stage the redundant and "false" lines are deleted and the ones constructed by CCs not being the part of any line are added.

Separation of the intersecting characters from the neighboring lines requires the preceding extraction of skeletons of the handwriting (i.e. the thin shapes – one-pixel broad – of the written characters), which are the core elements of the text. Their identification allows for division of CCs containing shapes from neighboring text lines (vertically) into smaller fragments, each containing the text from one line only. The problem is the assignment of the original text pixels to the particular skeletons, especially in the case of intersecting fragments.

4. Description of the implemented algorithms

To correctly separate fragments of the text into single words or phrases (consisting of continuous curves created by the ink) two operations are performed, presented below in details.

4.1. Text line detection

This step consists in the standard sequence of operations, as explained above. The proposed method was already implemented in the previous research [14]. Its main idea is the introduction of the supporting CCs to increase the number of detected lines. After the binarization using the Otsu algorithm and generation of three groups of CCs, only the group S_1 is considered for the text lines detection. We introduced the supporting CCs, which are built based on histograms calculated horizontally on the vertical fragments of the text. They were added to the set S_1 and processed similarly as already existing CCs.

Calculating the Hough transform for the extended set of CCs allowed for increasing the number of correctly detected text lines. Finally, the incorrectly detected and repeating lines are eliminated to present the unique set.

4.2. Text fragmentation

After detecting text lines, it is possible to have CCs intersected by multiple (at least two) lines. The following procedure is proposed by us to separate such a complex CC into separate ones (along the y axis) if it contains multiple text lines.

1. Zones Z_i are created as the areas between the neighboring lines (the i-th zone is created between the lines i and i+1). In contrast to [13], where all lines were constructed as parallel to each other and to the bounding box, we assume the angle between the neighboring lines may be between -5° and $+5^\circ$ (while the angle between the parallel lines is 0°). Therefore for each zone candidate it is checked if the line is parallel or not. In the first case, the zone is defined as the rectangle limited by the intersection points between the neighboring lines and boundaries of the CC [9] (Fig. 2). If the lines are not parallel, the proposed procedure is more computationally complex. It constructs the zone area by determining the top and the bottom pixel of the zone for each column of the image (each point in the x dimension). The proposed approach is more accurate than existing ones [9]; the lines are considered non-parallel even if the difference between the vertical positions of extreme line points belonging to the CC is 1. Two main cases are considered here. The actual text lines may form the parallelogram or the trapezoid (Fig. 3). The result of both effects may be the incorrect assignment of the handwriting pixels to the particular skeleton, which is illustrated in Fig. 4. Here the problem may be separating text pixels from the second and the third line. The letters "e" from the former and "l" (the fragment indicated by the black circle) from the latter should be

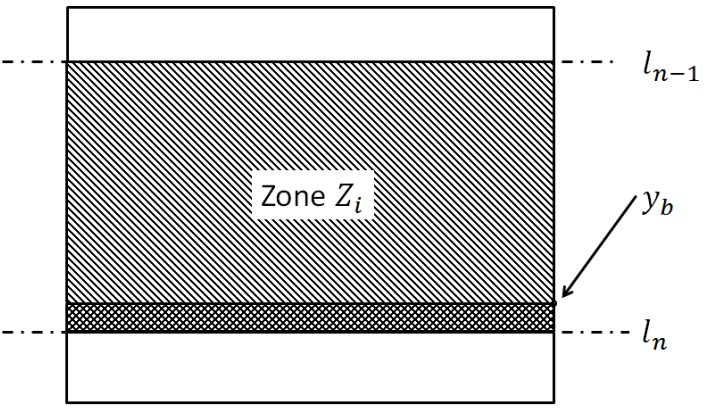


Fig. 2. The zone Z_i with the 10% of the area above the text line, where the condition (4) is met.

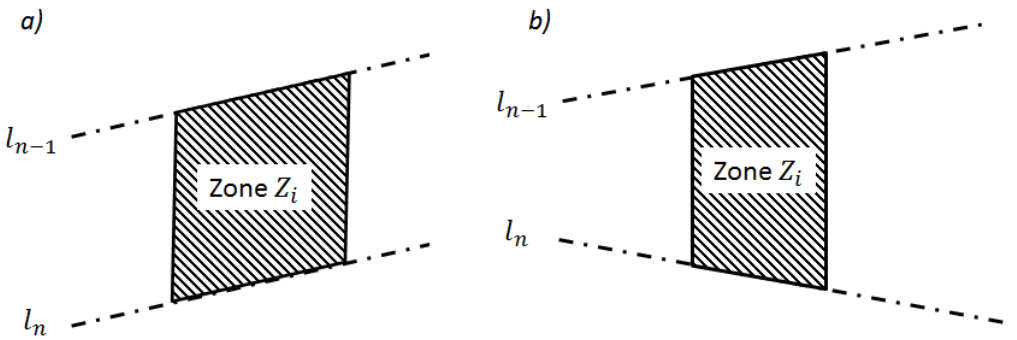


Fig. 3. Examples of the non-rectangular zones, created by the neighboring text lines: parallelogram (a) and trapezoid (b).

separated. The method from [13] will cut the text according to the thick dotted lines, while our approach will be more precise, separating both letters along the aslant grayed lines.

2. In each CC it is checked if the last line from the bottom is indeed the text line. In the Latin manuscript some letters (such as "j", "y" or "f") may be written with long strokes. In such a case, the multiple set of strokes at the same vertical level may be detected by the previously presented procedure as the separate text line. Normally such long fragments of letters overlap with the characters from the lower text line, but in the sequence of words (after which there is the significant empty vertical space), they

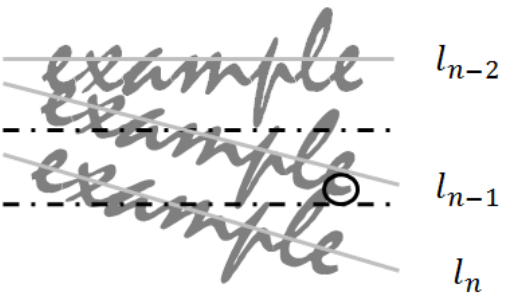


Fig. 4. Example of the neighboring text lines non-parallel to each other and the subsequent assignment of pixels of the intersecting characters to particular skeletons.

can be detected as the separate line. To avoid this false detection, 10% of the last zone (between the preceding and the bottom line) is processed to determine if the fraction of the black points (ink traces) here is at least 8% of the whole area:

$$0.08 < \frac{\lfloor Z_i \rfloor_{0.1}}{Z_i} = \frac{\sum_{x_b y_b}^{x_{n-1} y_{n-1}} I_{x,y}}{\sum_{x_s y_s}^{x_n y_n} I_{x,y}}\,, \tag{4}$$

$$y_b = (y_s - y_n) \cdot 10\,. \tag{5}$$

The last lines in the CC not meeting this criterion are eliminated as they do not construct the actual text.

3. Skeletons of the characters are created in each CC. This is the method of deciding, which parts of the overlapping curves belong to the particular text line. The skeleton is constructed as the one-pixel width shape going through the center of the character curve (which, if the document was scanned with the resolution high enough, will be much thicker). After this operation, the character shapes with minimum thickness are obtained. Next, the points of intersection between the characters belonging to different text lines are found and eliminated. The intersection point contains the ink pixel in the center and at least three neighboring black pixels. Finally, separated curves are assigned to the neighboring text lines. The fragments laying on the text lines are just assigned to them. For the fragments not laying on the text line, the subCC (rectangles constructed on the boundaries of the curve) are created. They are assigned to the text line being the closest one (in the Euclidean sense) to them. If there is no intersection in the CC, the zone Z_i is divided into two halves and black pixels from both areas are assigned to the corresponding text lines. Each skeleton point belonging to the same text line is labeled with this line's identifier.

4. The original character pixels are labeled according to the smallest distance from

the skeleton pixels. As the result, the whole text is assigned to the particular text line identifiers, illustrated in the following pictures with the subsequent colors.

5. Experimental results

The results of processing the document [15] (containing hundreds of pages) with our approach are shown in Fig. 5. Two pages of 25 selected for experiments are presented here. On the left side, the original image is presented. The right side contains results of computations. Each detected line with the text belonging to it is indicated by one of three colors: blue, red and green (used interchangeably to show, which elements of the text were assigned to which line). The black color indicates the text fragments not assigned to any line, which is not considered error, but used as a fail safe. If the bounding box contains the character too far from the text line, it is not assigned to it, although might be, if additional requirements (introduced during the future research) are met. In the ideal case, the whole line should be identified by the separate color. As can be seen, some fragments of the the lines are described by different colors, which indicates the error of identification (for instance, the left part of the third and the fourth line from the top of the first page in Fig. 5). This illustrates the complexity of the identification problem, especially if the text lines are not parallel (in the first page the text forms the parabolic shapes instead of the straight lines).

In the presented case, the proposed methodology gives acceptable results, as the line detection accuracy is above 90%. The segmentation accuracy for the correctly detected lines is difficult to assess at this stage - the appropriate assessment method will be proposed in the next paper. Some problems with the text segmentation are caused by the previous step, i.e. the text line detection, which should be improved in the future. For instance, the non-adaptive Otsu algorithm should be replaced with a more robust approach. Also, is was observed that the size of the S_1 CC set influences the lines identification and text segmentation accuracy.

6. Conclusions

The method presented in the paper ensures the more accurate division of words in the manuscript than the approaches used so far. The well established algorithm [9] assumes the text lines are in the horizontal position with small parts of the overlapping text. The presented problem is more difficult, as virtually all lines are interconnected with each other. This makes generation of the CCs a complex process, as many blocks will be intersected by the multiple lines. To avoid errors in the subsequent handwriting processing stages, the number of CCs assigned to the correct text lines should be maximized. The proposed approach is useful in dividing the bounding box into smaller counterparts even if the character curves intensely overlap. The proposed trapezoidal shape of the

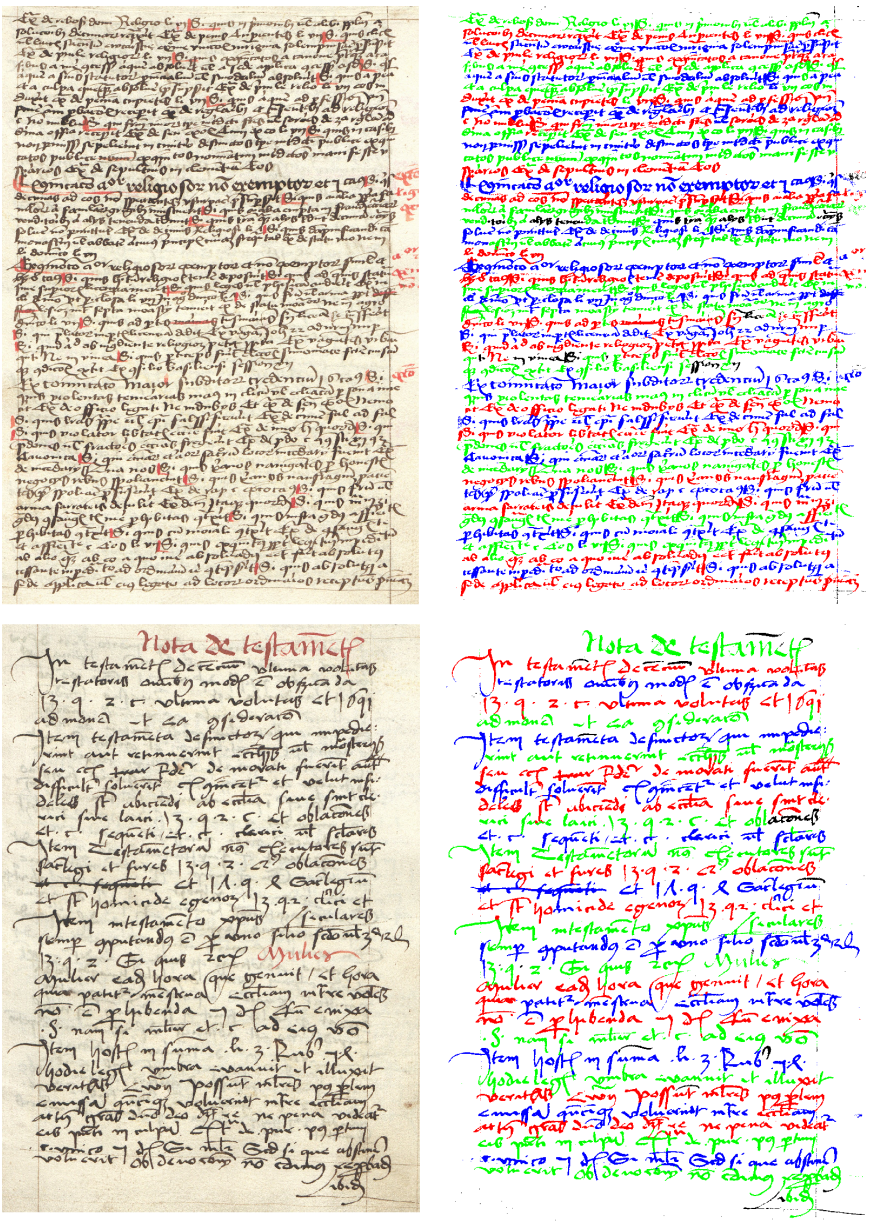


Fig. 5. Illustration of the performance of the proposed method. The left side presents the original text, while the right side contains the results of the processing.

bounding box produces more accurate zone areas. This allows for the better assignment of the particular CC fragments to the corresponding text lines. Such small differences between the obtained zones may have impact on the next stage, i.e. the handwriting features extraction. In shorter CCs the chance of having parallel lines is higher than that for the long ones.

References

1988

[1] L. A. Fletcher and R. Kasturi : A robust algorithm for text string separation from mixed text/graphics images. In IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 10, no. 6, pp. 910-918.

1995

[2] L. Likforman-Sulem and A. Hanimyan and C. Faure : A Hough based algorithm for extracting text lines in handwritten documents. In Proceedings of the Third International Conference on Document Analysis and Recognition, vol. 2, pp. 774-771.

[3] I. S. I. Abuhaiba, S. Datta, and M. J. J. Holt : Line extraction and stroke ordering of text pages. In the Third International Conference on Document Analysis and Recognition, vol. 1, pp. 390-393.

1999

[4] E. Bruzzone and M. C. Coffetti : An algorithm for extracting cursive text lines. In Proceedings of the Fifth International Conference on Document Analysis and Recognition ICDAR '99, pp. 749-752.

2000

[5] Yi-Kai Chen and Jhing-Fa Wang : Segmentation of single- or multiple-touching handwritten numeral string using background and foreground analysis. In IEEE Transactions on Pattern Analysis and Machine Intelligence 2000, pp. 1304-1317.

2004

[6] Z. Shi and V. Govindaraju : Line separation for complex document images using fuzzy runlength. In First International Workshop on Document Image Analysis for Libraries, pp. 306-312.

[7] S. Nicolas, T. Paquet, and L. Heutte : Text line segmentation in handwritten document using a production system. In IWFHR-9 2004. Ninth International Workshop on Frontiers in Handwriting Recognition, pp. 245-250.

2005

[8] Z. Shi, S. Setlur, and V. Govindaraju : Text extraction from gray scale historical document images using adaptive local connectivity map. In Eighth International Conference on Document Analysis and Recognition, vol. 2, pp. 794-798.

2006

[9] G. Louloudis, B. Gatos, I. Pratikakis, and K. Halatsis : A block-based Hough transform mapping for text line detection in handwritten documents. In Tenth International Workshop on Frontiers in Handwriting Recognition.

2007

[10] M. Arivazhagan, H. Srinivasan, and S. Srihari: A statistical approach to line segmentation in handwritten documents. In Proc. SPIE 6500, Document Recognition and Retrieval XIV, 65000T.

2011

[11] A. Alaei, P. Nagabhushan, and U. Pal: A new text-line alignment approach based on piece-wise painting algorithm for handwritten documents. In International Conference on Document Analysis and Recognition (ICDAR), pp. 324-328.

2014

[12] Chamchong, R. and Chun Che Fung: A combined method of segmentation for connected handwritten on palm leaf manuscripts. In 2014 IEEE International Conference on Systems, Man and Cybernetics (SMC), pp. 4158-4161.

[13] Gatos, B. and Louloudis, G. and Stamatopoulos, N.: Segmentation of Historical Handwritten Documents into Text Zones and Text Lines. In 14th International Conference on Frontiers in Handwriting Recognition (ICFHR 2014), pp. 464-469.

2015

[14] J. L. Pach, P. Bilski: A Robust Text Line Detection in Complex Handwritten Documents. In 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications. (IDAACs 2015), 24-26 Sept. 2015, Warsaw, Poland, pp. 271-275.

[15] Miscellanea theologica 2015. [Online]. Available: `http://polona.pl/item/12909419/0/`.

EBMBDT: Effective Block Matching Based Denoising Technique using Dual Tree Complex Wavelet Transform

M. Selvi
Aloy Labs, I-901, Springfields Apts, Sarjapur Road, Bellandur Gate, Bangalore – 560102, India

Abstract. In processing and investigation of digital image denoising of images is hence very important. In this paper, we propose a Hybrid de-noising technique by using Dual Tree Complex Wavelet Transform (DTCWT) and Block Matching Algorithm (BMA). DTCWT and BMA is a method to identify the noisy pixel information and remove the noise in the image. The noisy image is given as input at first. Then, bring together the comparable image blocks into the load. Afterwards Complex Wavelet Transform (CWT) is applied to each block in the group. The analytic filters are made use of by CWT, i.e. their real and imaginary parts from the Hilbert Transform (HT) pair, defending magnitude-phase representation, shift invariance, and no aliasing. After that, adaptive thresholding is applied to enhance the image in which the denoising result is visually far superior. The proposed method has been compared with our previous de-noising technique with Gaussian and salt-pepper noise. From the results, we can conclude that the proposed de-noising technique have shown better values in the performance analysis.

Key words: image denoising, Complex Wavelet Transform (CWT), dual tree CWT, block matching, soft thresholding.

1. Introduction

For image processing researchers for a lengthy period, Image denoising is a dynamic area of interest. Image denoising plans at lessening the noise in homogeneous areas whereas saving the image contours. For post processing techniques like segmentation, classification, object recognition, pattern analysis, registration, etc, Image denoising is significant [19]. The noise of image is not generally effortlessly eradicated in image processing. According to image characteristic, Noise statistical property and frequency spectrum allocation rule really computed, there are many researchers improved several techniques of eliminating noises that roughly are separated into space and transformation fields. The space field is information operation brought on the initial image, and processes the image gray value, like neighborhood average technique, Total Variation (TV) filter, ROF filter, wiener filter and so on [16]. Several dissimilar cases of alterations are there. One of the most common cases is alteration owing to preservative white Gaussian noise which can be caused by poor image achievement or by transmitting the image data in noisy communication channels. Other kinds of noises consist of impulse and speckle noises. A denoising algorithm has to acclimatize to image discontinuities in order to accomplish a good presentation in this reverence. For image denoising, Wavelet

transforms have been applied successfully supplying a means to utilize the relationships among coefficients at numerous scales [11].

To take away the noise while holding as much as feasible the significant signal features is the goal of denoising. Image denoising still stays a challenge for researchers since noise elimination launches artifacts and causes blurring of the images. To carry out denoising of images, numerous techniques are being improved [14]. Two fundamental strategies are there to image denoising, spatial filtering techniques and change domain filtering techniques. Spatial filters use a low pass filtering on groups of pixels with the statement that the noise absorbs the higher area of frequency spectrum. Spatial Low-pass filters will not just level away noise however moreover blur edges in signals and images while the high-pass filters can compose edges still sharper and develop the spatial resolution however will furthermore enlarge the noisy background [1]. Linear filters, which contain convolving the image with a steady matrix to attain a linear mixture of neighborhood values, have been extensively applied for noise removal in the existence of additive noise [15].

From the computational point of outlook, the discrete transform is incredibly competent. Wavelets provide a better presentation in image denoising owing to properties such as sparsity and multi resolution structure. It engages three steps: a linear forward wavelet transform, nonlinear thresholding step and a linear inverse wavelet transform. Wavelet thresholding (first suggested by Donoho [4]) is a signal evaluation method that utilizes the capabilities of wavelet transform for signal denoising. By killing coefficients, it takes away noise that are unimportant relative to some threshold, and turns out to be easy and efficient, depends heavily on the selection of a thresholding parameter and the selection of this threshold concludes, to a huge degree the effectiveness of denoising. Directional wavelet transforms [3]; steerable pyramids [2], complex wavelets [5], curvelets [6] and contourlets [9] are some examples where all of them are unneeded. Although the Discrete Wavelet Transform (DWT) is a dominant image and signal-processing device, it has three drawbacks that destabilize its usage in several applications.

Initially, it is shift responsive since input-signal shifts produce unpredictable alterations in DWT coefficients. Next, the DWT endures from poor directionality since DWT coefficients expose only three spatial points of references. Third, DWT study of real signals needs the phase data that precisely explains non-stationary signal performance. The Dual-Tree Wavelet Transform (DTWT) was produced by Kingsbury [5], [8] to overcome these problems, which is a redundant, complex wavelet transform with outstanding directionality, diminished shift sensitivity and clear phase information. The DTWT gives up superb results in applications where idleness is tolerable because of these benefits [8], [10]. The DTWT is outmoded since it contains a pair of filter banks that concurrently work on the input signal and offer two wavelet disintegrations. The wavelets related with the filter banks are a Hilbert pair. This asset is significant because it offers the benefits of reduced shift sensitivity, enhanced directionality and clear phase

information. Conversely, the aim of Kingsbury's DTWT filters is difficult since it needs an iterative optimization over the space of perfect-reconstruction filter banks.

The decisively modeled Discrete Wavelet Transform (DWT) has been effectively used to an extensive range of image denoising tasks. Though, because of the subsequent problems, its presentation is restricted [7]. The double-tree CWT is an important improvement of the traditional actual wavelet transform that is almost shifting invariant and, in elevated dimensions, directionally selective. As the actual and imaginary parts of the dual-tree CWT are, in truth, conservative real wavelet changes the CWT advantages from the vast theoretical, practical, and computational resources that have been progressed for the standard DWT [7], [12], [13].

Several works are available in the literature for the image de-noising using medical and other images. However, in the neighborhood of discontinuities, linear filtering such as the Gaussian filter removes noise but blurs edges significantly. This undesirable effect can be reduced by considering local geometries and statistics during the filtering process. In this work, we proposed an efficient denoising technique using DTCWT based on block matching algorithm. Here, block matching process is used in order to overcome the smoothing filter type and it will not affect the lower dimensions. Then DTCWT and adaptive thresholding are used for the image denoising and image enhancement phase. Finally, PSNR and SDME are used for the evaluation matrices. The rest of the paper is organized as follows: a brief review of some of the literature works in denoising technique is presented in Section 2. The proposed image denoising technique is detailed in Section 3. The experimental results and performance evaluation discussion is provided in Section 4. Finally, the conclusions are summed up in Section 5.

2. Related Works

For image denoising, a handful of researches have been offered in the literature. Lately, applying Dual Tree Complex Wavelet Transform (DTCWT) for image denoising has obtained a huge deal of awareness among researchers. A short assessment of some new researches is offered in Tab. 1.

3. Image Denoising Using Complex Wavelet Transform and Block Estimation

In the area of image processing, Image Denoising has continued a basic problem. Due to numerous inevitable reasons, it is not unusual that images are polluted by noise. When applying Complex Wavelet Transform (CWT), emphasized that matter such as selection of primary resolution (the scale level at which to begin thresholding) and selection of analyzing wavelet moreover have a great pressure on the victory of the shrinkage process. In such a wavelet transform, a great magnitude of a coefficient implies the existence of a singularity while the phase indicates its location inside the support of the wavelet.

Tab. 1. Set of selected researches and their works.

Authors	**Year**	**Algorithm**	**Purpose**	**Difference of our work**
Tamanna Howlader and Yogendra P. Chaubey [17]	2010	Complex wavelet Transform	Denoising of cDNA microarray images	Works well only on micro array images.
Benjamin Huhle et al. [18]	2010	Non Local Means	Denoising and resolution enhancement	Computationally expensive.
Rodrigo Moreno et al. [21]	2011	Tensor voting frame work	Denoising and surface reconstruction	It will remove the noise only on edge regions
Chen et al. [22]	2012	Dual tree complex wavelets	Image denoising	
Xiang-Yang Wang et al. [26]	2014	Extended discrete shear let transform, Hidden Markov Tree	Edge preserving denoising scheme	Works well only on localvar noise
Lakshmi Srinivasan et al. [25]	2014	Complex Gaussian scale mixture (CGSM) model, complex wavelet transform	Two channel micro array images denoising	Works well only on micro array images.
Norbert Remenyi et al. [24]	2014	Complex wavelet transform, bayes estimation	Image denoising	It will works only in diagonal shrinkage method.
Amel Baha Houda et al. [23]	2015	Dual tree complex wavelets	Medical image denoising	They are concentrated only on Blocky images
Ayushi Jaiswal et al. [27]	2014	Discrete Wavelet Transform	Image denoising	The proposed method works only in salt and pepper noise and Gaussian noise.

The complex wavelet transform (CWT) uses analytic or quadrature wavelets promising magnitude- phase illustration, shift invariance and no aliasing. A competent way of suggested denoising method is explained by Fig. 1.

Let us regard the noisy image $z : X \to R$ of the form, i.e.,

$$z(x) = y(x) + \eta(x) \tag{1}$$

Let us introduce the inspection model and memo applied right through the paper. We consider noisy observations $z : X \to R$ of the form $z(x) = y(x) + \eta(x)$, where $x \in X$ is 2D spatial coordinating that belongs to the image domain $X \subset Z^2$, y is the true image, and $\eta(x) \approx \aleph(0, \sigma^2)$, is white Gaussian noise of variance σ^2. By Z_x we indicate a block of fixed size $M \times N$ removed from z, which has $z(x)$ as its upper-left element; instead, we say that Z_x is situated at x. With $\hat{y}$ we assign the last estimate of the true image.

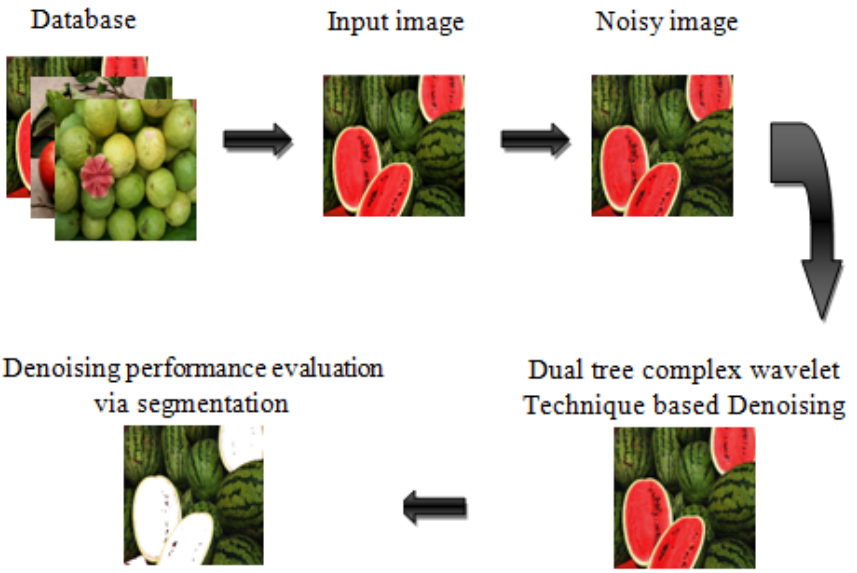


Fig. 1. Block diagram of the Proposed Method.

Let us utter the applied statements. We take for granted that some of the blocks (of fixed size $M \times N$) of the true image show mutual relationship. We moreover assume that the chosen unitary transform is able to signify sparsely these blocks. Let z be an image of dimension $M \times N$. Partition the image z into number of $M \times N$ non overlapping blocks. This can be signified as

$$z = \{zb_1, zb_2, ..., zb_{Nb}\} \tag{2}$$

where Nb signifies the total number of blocks in the image. Next, the related block from the image is recognized by applying the block matching process.

3.1. Block Matching

The property of self-similarity of objects is applied by block matching procedure in image and video compression. A few of the blocks acquired by dividing the image into numerous blocks are alike. Hence, the idea of block matching is applied to stop executing repetitive denoising on the similar block. Using block matching, related blocks in a specified input image are recognized i.e., the matched blocks for every indication block in an image. To compute the resemblance between the images, the Euclidean distance measure is applied.

Our suggested technique [20], computes the similarity of the a^{th} block for the block matching procedure by comparing the distance measure of the a^{th} block and its n neighboring blocks. Equation 3 is applied to compute the distance measure,

$$S_d = \sqrt{\sum (zb_a - zb_b)^2} \tag{3}$$

here Ib_a and Ib_b, where $b = \{1, 2, ..., n\}$, signify the current block and blocks next to the current block, correspondingly. This is illustrated in the Fig. 2.

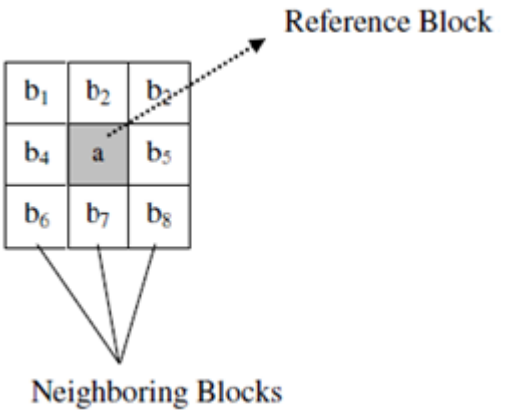


Fig. 2. Reference blocks and adjacent blocks.

The flag value is placed according to the threshold D_tsh, following the distance measure is computed. The computed distance S_d is compared with the threshold D_tsh as follows,

$$I_F^b = \begin{cases} I_F^b = 1; & if S_d < D_{tsh} \\ I_F^b = 0; & otherwise \end{cases} \tag{4}$$

Here b and F signify the image block and the flag value of every block of the image, correspondingly. Both a^{th} and b^{th} blocks are said to be alike if the b^{th} block yields a flag value of 1, when it is compared with the a^{th} block. Or else they are said to be different. This is shown in Fig. 3.

Hence we accumulate the indices by recognizing the blocks related to the a^{th} block. In fractal image compression, a^{th} block is the array block and the analogous similar blocks are domain blocks. As a substitute of all the similar domain blocks, we apply only the sort block once the indices of range block and its related domain blocks are

1	1	0
1	a	0
1	0	0

Fig. 3. Flags assigned to each domain blocks.

gathered. The time and memory difficulty is reduced by this. After joining the similar blocks into a heap, CWT is used to the domain blocks in each of the group.

3.2. Complex Wavelet Transform (CWT)

The image with the initial guessing pixel values is given to 2D dual tree M band wavelet transform, which offers local, multi-scale and directional analysis. The M- band multi-resolution analysis of $L^2(R)$ with $M \geq 2$ is defined by one scaling function (or father wavelet) $\psi \in L^2(R)$ and $M-1$ mother wavelets $\psi_m \in L^2(R), m \in \{1, ..., M-1\}$. These functions are the solution of the following scaling equation:

$$\frac{1}{M^{\frac{1}{2}}} \psi\, m\left(\frac{s}{\sqrt{M^2}}\right) = \sum_{k=-\infty}^{\infty} f_m[g]\psi_0(s-g) \tag{5}$$

where, $(f_m\,[g])_{g\in z}$ are the square summable sequences. In the following, we will assume that these functions (and thus the associated sequences $(f_m\,[g])_{g\in z}$ are real valued. The Fourier transform of $(f_m\,[g])_{g\in z}$ is a 2π periodic function, represented by H_m for the set of functions $\left(\bigcup_{m=1}^{M-1}\{M^{-j/2}\,\psi_m(M^{-j}s-g), (j,g)\in Z^2\}\right)$ to correspond to an orthonormal basis of $L^2(R)$, and the following paraunitary conditions should hold:

$$\sum_{p=0}^{M-1} H_m \frac{(wM+pM+2\pi)}{M} H^*_{m'} \frac{(wM+pM+2\pi)}{M} = M\lambda_{m-m'} \tag{6}$$

where, $\lambda_m = 1$ if $m = 0$ and 0 otherwise. Our aim is to build a "dual" M-band multi resolution analysis defined by scaling function ψ_0^H and mother wavelets $\psi_m^H, (m \in \{1, 2, ..., M-1\}$ More accurately, the mother wavelets will be obtained by means of a Hilbert transform from the "original" wavelets $\psi_m, m \in \{1, 2, ..., M-1\}$. In the Fourier domain, the desired property reads

$$\forall m \in \{1, ..., M-1\},\ |\hat{\psi}_m^H(w)| = -i\ sign(w)\,\hat{\psi}(w) \tag{7}$$

where, $(f_m\,[g])_{g\in Z}$ sign is the signum function, and $\hat{d}$ is the fourier transform of the function d. Moreover, the functions $\hat{\psi}_m^H$ are defined by scaling equations similar to (5)

involving real - valued sequences $(f_m[g])_{g\in Z}$. In order to create dual M-band orthonormal wavelet basis of $L^2(R)$, the Fourier transform G_m of the sequences $(f_m[g])_{g\in Z}$ should satisfy the paraunitary conditions. The Hilbert condition 8 yields

$$\forall m \in \{1, ..., M-1\} \qquad |\hat{\psi}_m^H(w)| = |\hat{\psi}_m(w)| \tag{8}$$

The scaling equation leads to

$$\forall m \in \{1, ..., M-1\} \qquad G_m(w) = e^{-i\theta m(w)} H_m(w) \tag{9}$$

where θ_m is 2π periodic. The frequency phase functions should also be odd (for real filters) and thus only need to be determined over $[0, \pi]$. In the 2-band case (under weak assumptions), θ_m is a linear function on $[-\pi, \pi]$. In the M band case, the constraint is slightly restricted on a smaller interval by imposing $\forall w \in [0, 2\pi/M],\ \theta_0(w) = \zeta w$, where $\zeta \in R$.

It can be deduced that, paraunitary M band filter bank conditions are obtained by selecting the phase functions defined by eqn 10,

$$\forall p \in \{0, ..., \left(\frac{M}{2}\right) - 1\}, \forall w \in \left[\frac{pM + 2\pi}{M}, (p+1)\frac{2\pi}{M}\right], \theta_0 = \left(d + \frac{1}{2}\right)(M-1)w - p\pi, \tag{10}$$

$$\forall m \in \{1, ..., M-1\}, \theta_m(w) \in \begin{cases} \frac{\pi}{2} - (d + \frac{1}{2})w & \text{if } w \in (0, 2\pi], \\ 0 & \text{if } w = 0, \end{cases} \tag{11}$$

where $d \in Z$ denotes the upper integer part of real u. The scaling function related to the dual wavelet composition is such that

$$\forall k \in N, \forall w \in [2k\pi, 2(k+1)\pi, \hat{\psi}_0^H(w) = (-1)^k e^{-1\left(d+\frac{1}{2}\right)w} \psi_0(w) \tag{12}$$

It should also be noted that except for the 2 band case, θ_0 exhibits discontinuities on $0, \pi$ due to the $p\pi$ term.

The 2D separable M-band wavelet bases are derived from the 1D dual tree decomposition. Thus, we obtained two bases of $L^2(R^2)$. The first one corresponds to the conventional 2D separable wavelet basis, but the second one results from the tensor product of the dual wavelet basis function. A discrete implementation of these wavelet decomposition starts from the level $j = 1$ to the coarsest resolution level $j \in N^*$. The decomposition on the former 2D wavelet basis function yields coefficients $\delta_{j,m,m'}[k, l]$, whereas the decomposition on the dual basis generates coefficients $\delta_{j,m,m'}[k, l]$.

3.3. Soft Thresholding

After using CWT to the domain blocks of the group, soft thresholding method is employed to denoise the blocks. In this suggested method, Normal Shrink is applied as soft

thresholding to denoise the noise blocks. The suggested technique executes soft thresholding with the data driven subband dependent threshold T_N which can be specified as,

$$T_N = \frac{\beta\hat{\sigma}^2}{\hat{\sigma}_y} \tag{13}$$

where,

β is the scale parameter,
$\hat{\sigma}^2$ is the noise variance,
$\hat{\sigma}_y$ is the standard deviation of the subband under consideration computed, by using the standard equation.

The scale parameter β is computed once for each scale using the following equation

$$\beta = \sqrt{\log\left(\frac{L_k}{J}\right)} \tag{14}$$

where,

L_k is the length of the sub band at k^{th} scale.

The noise variance $\hat{\sigma}^2$ can be calculated is from the subband HH1, using the formula

$$\hat{\sigma}^2 = \left[\frac{\text{median}\left(|Y_i j|\right)^{-2}}{0,6745}\right], Y_i j \in \text{sub band HH1} \tag{15}$$

After applying the soft thresholding to the noisy blocks, inverse CWT must be applied to rearrange the blocks.

3.4. Inverse CWT

Using the opposite complex wavelet transform, the resulting fused image is next generated by changing the united coefficient map. The wavelet coefficient images demonstrate the orientated nature of the complex wavelet subbands. Consequently the denoised image attained after the procedure of using converse complex wavelet transform.

4. Results and Discussion

In this section, we illustrate the effectiveness of the proposed scheme in image denoising by means of the results obtained from the experimentation. The proposed method was implemented in MATLAB (Matlab 7.13) and the proposed hybrid coding scheme was evaluated using both grayscale and color images. The test images used in the experiments

include: Lena, Barbara, Baboon, Peppers, Balloon, and Apple etc. The quality of the denoised images was determined by measuring the PSNR values and SDME values. The sample output obtained from the proposed method is presented in Tab 2.

In the figures 4, 5 and 6, the image is segmented based on red color. The efficiency of segmentation of denoised image is demonstrated beneath. The precision of the segmentation is more as compared to the segmentation ok noisy image. From the figures 4, 5 and 6, we can illustrate the significance of image denoising for image segmentation.

At this point we are applying three methods to compare the segmentation presentation of the image. These are

1. Rand Index,
2. Global Consistency Error,
3. Variation Index.

If the Rand Index is elevated, the segmentation presentation is excellent. The Global Consistency Error must be little to get improved segmentation result. The value of Variation Index must be fewer to generate improved segmentation. Tab. 3 and Fig. 7 demonstrates the efficiency of segmentation of noised and denoised image.

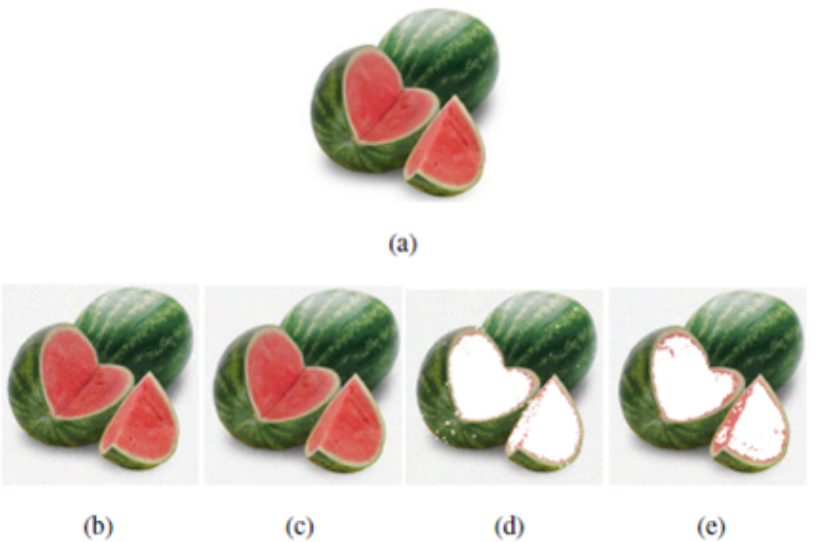
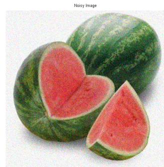
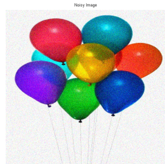
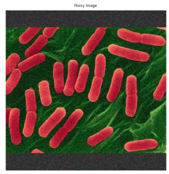
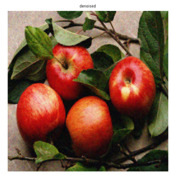
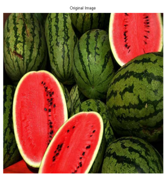
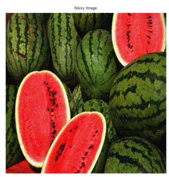


Fig. 4. Importance of Denoising (a) original image (b) noisy image (c) denoised image (d) noisy image segmentation (e) denoised image segmentation.

Tab. 2. Denoised image output.

Original image	**Noisy image**	**Denoised image**
		
		
		
		
		
		

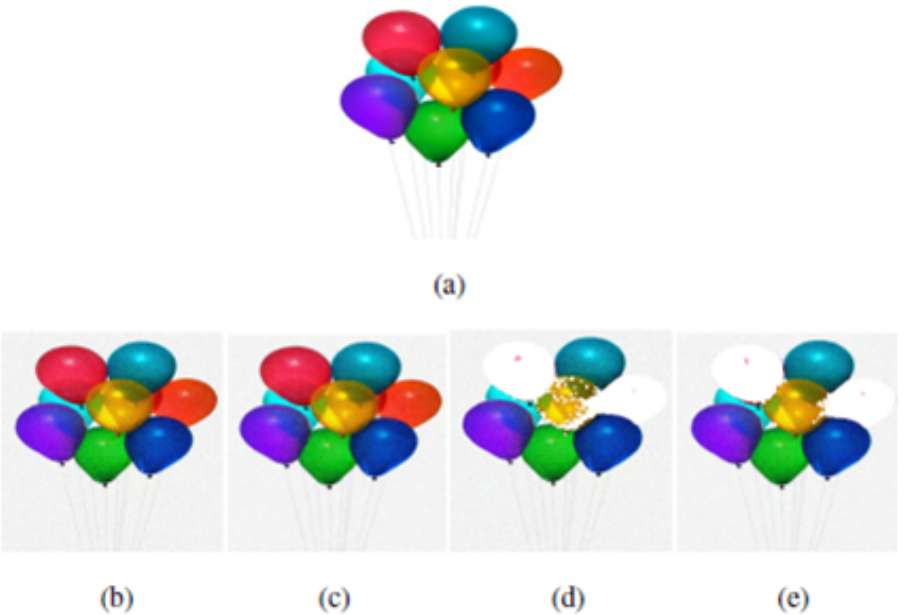


Fig. 5. Importance of Denoising (a) original image (b) noisy image (c) denoised image (d) noisy image segmentation (e) denoised image segmentation.

Tab. 3. Segmentation Comparison

Methods	Denoised Image	Noisy image
Rand Index (RI)	0.8121	0.6898
Global Consistency Error (GCE)	0.000022888	0.0177
Variation of Information (VI)	3.2990	4.3616

4.1. Comparative Analysis

The formulae used to compute the evaluation metrics PSNR and SDME values are given as follows:

Peak Signal to Noise Ratio (PSNR) The formula for PSNR value computation is,

$$\text{PSNR} = 10 \log_{10} \frac{E_{max}^2 \times I_w \times I_h}{\sum (I_{xy} - I_{xy}^*)} \tag{16}$$

where, I_w and $I_h \rightarrow$ Width and height of the de-noised image, $I_{xy} \rightarrow$ Original image

Fig. 6. Importance of Denoising (a) original image (b) noisy image (c) denoised image (d) noisy image segmentation (e) denoised image segmentation.

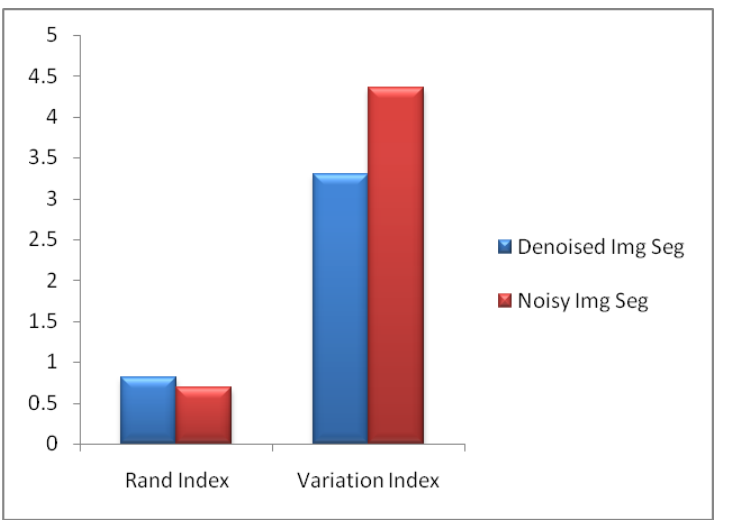


Fig. 7. Segmentation Comparison.

pixel value at coordinate (x, y), $I^*_{xy} \rightarrow$ De-noised image pixel value at coordinate (x, y), $E^2_{max} \rightarrow$ Largest energy of the image pixels.

Second-Derivative Measure of Enhancement (SDME) The formula for SDME value computation [15] is,

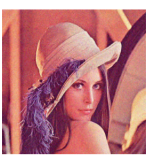
$$\text{SDME} = -\frac{1}{k_1 k_2} 20 \ln \left| \frac{I_{max;k,l} - 2I_{center;k,l} + I_{min;k,l}}{I_{max;k,l} + 2I_{center;k,l} + I_{min;k,l}} \right| \tag{17}$$

where the de-noised image is divided into $(k_1 \times k_2)$ blocks with odd size, $I_{max;k,l}$ and $I_{min;k,l}$ corresponding to the maximum and minimum values of pixels in each block whereas $I_{center;k,l}$ is the value of the intensity of the pixel in the center of each block.

Tab. 4 shows the comparison of PSNR and SDME values at various noise levels. The denoised output of the images for colored and gray scale images by ordinary DWT technique and suggested methods is illustrated in Fig. 8 and Fig. 9.

Tab. 4. PSNR and SDME comparison



Noise level	PSNR	SDME	PSNR	SDME	PSNR	SDME	PSNR	SDME
0.2	29.77923	92.02004	19.7439	93.1990	25.48929	81.79186	27.93929	87.21707
0.4	28.64254	86.91706	27.5100	90.44937	25.021	81.2642	26.8411	87.6524
0.6	28.22196	85.30869	27.6063	88.6106	24.65612	81.6432	25.9541	87.625
0.8	26.124	81.2356	26.489	86.1236	23.5516	81.2692	24.8782	87.5124
1.0	25.487	79.3654	25.985	82.3642	22.98454	81.3645	23.6423	87.35845

Fig. 8 and Fig. 9 shown that the suggested system presents improved presentation when compared to the ordinary DWT technique. The visuality of the improved image of suggested technique is high. The PSNR comparison is explained as follows.

Tab. 5 presents the values of the PSNR for both DWT and the suggested CWT technique for Lena and Pepper images. Fig. 10 illustrates the efficiency of the suggested technique as compared with normal DWT technique.

Tab. 5. The PSNR values for both DWT and CWT

Denoising methods	Lena	Peppers
DWT	28.12	28.43
CWT	33.45	32.17

Fig. 8. Comparison of Proposed method with existing DWT (a) Input noisy image (b) Denoising using ordinary DWT (c) Proposed method output.

5. Conclusion

Using Dual tree Complex Wavelet Transform, our document explains the idea of de-noising the images. The dual-tree CWT is an important improvement of the traditional real wavelet transform that is almost shifting invariant and, in elevated dimensions, directionally choosy. As the actual and imaginary fractions of the dual-tree CWT are, in reality, conventional real wavelet changes the CWT advantages from the vast theoretical, practical, and computational resources that have been improved for the standard DWT. In this research, we applied block matching method to recognize the related blocks to diminish the time consumption. The image denoising algorithm employs soft thresholding to offer smoothness and enhanced edge preservation. The suggested algorithm is checked with numerous kinds of images and the result demonstrates that the produced output images are enhanced in quality with less noise. The comparison of the de-noising results on natural images indicated that the proposed technique outperformed the other one in terms of the objective PSNR and SDME values and the visual quality assessment.

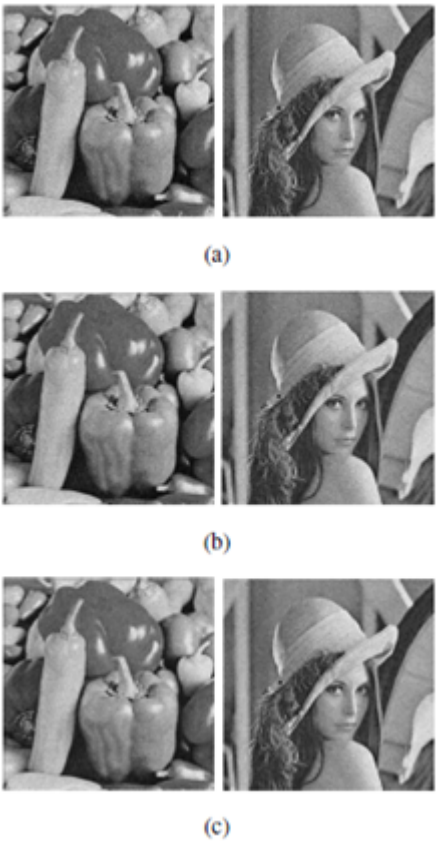


Fig. 9. Comparison of Proposed method with existing DWT (a) Input noisy image (b) Denoising using ordinary DWT (c) Proposed method output.

References

1985

[1] McVeigh E. R., Henkelman R. M., Bronskill M. J.: Noise and filtration in magnetic resonance imaging. Med. Phys., 12(5):586-591.

1992

[2] Simoncelli E. P., Freeman W. T., Adelson E. H., Heeger D. J.: Shiftable multi-scale transforms. IEEE Transaction of Information Theory, 38(2):587-607.

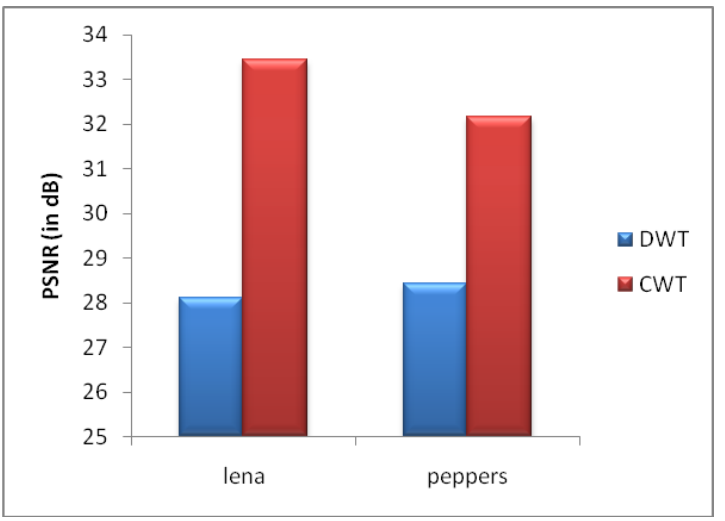


Fig. 10. PSNR Comparison.

1993

[3] Antoine P., Carrette P., Murenzi R., Piette B.: Image analysis with two-dimensional continuous wavelet transform. Signal Processing, 31:241-272.

1995

[4] David L. Donoho: De-noising by soft thresholding. IEEE Transactions on Information Theory, 41(3):613-627.

1999

[5] Candes E. J., Donoho D.: Curvelets – a surprisingly effective non adaptive representation for objects with edges. In: Curve and Surface Fitting, Saint-Malo, Vanderbuilt University Press.

[6] Kingsbury N.: Image processing with complex wavelets. Phil. Trans. R. Soc. London, UK.

[7] de Rivaz P., Kingsbury N.: Complex wavelet features for fast texture image retrieval. In: Proc. of the IEEE International Conference in Image Processing.

2000

[8] Kingsbury N.: Complex wavelets for shift invariant analysis and filtering of signals. Applied and Computational Harmonic Analysis.

2001

[9] Do M. N.: Directional multiresolution image representations. Ph.D. thesis, EPFL, Lausanne, Switzerland.

2002

[10] Romberg J. K., Choi H., Baraniuk R. G., Kingsbury N. G.: Hidden Markov tree models for complex wavelet transforms. IEEE Transactions on Signal Processing.

2005

[11] Gnanadurai D., Sadasivam V.: An efficient adaptive thresholding technique for wavelet based image denoising. International Journal of Signal Processing, 2(2):114-119.

[12] Marusic S., Deng G., Tay D.: Image Denoising Using Over-Complete Wavelet Representations. In: Proc. of the 13th European Signal Processing Conference, Antalya, Turkey.

[13] Selesnick I. W., Baraniuk R. G., Kingsbury N. G.: The Dual-Tree Complex Wavelet Transform. IEEE Signal Processing Magazine, 22(6):123-151.

2007

[14] Arivazhagan S., Deivalakshmi S., Kannan K.: Performance Analysis of Image Denoising System for different levels of Wavelet decomposition. International Journal of Imaging Science and Engineering (IJISE), 1(3):104-107.

2008

[15] Kachouie N. N.: Image Denoising Using Earth Mover's Distance and Local Histograms. International Journal of Image Processing, 4(1):66-76.

2010

[16] Aliaa A. A. Youssif, Darwish A. A., Madbouly A. M. M.: Adaptive Algorithm for Image Denoising Based on Curvelet Threshold. IJCSNS International Journal of Computer Science and Network Security, 10(1):322-328.

[17] Howlader T., Chaubey Y.P.: Noise Reduction of cDNA Microarray Images Using Complex Wavelets. IEEE Transactions On Image Processing, 19(8).

[18] Huhle B., Schairer T., Jenke P., Straßer W.: Fusion of range and color images for denoising and resolution enhancement with a non-local filter. Computer Vision and Image Understanding, 114:1336-1345.

[19] Palhano Xavier de Fontes F., Barroso G. A., Coupe P., Hellier P.: Real time ultrasound image denoising. Journal of Real-Time Image Processing, 2:1-14.

2012

[20] Chandan R., Sukadev M. A.: Hybrid Image Compression Scheme Using DCT and Fractal Image Compression. The International Arab Journal of Information, 10(6).

2011

[21] Moreno R., Garcia M. A., Puig D., Julià C.: Edge-preserving color image denoising through tensor voting. Computer Vision and Image Understanding, 115:1536-1551.

2012

[22] Chen G., Zhu W.P., Xie W.: Wavelet-based image denoising using three scales of dependency. IET Image Processing, 6(6):756-760.

2014

[23] Jaiswal A., Upadhyay J., Somkuwar A.: Image denoising and quality measurements by using filtering and wavelet based techniques. Int. J. Electron. Commun. (AEÜ), 68(8):699-705.

[24] Remenyi N., Nicolis O., Nason G., Vidakovic B.: Image Denoising With 2D Scale-Mixing Complex Wavelet Transforms. IEEE Transactions On Image Processing, 23(12):5165-5174.

[25] Srinivasan L., Rakvongthai Y., Oraintara S. : Microarray Image Denoising Using Complex Gaussian Scale Mixtures of Complex Wavelets. IEEE Journal Of Biomedical And Health Informatics, 18(4):1423-1430.

[26] Wang X.-Y., Liu Y.-C., Yang H.-Y. : Image denoising in extended Shearlet domain using hidden Markov tree models. Digital Signal Processing, 30:101-113.

2015

[27] Adamou-Mitiche A. B. H., Mitiche L., Naimi H. : Medical image denoising using dual tree complex thresholding wavelet transform and Wiener filter. Journal of King Saud University – Computer and Information Sciences, 27(1):40-45.

Syntactic Pattern Recognition of ECG for Diagnostic Justification

P. Flasiński*
IT Systems Department, Jagiellonian University, ul. St. Lojasiewicza 4, Cracow 30-384, Poland

Abstract. A novel hybrid structural-parametric model for ECG diagnostic justification is presented in the paper. In order to distinguish between specific subclasses of heart dysfunction phenomena both grammars and automata are enhanced with a formalism of dynamic programming. It allows one to construct a system, which is feasible for aiding a process of teaching and evaluating medical students' diagnostic reasoning in the area of electrocardiography.

Key words: syntactic pattern recognition, ECG analysis, diagnostic justification.

1. Introduction

Computers have been already aided ECG diagnosing for more than fifty years [1]. In the area of computer ECG analysis both decision-theoretic approach [10, 16, 20, 21, 23, 24, 26, 30, 32, 33] and syntactic pattern recognition methods [3, 6, 7, 11, 13, 15, 17, 25] have been commonly used. In syntactic pattern recognition a pattern is treated as a complex structure, which is decomposed into subpatterns that in turn are decomposed into simpler subpatterns, etc [4, 5, 8, 14]. In cardiology an ECG signal pattern is also treated as a linear structure, which consists of separable substructures describing the different phases of human heart's beating (e.g. P wave, T wave, ST segment, QRS complex). According to the syntactic pattern recognition paradigm a set of various structures is treated as a formal language. Words (structural patterns) of such a language can be analyzed by formal automata [4, 5, 8, 14], which not only are able to identify proper categories (diseases) for patterns, but also can characterize their structural features. Therefore, syntactic pattern recognition seems to be convenient, if a descriptive structural characterization is a goal of ECG analysis rather than only its classification (i.e. assigning an ECG signal to one of classes of heart dysfunction phenomena).

Providing an adequate *diagnostic justification* is a basic skill that is required during medical studies as well as at early stages of a physician professional development [12]. A diagnostic justification consists in explaining how key findings identified by a physician have allowed him/her to formulate initial hypotheses in order to achieve a final diagnosis [34]. Unfortunately, although this skill is crucial for improving diagnostic

*The author has conducted the research in a cooperation with IT Systems Department, Jagiellonian University.

Machine GRAPHICS & VISION 24(3/4):43–55, 2014. DOI: 10.22630/MGV.2014.24.3.4 .

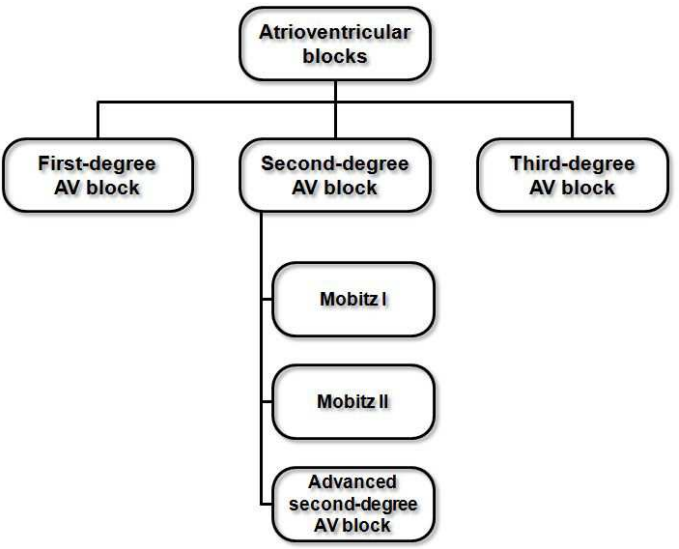


Fig. 1. A class of AV (atrioventricular) blocks and its subclasses.

competencies, medical students frequently exhibit a poor diagnostic justification performance [36]. The skill is especially important, if an interpretation of charts like ECG, EEG is concerned [18, 19]. Such an interpretation is made on the basis of structural features of the charts, as well as parametric values (frequencies of waves, lengths of segments, etc). Therefore, a research into constructing a syntactic pattern recognition-based system for teaching and evaluating students' diagnostic reasoning in the area of electrocardiography has been led since 2012 at the IT Systems Department, Jagiellonian University in Cracow.

Firstly, a set of structural primitives, which is feasible for the purpose of an ECG diagnostic justification has been identified in [35]. Then, a class of programmed attributed regular grammars, PARG has been defined as a tool for generating ECG patterns. System for Teaching ElectroCardioGraphy, STECG has been constructed as a syntax analyzer based on a class of programmed attributed finite-state automata, PAFSA [35]. During the use of the STECG system it has turned out that although the system distinguishes main classes of ECG abnormalities (e.g. various AV (atrioventricular) blocks, various branch blocks) it cannot differentiate between some of their specific subclasses in certain cases (e.g. between *Mobitz I* and *Mobitz II* subclasses of the *Second-degree AV (atrioventricular) block* class – cf. Fig. 1). It results from a too weak generative power of PARG grammars, and in consequence too weak discriminative power of corresponding

PAFSA automata. The results of a research into enhancing a generative/discriminative power of the model are presented in the paper.

In the section 2 preliminary definitions of programmed grammars and automata introduced in [35] are presented and reasons of a necessity of their enhancement are discussed. Enhanced dynamically programmed attributed regular grammars, DPARG are defined in section 3 for the purpose of generating patterns of specific subclasses of AV (atrioventricular) blocks. In section 4 a dynamically programmed attributed finite-state automaton, DPAFSA, which is an enhanced model of PAFSA automaton, is constructed. Its big discriminative power is discussed with the help of an example of recognizing *Mobitz I* and *Mobitz II* subclasses of the *Second-degree AV (atrioventricular) block* class. Concluding remarks concerning the second version of System for Teaching ElectroCardioGraphy, STECG v.2 and its role in teaching and verifying ECG diagnosis justification are contained in the section 5.

2. Preliminaries

As we have mentioned it above, the model proposed in [35] is based on the syntactic pattern recognition paradigm. It means that ECG charts are considered as defined with a set of structural primitives. For example, a subset of primitives used for a description of *Mobitz I* and *Mobitz II* subclasses of the *Second-degree AV block* class, which are discussed in this paper are shown in Fig. 2. On the other hand, during modeling and recognizing ECG charts for a purpose of a diagnostic justification parametric values of structural primitives (frequencies of waves, lengths of segments, etc) should be taken into account. Therefore, we have defined attributed grammars and automata and we have enhanced them by a programming formalism allowing us to make both production applications and transitions in automata conditional on values of primitive parameters.

In the succeeding two subsections (2.1, 2.2) we present definitions of programmed attributed regular grammar, PARG and programmed attributed finite-state automaton, PAFSA, which are characterized in a detailed way in [35].

2.1. Programmed attributed regular grammar

Let us introduce a definition of a programmed attributed regular grammar, PARG [35].

Definition 1. A programmed attributed regular grammar, PARG is a quadruple

$$G = (V, \Sigma, P, S), \quad \textit{where}$$

V is a finite set of symbols,
$\Sigma \subset V$ is a set of terminal symbols, $N = V \setminus \Sigma$ is a set of nonterminal symbols,
P is a finite set of productions of the form:

$$(\pi : \quad X \longrightarrow \alpha), \quad \textit{in which}$$

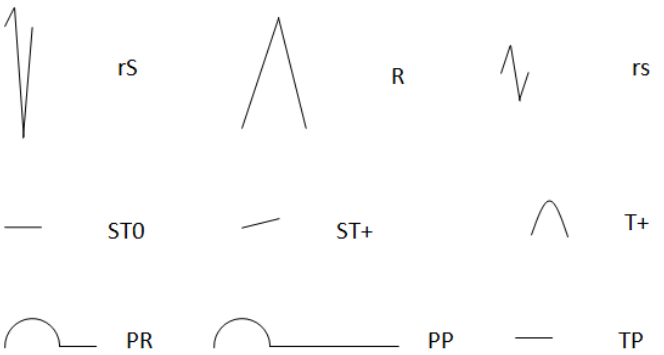


Fig. 2. A set of primitives used for a description of *Mobitz I* and *Mobitz II* subclasses of the *Second-degree AV block* class.

$\pi : \mathcal{A} \longrightarrow \{TRUE, FALSE\}$ *is the predicate of the production applicability,* $\mathcal{A}$ *is a finite set of attributes,* $X \in N$ *,*$\alpha \in \Sigma \cup \Sigma N$*,*
$S \in N$ *is the starting symbol.*

As we have already mentioned it, PARG is strong enough to model/describe structural patterns of general classes of phenomena observed in electrocardiography and their subclasses. It results from the use of the predicate of the production applicability, which tests whether primitive parameters fulfill certain *predefined* conditions.

However, during the use of the STECG system (System for Teaching Electrocardiography), it has turned out that in order to distinguish between specific classes of ECG phenomena, like *Mobitz I* and *Mobitz II* (see Fig. 1), in some cases parameters of primitives should be compared not with predefined constants, but with certain parameters of primitives analyzed previously. Therefore, in section 3 we will define an enhanced PARG with a programming formalism, which enables such comparisons.

2.2. Programmed attributed finite-state automaton

For a language of ECG patterns generated with the help of a programmed attributed regular grammar, a programmed attributed finite-state automaton, PAFSA as a tool for a syntax analysis has been defined in [35].

Definition 2. A programmed attributed finite-state automaton, PAFSA is a quintuple

$$A = (Q, I, \delta, q_0, F), \quad \textit{where}$$

Q *is a finite nonempty set of states,*
I *is a finite set of input symbols,*
δ *is the transition function of the form:*

$$\delta : Q \times I \times \Pi \longrightarrow Q \ , \textit{ in which}$$

$\Pi : \mathcal{A} \longrightarrow \{TRUE, FALSE\}$ *is the predicate of the transition permission,* $\mathcal{A}$ *is a set of attributes,*
$q_0 \in Q$ *is the initial state,*
$F \subseteq Q$ *is a set of final states.*

Similarly as in case of a programmed attributed regular grammar, the predicate of the transition permission in PAFSA should be predefined. It means that only a static parametrization of a transition is possible. Therefore, in section 4 we will define an enhanced PAFSA with a programming formalism, which allows one to make transitions in the automaton conditional on comparisons between (changing) parameters of various structural primitives processed till a current step of the automaton analysis.

3. Dynamically programmed attributed regular grammar

In order to show a necessity of introducing a dynamic programming mechanism to our PARG grammar/PAFSA automaton model, let us consider the following example of two specific subclasses of the *Second-degree AV blocks* class, namely: *Mobitz I* and *Mobitz II* (see Fig. 1). Their structural patterns are shown in Figure 3 (a) and (b), respectively. In both classes an occasional lack of a QRS complex occurs as it is shown in Fig. 3. However, whereas in *Mobitz II* a PR segment is of a constant length, in *Mobitz I* the first PR segment after the QRS complex lack is shorter than the last one before this lack. It means that in order to identify *Mobitz I* we should compare both PR segments. As a result, both a grammar and an automaton should remember the length of a previous PR segment and in case of the QRS complex lack the lengths of the corresponding PR segments should be compared.

Let us define a dynamically programmed attributed regular grammar, DPARG in the following way.

Definition 3. A dynamically programmed attributed regular grammar, DPARG is a quadruple

$$G = (V, \Sigma, P, S), \quad \textit{where}$$

V *is a finite set of symbols,*
$\Sigma \subset V$ *is a set of terminal symbols,* $N = V \setminus \Sigma$ *is a set of nonterminal symbols,*
P *is a finite set of productions of the form:*

$$(\pi : \quad X \longrightarrow \alpha, \quad CM), \quad \textit{in which}$$

$\pi : \mathcal{A}_\Sigma \cup \mathcal{V}_A \longrightarrow \{TRUE, FALSE\}$ *is the predicate of the production applicability,* $\mathcal{A}_\Sigma$ *is a finite set of attributes of terminal symbols,* $\mathcal{V}_A$ *is a finite set of auxiliary variables,*
$X \longrightarrow \alpha, \ X \in N \ , \alpha \in \Sigma \cup \Sigma N$ *is called the* core,
$CM : \mathcal{V}_A \longrightarrow VAL_A$ *is the control mapping ascribing values to auxiliary variables,* VAL_A *is a set of admissible values,*
$S \in N$ *is the starting symbol.*

Now, we define DPARG productions for the the subclass *Mobitz I*. We will use

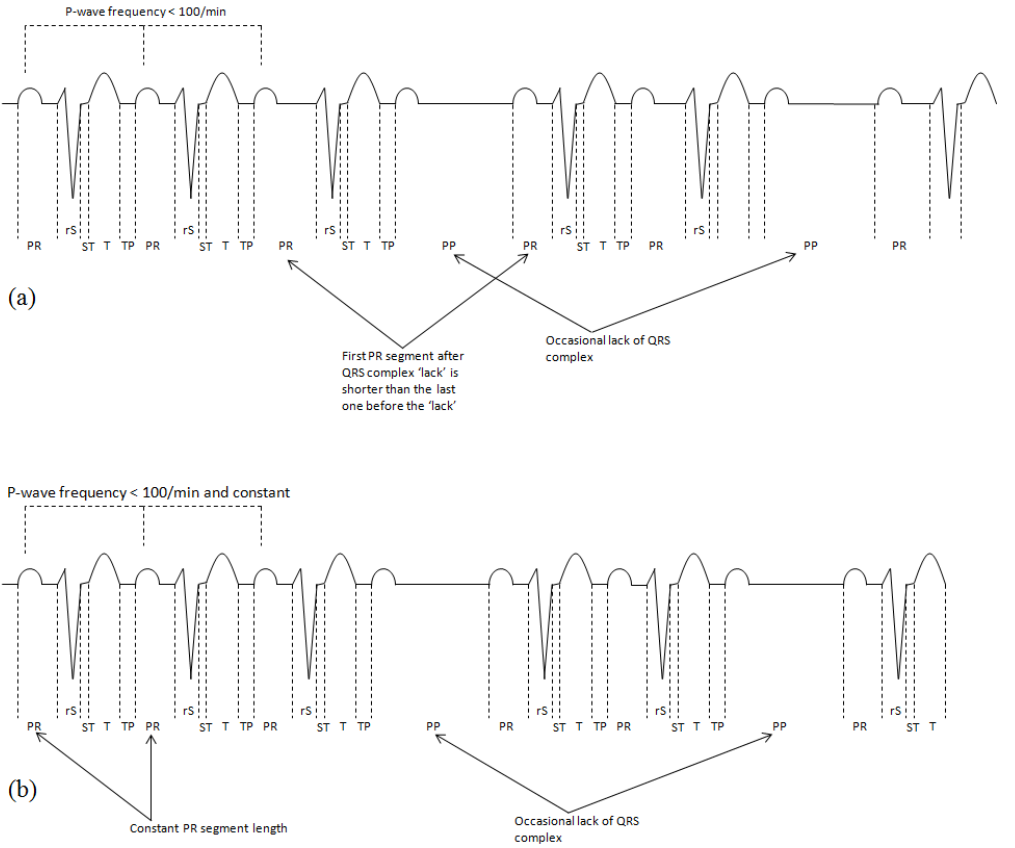


Fig. 3. Second-degree AV blocks of: (a) *Mobitz I* type, (b) *Mobitz II* type.

two auxiliary variables: the Boolean qrs_lack that will be set to $TRUE$ each time a lack of a QRS complex occurs, and the real $potent\_last\_before\_qrs\_lack$ used for storing the length of the last PR segment. We denote the length of a PR segment with l_{PR}. The structural primitives used for defining *Mobitz I* are denoted with: $\mathbf{PR}, \mathbf{rs}, \mathbf{ST}+, \mathbf{T}+, \mathbf{TP}, \mathbf{PP}$ (cf. Figures 2 and 3).

Firstly, the auxiliary variables are set to their initial values:
$qrs\_lack := FALSE$,
$potent\_last\_before\_qrs\_lack := 0$.
The DPARG productions are defined in the following way.

1. $\pi = (qrs\_lack \implies l_{PR} < potent\_last\_before\_qrs\_lack)$:
   $X^{(0)} \longrightarrow \mathbf{PR}\, X^{(1)}$,
   CM : $potent\_last\_before\_qrs\_lack := l_{PR}$; $qrs\_lack := FALSE$,
2. $\pi = TRUE$: $X^{(1)} \longrightarrow \mathbf{rs}\, X^{(2)}$, $CM : none$,
3. $\pi = TRUE$: $X^{(2)} \longrightarrow \mathbf{ST}+\, X^{(3)}$, $CM : none$,
4. $\pi = TRUE$: $X^{(3)} \longrightarrow \mathbf{T}+\, X^{(4)}$, $CM : none$,
5. $\pi = TRUE$: $X^{(4)} \longrightarrow \mathbf{TP}\, X^{(5)}$, $CM : none$,
6. $\pi = TRUE$: $X^{(5)} \longrightarrow \mathbf{PP}\, X^{(6)}$, CM : $qrs\_lack := TRUE$,
7. $\pi = TRUE$:
   $X^{(5)} \longrightarrow \mathbf{PR}\, X^{(7)}$,
   CM : $potent\_last\_before\_qrs\_lack := l_{PR}$; $qrs\_lack := FALSE$,
8. $\pi = (qrs\_lack \implies l_{PR} < potent\_last\_before\_qrs\_lack)$:
   $X^{(6)} \longrightarrow \mathbf{PR}\, X^{(7)}$,
   CM : $potent\_last\_before\_qrs\_lack := l_{PR}$; $qrs\_lack := FALSE$,
9. $\pi = TRUE$: $X^{(7)} \longrightarrow \mathbf{rs}\, X^{(2)}$, $CM : none$.

As one can easily see, the productions defined above allow us to model *Second-degree AV block* of the *Mobitz I* type (see Fig. 3a).

4. Dynamically programmed attributed finite-state automaton

After defining a dynamically programmed grammar, we can construct a dynamically programmed attributed finite-state automaton, DPAFSA, which is based on the PAFSA [35] presented in section 2.2. In a DPAFSA automaton a mechanism of a transition control is strengthened by adding a set of working memory objects allowing the automaton to store the values of those attributes, which occur in one of the predicates of the transition permission.

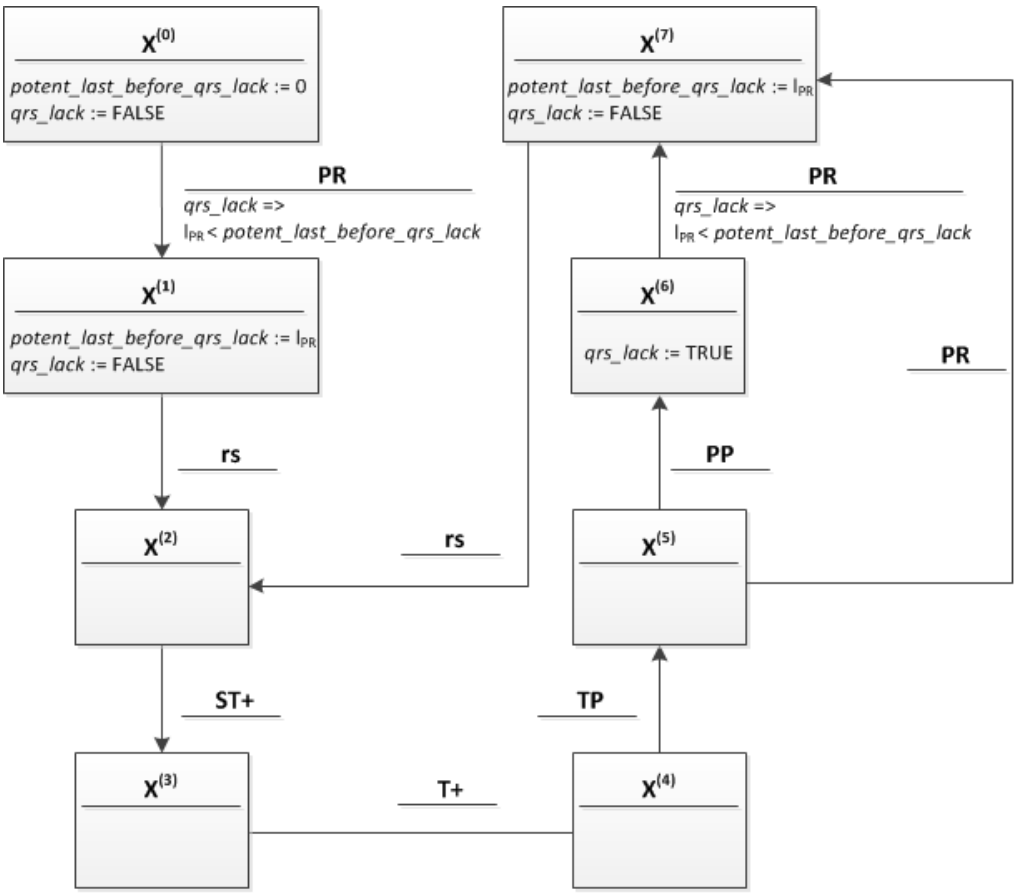


Fig. 4. A part of a dynamically programmed attributed finite-state automaton for a class *Second-degree AV block* of the *Mobitz I* type.

Let us introduce the following definition of a dynamically programmed attributed finite-state automaton, DPAFSA.

Definition 4. A dynamically programmed attributed finite-state automaton, DPAFSA is a sextuple

$$A = (Q, I, M, \delta, q_0, F), \quad \textit{where}$$

Q is a finite nonempty set of states,
I is a finite set of input symbols,
M is a finite set of working memory objects,
δ is the transition function of the form:

$$\delta : Q \times I \times \Pi \longrightarrow Q \times CM, \textit{ in which}$$

$\Pi : \mathcal{A}_I \cup \mathcal{A}_M \longrightarrow \{TRUE, FALSE\}$ is the predicate of the transition permission, $\mathcal{A}_I$ is a set of attributes of input symbols, $\mathcal{A}_M$ is a set of attributes of working memory objects, $CM = \{cm_{q_i,q_j} : \mathcal{A}_M \longrightarrow VAL_M, q_i, q_j \in Q\}$ is a set of control mappings ascribing values to attributes of working memory objects, VAL_M is a set of admissible values,
$q_0 \in Q$ is the initial state,
$F \subseteq Q$ is a set of final states.

An example of a part of a dynamically programmed attributed finite-state automaton for a class *Second-degree AV block* of the *Mobitz I* type is shown in Fig. 4. As we can see there is a one-to-one correspondence between transitions of the DPAFSA automaton and the productions of the DPARG grammar defined in section 3.

5. Conclusions

As we have mentioned it in the introduction, providing an adequate *diagnostic justification* is a basic skill that is required during medical studies and first years of a physician professional development. On the other hand, medical students and novice physicians frequently exhibit a poor performance in this area. There are two types of information processing in medicine: analytic and automatic [31]. During automatic processing a diagnostic justification is made with the help of a pattern recognition-like mechanism, i.e. an unknown case to be diagnosed is unconsciously compared with known cases from the past [31]. Thus, this pattern recognition-like mechanism is analogous to (standard) pattern recognition scheme used in computer science. This kind of achieving a diagnosis is typical for experienced physicians. On the other hand, analytic processing consists in analyzing, synthesizing and interpreting the case itself on the basis on a biomedical knowledge (not the clinical one) [31]. This kind of diagnosing is, thus, based on general models learnt during a medical education and it is typical for students and novice physicians. In case of an interpretation of ECG, a structure of a chart and its generic parametrization is analyzed during a diagnosing process. It in turn is analogous to syntactic pattern recognition scheme in computer science. Therefore, this approach seems

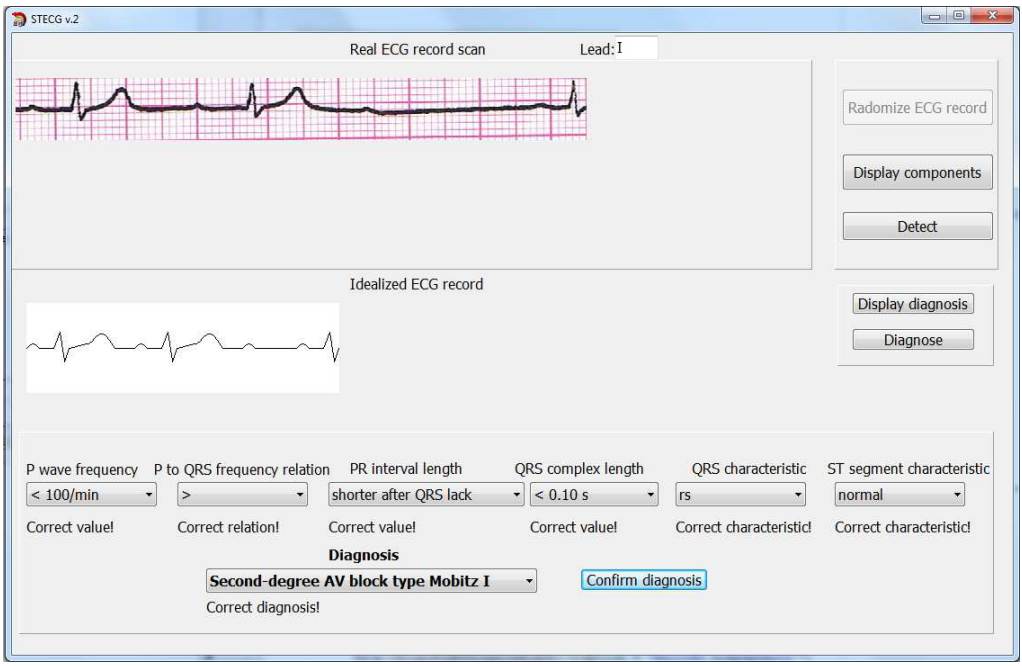


Fig. 5. An example of testing a diagnostic justification of *Second-degree AV (atrioventricular) block* of the *Mobitz I* type by the STECG v.2 system.

to be more feasible for designing systems for aiding medical students in achieving the ECG diagnostic skill *via* testing their diagnostic justification.

The System for Teaching ElectroCardioGraphy, STECG has been implemented on the basis of programmed attributed regular grammars and programmed attributed finite-state automata [35]. During its use it has turned out that although STECG distinguishes main classes of ECG abnormalities, some specific ECG types, like *Mobitz I* and *Mobitz II* are hardly distinguishable, because of a too weak discriminative power of the automaton mentioned above and used as a syntax analyzer. Therefore, the formal model introduced in [35] had to be enhanced. The dynamically programmed grammars and automata presented in this paper have helped to solve this problem.

The STECG v.2 system allows us to analyze ECG signals, in which current and previous values of primitives' parameters are to be compared. This way the system is able to differentiate between *Mobitz I* and *Mobitz II* types of *Second-degree atrioventricular blocks*. An example of testing a medical student diagnostic justification for *Mobitz I* is shown in Fig. 5.

References

1961

[1] Stallmann F. W., Pipberger H. V.: Automatic recognition of electrocardiographic waves by digital computer. Circ. Res., 9:1138-1143.

1969

[2] Rosenkrantz D. J.: Programmed grammars and classes of formal languages. Journal of the Association for Computing Machinery, 16:107-131.

1975

[3] Horowitz S. L.: A syntactic algorithm for peak detection in waveforms with applications to cardiography. Comm. ACM, 18:281-285.

1977

[4] Pavlidis T.: Structural Pattern Recognition. Springer, New York.

1978

[5] Gonzales R. C., Thomason M. G.: Syntactic Pattern Recognition: An Introduction. Addison-Wesley, Reading.

1979

[6] Belferte G., De Mori R., Ferraris F.: A contribution to the automatic processing of electrocardiograms using syntactic methods. IEEE Trans. Biomed. Eng., 26:125-136.

1980

[7] Udupa J., Murthy I. S. N.: Syntactic approach to ECG rhythm analysis. IEEE Trans. Biomed. Eng., 27:370-375.

1982

[8] Fu K. S.: Syntactic Pattern Recognition and Applications. Prentice Hall, Englewood Cliffs.

1983

[9] Bunke H. O.: Graph grammars as a generative tool in image understanding. Lecture Notes in Computer Science, 153:8-19.

1985

[10] Pan J., Tompkins W.J.: A real-time QRS detection algorithm. IEEE Trans. Biomed. Eng., 32:230-236.

1986

[11] Papakonstantinou G., Skordalakis E., Gritzali F.: An attribute grammar for QRS detection. Pattern Recognition, 19:297-303.

[12] Patel V., Groen G.: Knowledge based solution strategies in medical reasoning. Cognitive Science, 10:91-116.

[13] Skordalakis E.: Syntactic ECG processing: a review. Pattern Recognition, 19:305-313.

1990

[14] Bunke H. O., Sanfeliu A. (eds.) : Syntactic and Structural Pattern Recognition - Theory and Applications. World Scientific, Singapore.

[15] Trahanias P., Skordalakis E. : Syntactic pattern recognition of the ECG. IEEE Trans. Patt. Analysis Mach. Intell., 12:648-657.

1991

[16] Piętka E. : Feature extraction in computerized approach to the ECG analysis. Pattern Recognition, 24:139-146.

1995

[17] Koski A., Juhola M., Meriste M. : Syntactic recognition of ECG signals by attributed finite automata. Pattern Recognition, 28:1927-1940.

1997

[18] Gilhooly K. J., McGeorge P., Hunter J. et al. : Biomedical knowledge in diagnostic thinking: the case of electrocardiogram (ECG) interpretation. European Journal of Cognitive Psychology, 9:199-223.

[19] Kolykhalov I. V., Iznak, A. F., Chayanov N. V. et al. : EEG mapping in diagnostic assessment of patients with mild dementia. European Neuropsychopharmacology, 7:248.

1998

[20] Barro S., Fernandez-Delgado M., Villa-Sobrino J. A., Regueiro C. V., Sanchez E. : Classifying multichannel ECG patterns with an adaptive neural network. IEEE Eng. Med. Biol. Mag., 17:45-55.

[21] Maglaveras N., Stamkopoulos T., Diamantaras K., Pappas C., Strintzis M. : ECG pattern recognition and classification using non-linear transforms and neural networks: a review. Int. J. Med. Inform., 52:191-208.

1999

[22] Flasiński M., Jurek J. : Dynamically programmed automata for quasi context sensitive languages as a tool for inference support in pattern recognition-based real-time control expert systems. Pattern Recognition, 32:671-690.

2001

[23] Osowski S., Linh T.H. : ECG beat recognition using fuzzy hybrid neural network. IEEE Trans. Biomed. Eng., 48:1265-1271.

2002

[24] Sternickel K. : Automatic pattern recognition in ECG time series. Comp. Meth. Programs in Biomedicine, 68:109-115.

2003

[25] Tumer M. B., Belfore L. A., Ropella K. M. : A syntactic methodology for automatic diagnosis by analysis of continuous time measurements using hierarchical signal representations. IEEE Trans. on Syst. Man Cybern., 33:951-965.

2004

[26] Engin M.: ECG beat classification using neuro-fuzzy network. Patt. Rec. Lett., 25:1715-1722.

[27] Flasiński M., Reroń E., Wójtowicz P., Atlasiewicz K.: Mathematical linguistics model for medical diagnostics of organ hearing in neonates, Lecture Notes in Computer Science (LNAI), 3019:746-753.

[28] Tadeusiewicz R., Ogiela M. R.: Medical Image Understanding Technology. Springer, Berlin-Heidelberg-New York.

2006

[29] Flasiński M., Jurek J.: On the analysis of fuzzy string patterns with the help of extended and stochastic GDPLL(k) grammars. Fundamenta Informaticae, 71:1-14.

2007

[30] Dong J., Xu S., Zhan C.: ECG recognition and classification: approaches, problems and new method. J. Biomed. Eng., 24:1224-1229.

[31] McLaughlin K. J.: The Contribution of Analytic Information Processing to Diagnostic Performance in Medicine, Ph.D. Thesis, Erasmus University Rotterdam.

2009

[32] Martis R. J., Chakraborty C., Ray A. K.: A two-stage mechanism for registration and classification of ECG using Gaussian mixture model. Pattern Recognition, 42:2979-2988.

[33] Noponen K., Kortelainen J., Seppanen T.: Invariant trajectory classification of dynamical systems with a case study on ECG. Pattern Recognition, 42:1832-1844.

2013

[34] Cianciolo A. T., Williams R. G., Klamen D. L., Roberts N.K.: Biomedical knowledge, clinical cognition and diagnostic justification: a structural equation model. Medical Education, 47:309-316.

[35] Flasiński M., Flasiński P., Konduracka E.: On the use of programmed automata for verification of ECG diagnoses. Advances in Intelligent Systems and Computing 226, Springer, Berlin-Heidelberg-New York, 591-599.

2014

[36] Williams R. G., Klamen D. L., Markwell S. J. et al: Variations in senior medical student diagnostic justification ability. Academic Medicine, 89:790-798.

A Survey of Passive 3D Reconstruction Methods on the Basis of More than One Image

Mariusz Siudak[1], Przemyslaw Rokita[2]
[1] *Military University of Technology, Institute of Teleinformatics and Automatics, Warsaw, Poland* *msiudak@wat.edu.pl*
[2] *Warsaw University of Technology, Faculty of Electronics and Information Technology, Institute of Computer Science, Warsaw, Poland* *pro@ii.pw.edu.pl*

Abstract. The research on the 3D scene reconstruction on the basis of its images and video recordings has been in progress for many years. As a result there is a number of methods concerning how to manage the reconstruction problem. This article's goal is to present the most important methods of reconstruction including stereo vision, shape from motion, shape from defocus, shape form silhouettes. shape from photo-consistency. All the algorithms explained in this article can be used on images taken with casual cameras in an ordinary illuminated scene (passive methods).

Key words: 3D reconstruction, stereo vision, shape from motion, shape from defocus, shape from silhouettes, shape from photo-consistency, shape from X.

1. Introduction

One of the key trends in research on machine vision is 3D scene reconstruction on the basis of typical, digital images and videos. There is a huge interest in this field, because the number of possible applications is immense. To name but a few, which seem to be the most obvious: in medicine, helping visually impaired, robotics, navigation, the film industry, video games, 3D reality and augmented reality. 3D object models can be obtained by using devices based on active methods, that is, methods which control the illumination of the scene (such as Kinect). The reconstruction of the scene can be obtained be measuring the return time of reflected waves or measuring their shift in the phase (laser scanner). Active methods do not perform as effectively in sunlight or in a poorly controlled environment. Usually, specialist scanning devices are expensive and only but a few experts have access to them. Digital cameras are generally available, miniature CCD/CMOS converters can be found in nearly every mobile phone. A cheap digital video camera connected with a 3D reconstruction algorithm can be a satisfactory alternative to expensive specialist equipment. Passive reconstruction algorithms are advantageous, because they may help create 3D models of objects, which no longer exist, but which can be seen only on archive images and recordings, such as the reconstruction of Warsaw from 1935. Widespread availability of cheap video cameras may be of huge significance when thinking of the visually impaired. The mobile phone could

create a map of the surroundings and inform with sounds about the distance of nearby obstacles. The passive reconstruction methods also give the opportunity to use massive amounts of visual information available on the web. To give an example, there is a project, "Building Rome in a Day" in which basing on a collection of thousands of images of Rome, they made a visual reconstruction of one of the city's main parts. Unfortunately passive 3D methods also have limitations. Even though there has been over 50 years of intensive research there is still not a single universal and faultless passive 3D reconstruction method.

Digital images are a recorded projection of the three-dimensional scene on a two-dimensional CCD/CMOS matrix's surface. By projecting the three-dimensional object on the two-dimensional surface we lose the information about the third dimension. One point on the image plane may be a projection of infinite number of 3D points. Without additional knowlage about the scene, basing on the analysis of geometrical relations in one image it is not possible to measure the distance of scene points from the image plane. The information about the third dimension is present in image but not directly (for example,in the relationship between the surface radiation and the angle between the source of light and the main point of the video camera; the blurring depends on the distance from the place of its main focus; the perspective phenomenon). The reconstruction of the 3D scene's shape on the basis of one image is difficult and most often does not give satisfactory results. We can obtain much better results with video camera recordings from different angles or from many video cameras recording at the same time and given that one can analyze the geometrical dependencies between the projections of the same points on different captions. One of most crucial as well as the most difficult steps in multi-image methods are camera pose estimation and matching of corresponding pairs of points. The matter is complicated by the fact that these both tasks are interdependent. Making an error when matching the points leads to an incorrectly determined position of the camera and the other way around, the errors made while positioning the cameras lead to faulty determination of the 3D points' position (leading to mismatch points). The reason why there is a difficulty in finding the suitable pairs comes from the fact that the decision in matching the right pixels in pairs has to be made only on the basis of the brightness function. One should remember that the acquisition of images may happen in different illumination, that there are occlusions in images or that there may be reflections of the light, or that there might be acquisition when the objects or the video camera are in motion. The fundamental significance for the passive methods lies in the quality of the analyzed image. Unfortunately the digital image drastically differs from the ideal optical image.

The digital image is an approximation of the optical image given by sampling brightness function and by attributing values to the areas, called pixels. The rectangular shape of the pixel does not reflect the photo-optical element's real shape, which in addition

does not cover the whole area represented by the pixel. The brightness value of a single pixel additionally cumulates in itself the errors coming from the motion blur, multiplicative and additive distortions, noises, quantizations, spatial reallocation of the colors and errors coming from the use of Bayer filter. The optical system generates additional problems in mapping the scene. Spherical distortions (the tangent and radial), chromatic aberrations, effects connected with depth of focus (blurring outside of the focus area), end up in loss or deformation of the information about the source image. The last element which we would like to mention is the loss of information connected to faulty image compression. The elimination of higher frequencies, averaging the brightness of image areas and other artifacts connected with the compression may lead to additional errors. The final source of information about the shape of the scene for the passive methods are whirred, distorted and discrete in space and values two-dimensional brightness functions.

The passive methods of reconstruction differ in required means of image acquisition (the number and arrangement of cameras), the method of calibration of the mutual camera location, the method of matching points and also differ in the source of information about the scene's shape. For some number of methods the camera position calibration may be an independent process, but for other methods it their integral part. The methods are to a different degree resistant to the errors in mapping the real image, mostly if it comes to noises, faulty point matching and wrong calibration. It is assumed that the methods (or the group of methods) are called "shapes from X", where "X" is the main source of information about shape of the scene, such as the shape from photo-consistency. The following chapters contain descriptions of the most important passive methods in 3D reconstruction. It is not the full list of methods mentioned in the literature, however, the chosen methods represent the key ideas in 3D reconstruction. First, we would like to explain the method of obtaining the shape from stereo-vision.

2. The reconstruction methods

2.1. The shape from stereo-vision

Stereo-vision may be defined as the method of calculating the depth map from two images obtained from two camera system. It is characteristic for stereo-vision that the distance of the cameras is relatively small and constant (the cameras are connected together by a stiff construction). It is assumed that, the image acquisition takes place in the same time for both cameras. One can distinguish the following stereo-vision arrangements [61]:

- side-camera canonical (parallel optical axes)
- side-camera with intersecting optical axes
- axis-motion

The mostly common in the literature is the canonical arrangement, as can be seen in Fig. 1. This arrangement consists of two identical video cameras that have CCD/CMOS converters on the same surface, the appropriate matrix edges are parallel to each other and the optical means are only moved on the X axis.

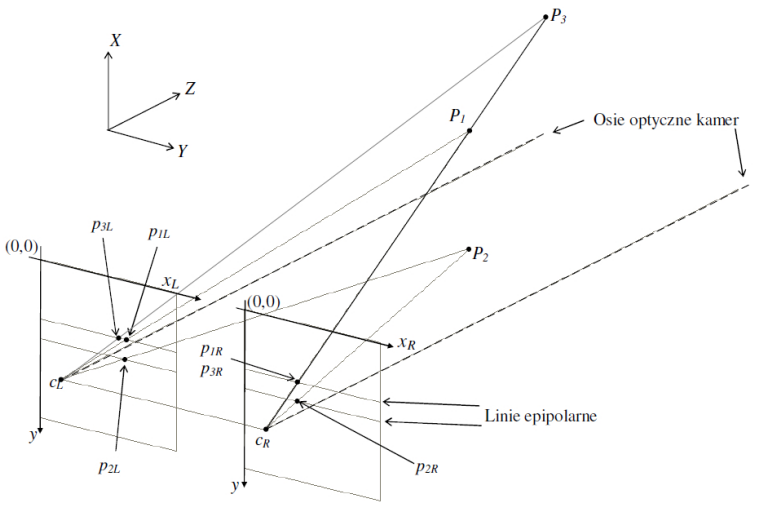


Fig. 1. Canonical stereo-vision arrangement [52].

With knowledge of the distance between cameras and the location of the same points in the image from the left and the right camera, we can determine the location of 3D points using the triangulation method. Unfortunately, one can never observe the canonical arrangement in practice. It is due to the fact, the stereo-vision arrangements consist of two, independently produced digital cameras, the optical axes are not ideally coaxial, CCD matrices are not ideally parallel and Y and Z axes are reallocated. Additionally, every optical system causes distortions in the image. The optical distortions parameters are usually different for both video cameras. To able to use one of the triangular methods, one should change the real arrangement into the canonical arrangement. All the stereo-vision algorithms steps could be differentiated as follows:

- camera calibration
- images standardization
- stereo matching
- triangulation

Stereo-vision may be the source of a dense, but also of a sparse depth map. The difference in implementation lies in the way of choosing the appropriate pairs of points.

Most commonly used methods to obtain sparse map are based on image local feature descriptors (such as SIFT, SURF). Unfortunately dense maps are much more difficult to obtain because it requires unambiguously matching for every pair of pixels. In the further part of the article, we will discuss the algorithms which allow us to create a dense map.

2.1.1. Stereo calibration

Calibration of camera set is a process of estimating the parameters of image formation equations and mutual rotation and translation of cameras. Calculating the matrices of inner parameters of single camera and vector's distortion parameters may be conducted separately for every camera. The topic of single camera calibration is another matter, widely described in the literature. It is characteristic for stereo-vision to calculate the mutual rotation and translation of the cameras. When one knows the parameters, its possible to calculate an essential matrix, using the rules of epipolar geometry. An essential matrix is created by multiplying an anti-symmetrical matrix (obtained from a translation vector) by a rotation matrix. An essential matrix has to comply with the equation (1):

$$(x')^T E x = (x')^T [T]_\times R x = 0 \tag{1}$$

where:

$x, x' \in \mathbb{P}^2(\mathbb{R})$	–	homogeneous coordinates of the same 3D point projection on the left and right image plane
$E \in \mathbb{R}^{3\times3}$	–	essential matrix
$[T]_\times \in \mathbb{R}^{3\times3}$	–	anti-symmetric translation matrix is such that for every vector $[T]_\times v = T \times v$
$R \in SO(3)$	–	rotation matrix

By multiplying the point's coordinates on the first image plane by an essential matrix we calculate, so called, epipolar line. It is a line on the image plane of the second camera on which there has to lie corresponding point. Information about the essential matrix allows to reduce the complexity of point matching. Instead of analyzing the two-dimensional spaces one can analyze the one-dimensional (only points which create epipolar lines). The most popular algorithm of calculating an essential matrix is the eight-point algorithm [13]. By singular value decomposition and choosing from four possible solutions to the system of equations one may obtain the translation and rotation matrices. Direct use of essential matrices is unhandy, because it is necessary to use image plane coordinates . This problem may be solved, using a fundamental matrix instead of an essential matrix. A fundamental matrix can be calculated by multiplying an essential matrix (E) by matrices of internal parameters ($K_1, K_2 \in \mathbb{R}^{3\times3}$), fulfills the equation (2):

$$F = \left(K_2{}^{-1}\right)^T E K_1{}^{-1} \tag{2}$$

Fundamental matrix allows one to calculate the position of epipolar lines in pixel coordinates system. Calculating a fundamental matrix is possible using one of the known algorithms [24, 20, 47]. Due to the fact that the pairs of points location is uncertain (it results from noises and limited possibility of describing characteristic points), robust algorithms have huge impact on their practical implementation. A well known algorithm, improving the resistance to faulty point matching is RANSAC (random sample consensus) [3].

2.1.2. The standardization of images

The standardization is the transformation of images which enables them to be effectively processed by algorithms. Standardized images allows for use of fast algorithms for pixel matching. The first task of standardization is to transform the image and by that correct the differences in the internal camera parameters, such as scaling, rotation and reallocation of the image's centre. The next important step is to erase the distortion effect. The correction algorithm consists of two phases. First of all, one needs to calculate the correct distance of the points from the optical image center and the distortion coordinates' vector. The second phase is called resampling. When fundamental matrix is known it is possible to find the corresponding pixel, going through the set of pixels which build the epipolar line. Improve of algorithms speed is achieved by the next step called rectification. Rectification is an affine image transformation in which corresponding epipolar lines are horizontal, parallel and have the same y coordinate. Rectification effects can be seen in the Fig. 2.

An infinite number of affine transformations fulfills the rectification conditions. From the space of possible solutions is chosen solution that minimize image deformation. A number of rectification algorithms were developed [25, 26, 28, 34, 50, 54, 68, 72]. As a side effect of correcting distorsion and rectification there are blank spots in the image. Last step in the image standardization is to seperate those unused image fragments and standardization its size.

2.1.3. Stereo matching

The most popular subject in the literature about stereo-vision is the problem of finding the appropriate pixel pairs. The aim is to calculate the shift between pixel pairs on the x axis between images. This reallocation is called horrizontal parallax or most commonly, disparity. Even for standardized images the problem of calculating the appropriate pixel pairs is difficult. It is mostly due to the noises in the images, occlusion, loss of a significant amount of visual information through quantification of brightness levels. The classification of matching stereoscopic pairs methods [87]:

- ◇ pixel matching
  - ○ regions matching (local)

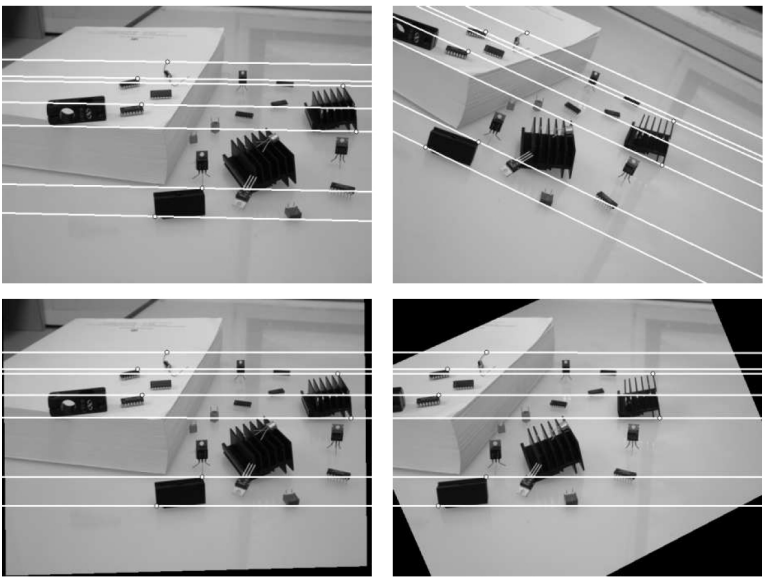


Fig. 2. The upper line – the images before rectification, the bottom line – the images after rectification [50].

- cooperational
- window matching

○ image matching (global)
- dynamic programming
- graph cut
- relaxation
- genetic algorithms
- non-linear diffusion

◇ image features matching
- ○ edges
- ○ corners
- ○ tensors
- ○ edgels

It would take far too long to describe every group, so we will concentrate only on the two most popular approaches: windows matching and graphs cut.

Matching windows is so called a local method because decision of choosing the appropriate pixel is based solely on the pixels local neighborhood. Cost function of matching

is defined as sum of brightness difference in the analysis window around the pixel in the first image and for the second image. The functions minimal value is set by the pixel which surroundings are the most similar and on this basis it is chosen as the best fitting pixel. Matching pixels aims to determine the disparity (3)

$$d_{ij}^{*} = \arg\min_{d}\{C(i,j,d)\} \tag{3}$$

where:
C – matching cost function
i, j – the pixel's coordinates in the first image
d – disparity

Matching cost function has many forms in the literature. As the most popular we can consider:

- sum of absolute differences (SAD):

$$C_{\mathrm{SAD}}(i,j,d) = \sum_{\{m,n\}\in W} |I_1(i+m, j+n) - I_2(i+m+d, j+n)| \tag{4}$$

- sum of squared differences (SSD):

$$C_{\mathrm{SSD}}(i,j,d) = \sum_{\{m,n\}\in W} \left(I_1(i+m, j+n) - I_2(i+m+d, j+n)\right)^2 \tag{5}$$

- normalized cross correlation (NCC):

$$C_{\mathrm{NCC}}(i,j,d) = \frac{\sum\limits_{\{m,n\}\in W} I_1(i+m, j+n)\cdot I_2(i+m+d, j+n)}{\sqrt{\sum\limits_{\{m,n\}\in W} I_1^2(i+m, j+n)\cdot \sum\limits_{\{m,n\}\in W} I_2^2(i+m+d, j+n)}} \tag{6}$$

where:
W – analysis window
I_1, I_2 – brightness functions of, correspondingly, the first and the second image

The method of matching windows is easy in implementation and allows the image to be processed in real time. Unfortunately, the local methods tend not to give satisfactory results. One of the reasons why they are rather ineffective is that the pixels are analysed separately. Moreover, some of the pixels from the first image may be connected with the same pixels from the second image. Due to the fact that only the difference in pixels brightness is analyzed, this method is highly sensitive to all types of noise in the brightness function.

Significant progress in the field of matching stereo-vision images was made by using graph cuts methods [32, 62, 81]. The task of matching a stereoscopic pair could be defined as a task of attributing every pixel p from a set of image pixels $P = \{0, .., x_{\max}\} \times \{0, .., y_{\max}\}$ to a disparity value from the set $\zeta = \{d_{\min}, .., d_{\max}\}$. In the language of graph theory it is the problem of labeling the vertices of the graph. Problem of labeling could be described as the task of finding such a set of pixel labels $L = \{d_p : p \in P, d \in \zeta\}$ for which the function cost is minimal (7):

$$E(L) = \underbrace{\sum_{p \in P} C\left(d_p\right)}_{\text{data component}} + \underbrace{\sum_{p,q \in N(p)} K(d_p, d_q)}_{\text{smoothness component}} \tag{7}$$

where:

$C\left(d_p\right)$	–	cost function of assign pixel p an label d_p
$N(p)$	–	set of neighborhood pixels of pixel p
$K(d_p, d_q)$	–	cost function of assigning pixel p a label d_p and pixel q a label d_q

In the literature exists many forms of function C including previously described C_{SAD}. The function $K(d_p, d_q)$ defines the fee (the additional cost) in a case when $d_p \neq d_q$. Finding global minimum of function's $E(L)$ is in general case an NP problem. The acceptable minimization times can be obtained by looking for a close solution by calculating the minimal s/t cuts in a specially constructed graph [32]. The algorithm begins with construction of graphs $G = \langle V, \xi \rangle$, where V is a set of vertices and ξ is a set of edges. The set of vertices consists all elements of Cartesian product of pixels set and the labels set $U = P \times \zeta$. Additionally, all of the first image's pixels (referential) are added to the set of vertices, defining them as "sources" s and pixels of the second image, defining them as "sinks" t. At the end the set of all the vertices is $V = s \cup U \cup t$. For every pair of adjacent vertices from the set U, so called, n-edge is added $N = \left\{\langle p, q\rangle \in U^2 : \|p - q\| = 1\right\}$. For every vertex of minimal and maximal disparity a t-edge is built which conjoins them with corresponding vertices of the appropriate set s and t; hence $\Theta = \{\langle\langle p, q\rangle, \langle p, q, d\rangle\rangle \in s \times U : d = d_{\min}\} \cup \{\langle\langle p, q, d\rangle, \langle p, q\rangle\rangle \in U \times t : d = d_{\max}\}$. Finally, the set of edges is $\xi = N \cup \Theta$. The illustration of such a graph has been presented in Fig. 3 (middle).

Then, the weight function $w : \xi \to \mathbb{R}$ which represent the cost assignment to the edges is defined. The n-edge's weight represents the fee for lack of smoothness, however, t-edges' weight is the value of cost function for a single pixel. S/T graph cut is defined as a division of a set of graph's G vertices V on subsets and such, in which every "sinks" are in set T and every "sources" are in set S. An example of s/t cuts can be seen in 3 (left) and (right). The solution to the task of labeling pixels is done by iterative algorithms based on $\alpha\beta$-swap or α-expansion. The second algorithm give slightly better results [58].

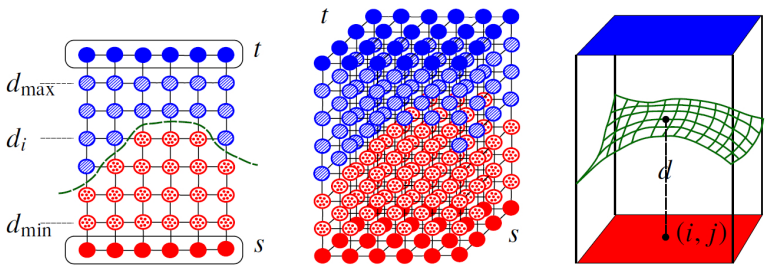


Fig. 3. A cut of one of the 3D graph's surfaces (left), the graphical graph's representation (middle) , the surface obtained in the result of cutting the graph (right) [50].

The α-expansion algorithm for every iteration in every label $l \in \zeta$ finds the s/t cut that separates the pixels which an label is attributed to l and the other ones. If the new cut lowered the cost function's value, it is accepted and a subset of the pixels obtain a new label. New labeling which did not lower value of global cost function are discarded. Stop condition is no change of the cost function. The result of the algorithm is such an s/t cut of G graph that all the set S elements which edge was cut are wanted set of labels (Fig. 3 right). The defined algorithm is only one of the possible variants of using the minimal cut described in the literature. The examples of the disparity map, obtained by using the graphs cut method can be see in the Fig. 4b and c.

The result of the matching algorithms is a two-dimensional disparity function called a disparity map. In most cases algorithms fail to get the full map (some of the pixels do not have an attributed value of disparity) or noise level in disparity function is high. One of the popular methods to improve the quality of obtained results is to apply filters which reduces noise, such as a bilateral filter [22].

2.1.4. Triangulation

The last stage of stereo-vision is to calculate the 3D coordinates of the scene's points.The metrical reconstruction of the approximated shape of the scene is possible only for a calibrated set of video cameras. For the most popular in the literature set of side video cameras there is a simple triangulation algorithm for the 3D points' location. The disparity map may be defined as a set of ordered triple $\langle x, y, d \rangle$, where x, y are pixel's coordinates of the referential image and d is disparity value. The corresponding pixels in the second image are located on the same line as in the first image $y' = y$. This is why the coordinate of the pixel in the second image equals $x' = x + d$. By entering a designation for the central point x_0, the focal distance f, the base distance B, a geometrical stereoscopic set my be presented Fig. 5

Using rule of similarity of triangles it is possible to derive formulas which enable to

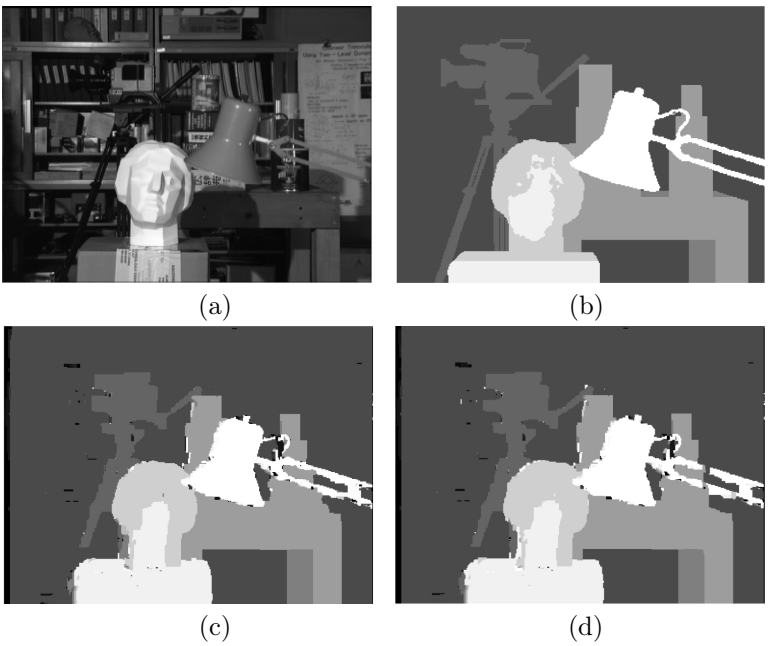


Fig. 4. One of the two stereo-vision images (a), ground truth disparity (b), disparity map obtained by $\alpha\beta$-swap (c), disparity map obtained by α-expansion (d) [32].

calculate the 3D point's coordinates:

$$X = \frac{B(x + x' - 2x_0)}{2d}, Y = \frac{B \cdot y}{d}, Z = \frac{B \cdot f}{d} \tag{8}$$

The stereo-vision is a fast and universal method to obtain information about the 3D structure of the scene. Unfortunately it has faults which limit its range of application. First of all the requirement of image acquisition, using a special set of video cameras. The second significant fault is the need of highly intensive scene texturing. When the scene's colour is homogeneous with a small amount of characteristic points then no matter which algorithm of matching images will be used, the disparity map's quality is usually low.

2.2. Shape from motion

A group of methods which enable to reconstruct the scene's 3D shape with simultaneous calculating the camera's location are called "shape from motion" (SfM) or "visual simultaneous localisation and mapping". As the names of these methods suggest that they

work on the basis of images obtained from different video camera positions. All the SfM methods are based on the analysis of relations between the position of pixels, rotation, translation of camera and the position of points on the 3D surface. SfM methods form a numerous group which can be divided into a few categories, depending on:

- the number of simultaneously processed images of the scene
  - two images methods
  - three images methods
  - multiple images methods
- camera model
  - orthogonal
  - quasi-perspective
  - perspective
- the type of the scene
  - subject to rules of rigid bodies' movement
  - not subject to rules of rigid bodies' movement
- analyzed objects of the image
  - points
  - lines
- methods of analyzing data

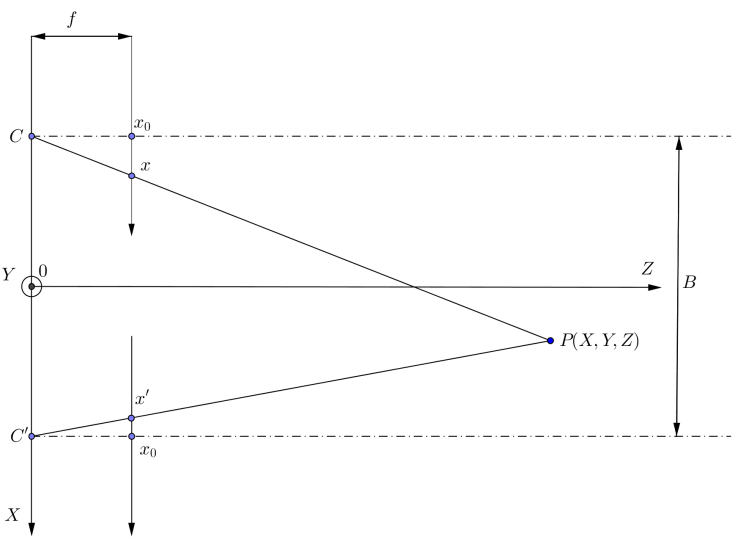


Fig. 5. Geometrical canonical model of the stereo-vision set [52].

∘ deterministic
∘ probabilistic

We will explain the three main methods, which are the basis for other methods or which are used for practical implementations. The first one is based on the results of work done by Kruppa [1], Longuet-Higgins [4] and Hartey [13, 44]. The first author proved that when you have two different images of five distinguishable 3D points, it is possible to calculate the position of cameras and 3D points (with unknown scale factors). Longuet-Higgins's work described the algorithm of calculation a essential matrix and its decomposition into rotation matrix and translation vector. Harley developed the method by adding operations which improve the algorithm's numerical stability and also he proposed triangulation methods. With the assumption that the internal parameters $K, K' \in \mathbb{R}^{3\times3}$ of cameras are known, it is possible to write down the reconstruction algorithm. Characteristic points of pictures as well as the corresponding pixel pairs are found for each of the two images $M = \left\{\langle p_i, {p_i}'\rangle \in P \times P' : \|d(x_i) - d({x_i}')\| \leqslant \xi, i = \overline{1,n}\right\}$. The function $d(\cdot)$ depends on local features descriptor. The constant ξ is threshold. The algorithm in the presented form will work if $n \geqslant 8$ (eigth-point algorithm). All the pixel coordinates are transformed into homogenic normalised coordinates.

$$\hat{M} = \left\{ \begin{array}{l} \langle \hat{p}_i, \hat{p}'_i\rangle \in \mathbb{P}^2(\mathbb{R}) \times \mathbb{P}^2(\mathbb{R}) : \\ \hat{p}_i = K^{-1}[{p_i}^T, 1]^T, \hat{p}'_i = (K')^{-1}[{p'_i}^T, 1]^T, \langle p_i, p'_i\rangle \in M, i = \overline{1,n} \end{array} \right\} \tag{9}$$

Shifting and rescaling the coordinates in order to position the centroid in the middle of the set of coordinates and also adjusting the variation to one, improves the stability of the numerical algorithm. For $\hat{M} \ni \langle \hat{p}_i, \hat{p}'_i\rangle = \left\langle [\hat{x}_i, \hat{y}_i, 1]^T, [\hat{x}'_i, \hat{y}'_i, 1]^T \right\rangle$ new coordinates are:

$$\begin{array}{l} \tilde{x}_i = s(\hat{x}_i - \mu_x), \tilde{y}_i = s(\hat{y}_i - \mu_y) \\ \tilde{x}'_i = s(\hat{x}'_i - \mu'_x), \tilde{y}_i = s(\hat{y}'_i - \mu'_y) \end{array} \tag{10}$$

It can be written in a form of a matrix:

$$\tilde{p}_i = \begin{bmatrix} \tilde{x}_i \\ \tilde{y}_i \\ 1 \end{bmatrix} = \begin{bmatrix} s & 0 & -s\cdot\mu_x \\ 0 & s & -s\cdot\mu_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{x}_i \\ \hat{y}_i \\ 1 \end{bmatrix} = N\hat{p}_i \text{ and by analogy } \tilde{\mathrm{p}}'_{\mathrm{i}} = \mathrm{N}'\hat{\mathrm{p}}'_{\mathrm{i}} \tag{11}$$

$s, s', \mu_x, \mu'_x, \mu_y, \mu'_x$ are chosen, so that the equations (12)(13) are satisfied:

$$\sum\nolimits_i \tilde{x}_i = \sum\nolimits_i \tilde{x}'_i = \sum\nolimits_i \tilde{y}_i = \sum\nolimits_i \tilde{y}'_i = 0 \tag{12}$$

$$\sum\nolimits_i \tilde{x}_i^2 + \sum\nolimits_i \tilde{y}_i^2 = \sum\nolimits_i (\tilde{x}'_i)^2 + \sum\nolimits_i (\tilde{y}'_i)^2 = 2n \text{ where } \mathrm{n} = \left|\hat{\mathrm{M}}\right| \tag{13}$$

hence

$$\tilde{M} = \left\{ \begin{array}{l} \langle \tilde{p}_i, \tilde{p}'_i \rangle \in \mathbb{P}^2(\mathbb{R}) \times \mathbb{P}^2(\mathbb{R}) : \\ \tilde{p}_i = N\hat{p}_i, \tilde{p}'_i = N'\hat{p}'_i, \langle \hat{p}_i, \hat{p}'_i \rangle \in \hat{M}, i = \overline{1,n} \end{array} \right\} \tag{14}$$

On the basis of the set $\tilde{M}$ we may approximate the essential matrix, using its features, that $\forall \langle \tilde{p}, \tilde{p}' \rangle \in \tilde{M}$ has to occur $(\tilde{p}')^T \tilde{E} \tilde{p} = 0$, a system of equations may be built in which every line takes a form:

$$\tilde{\mathrm{x}}_i\tilde{\mathrm{x}}_i'\tilde{\mathrm{e}}_{00}+\tilde{\mathrm{x}}_i'\tilde{\mathrm{y}}_i\tilde{\mathrm{e}}_{01}+\tilde{\mathrm{x}}_i'\tilde{\mathrm{e}}_{02}+\tilde{\mathrm{x}}_i\tilde{\mathrm{y}}_i'\tilde{\mathrm{e}}_{10}+\tilde{\mathrm{y}}_i\tilde{\mathrm{y}}_i'\tilde{\mathrm{e}}_{11}+\tilde{\mathrm{y}}_i'\tilde{\mathrm{e}}_{12}+\tilde{\mathrm{x}}_i\tilde{\mathrm{e}}_{20}+\tilde{\mathrm{y}}_i\tilde{\mathrm{e}}_{21}+\tilde{\mathrm{e}}_{22}=0 \tag{15}$$

for $\left\langle [\tilde{x}_i, \tilde{y}_i, 1]^T, [\tilde{x}'_i, \tilde{y}'_i, 1]^T \right\rangle \in \tilde{M}, i = \overline{1,n}$

After converting to the matrix form:

$$A\tilde{e} = 0 \tag{16}$$

where:
$A \in \mathbb{R}^{n \times 9}$ – matrix which only depends on the pixels coordinates
$\tilde{e} = [\tilde{e}_{00}, \tilde{e}_{01}, \tilde{e}_{02}, \tilde{e}_{10}, \tilde{e}_{11}, \tilde{e}_{12}, \tilde{e}_{20}, \tilde{e}_{21}, \tilde{e}_{22}]^T$ – vector built on the basis of the aproximated essential matrix's elements

The elements of the matrix A kernel are the the vectors $\tilde{e}$. The result is obtained by solving the linear equation of the least squares using SVD decomposition. Because scale's factor is unknown solution that fulfills constraint $\|\tilde{e}\| = 1$ is chosen.

The approximated essential matrix can be obtained using the transformation of initial normalized coordinates.

$$\hat{E} = (N')^T \tilde{E} N \tag{17}$$

Next step is SVD decomposition of E

$$\hat{E} = U \Sigma V^T \tag{18}$$

where:
$\Sigma = diag(\lambda_1, \lambda_2, \lambda_3),\ \lambda_1 \geq \lambda_2 \geq \lambda_3,\ \lambda_1, \lambda_2, \lambda_3 \in \mathbb{R}$
$U, V \in SO(3) = \left\{ R \in \mathbb{R}^{3\times 3} : R^T R = I, \det(R) = 1 \right\}$

According to Huang's and Faugeras' theorem, a essential matrix has to have such an SVD decomposition that $\Sigma = diag(\sigma, \sigma, 0)$. and $U, V \in SO(3)$. To satisfy those requirements an approximated essential matrix is projected on nearest E matrix from essential matrix space. Due to the fact that a matrix may be approximated up to the scale's factor $E = U diag\{1, 1, 0\} V^T$

From known matrix E it is possible to calculate rotation, translation and projection matrices. There are four alternative solutions for a SVD in form $E = U\Sigma V^T = U diag\{1, 1, 0\} V^T = [T]_\times R$:

$$\begin{array}{c} R\left(\pm\frac{\pi}{2}\right) = UR_Z^T\left(\pm\frac{\pi}{2}\right)V^T \\ [T]_\times\left(\pm\frac{\pi}{2}\right) = UR_Z\left(\pm\frac{\pi}{2}\right)\Sigma U^T \end{array} \tag{19}$$

where:

$$R_Z\left(\pm\frac{\pi}{2}\right) = \begin{bmatrix} 0 & \mp 1 & 0 \\ \pm 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

For each alternative solution we may build a rotation (20), translation (21), and projection (22) matrices in reference to external coordinates system.

$$R_\pm = R\left(\pm\frac{\pi}{2}\right)R^{(t-1)} \tag{20}$$

$$T_\pm = T^{(t-1)} - R^T\left(\pm\frac{\pi}{2}\right)T\left(\pm\frac{\pi}{2}\right) \tag{21}$$

$$P_{\pm\pm} = \begin{bmatrix} R_\pm & -R_\pm T_\pm \end{bmatrix} \tag{22}$$

where:

$R^{(t-1)} \in SO(3)$ – camera's rotation matrix at frame $t-1$ from previous iteration step (for first frame $R^{(0)} = I$)

$T^{(t-1)} \in \mathbb{R}^3$ – camera's translation vector at frame $t-1$ from previous iteration step (for first frame $T^{(0)} = 0$)

$T_{(\cdot)} \in \mathbb{R}^3$ – translation vector built from the elements of the matrix $[T]_\times$

Projection matrix at frame $t-1$:

$$P^{(t-1)} = \begin{bmatrix} R^{(t-1)} & -R^{(t-1)}T^{(t-1)} \end{bmatrix} \tag{23}$$

For every variant of the camera's location and orientation a triangulation of 3D points' coordinates is performed. By using the relations $x_i = P^{t-1}X_i$, $x_i' = P_{\pm\pm}X_i$ and $x_i \times PX_i = 0$ for $\hat{M} \ni \langle \hat{p}_i, \hat{p}_i' \rangle = \left\langle [\hat{x}_i, \hat{y}_i, 1]^T, [\hat{x}_i', \hat{y}_i', 1]^T \right\rangle$ a system of equations (24) is built

$$A_{\pm\pm i}X_i = 0 \tag{24}$$

where:

$$A_{\pm\pm i} = \begin{bmatrix} \hat{x}_i p^{3T} - p^{1T} \\ \hat{y}_i p^{3T} - p^{2T} \\ \hat{x}_i' p'^{3T} - p'^{1T} \\ \hat{y}_i' p'^{3T} - p'^{2T} \end{bmatrix}$$

p^1, p^2, p^3 – rows of matrix $P^{(t-1)}$

p'^1, p'^2, p'^3 – rows of matrix $P_{(\cdot)}$

The system of equations is calculated using least squares method by implementating the SVD decomposition with $\|X_j\| = 1$. Triangulation result for every element of $\hat{M}$ is the set of 3D points' coordinates:

$$S_{\pm\pm} = \left\{ X_i \in \mathbb{P}^3(\mathbb{R}) : A_{\pm\pm i} X_i = 0, i = \overline{1,n} \right\} \tag{25}$$

The set of four alternative results has the following form:

$$Q = \{ \langle R_-, T_-, S_{--} \rangle, \langle R_-, T_+, S_{-+} \rangle, \langle R_+, T_-, S_{+-} \rangle, \langle R_+, T_+, S_{++} \rangle \} \tag{26}$$

Function $f : Q \to \mathbb{N}$ which describes each element of Q as number of points located in front of cameras image plane, allows to select one result from four. Element from the set Q is selected for which the number of 3D points with positive value of coordinates on the Z axis:

$$\langle R^*, T^*, S^* \rangle = \underset{\langle R,T,S \rangle \in Q}{\arg\max} \left(f(\langle R, T, S \rangle) \right) \tag{27}$$

Finally:

$$R^{(t)} = R^*, T^{(t)} = T^*, S^{(t)} = S^{(t-1)} \cup S^* \tag{28}$$

The described algorithm requires having knowledge of the camera's internal parameters. If the parameters are unknown, the obtained is only projective reconstruction. The problem is solved by extending the SfM method using autocalibration algorithms, that is, the methods of automatically calculating internal parameters without a calibration pattern. The autocalibration methods are a widely described in the literature (a review may be found in [45]). Another fault of the presented algorithm is that in the result one may obtain a set of points of rather low cardinality. An improvement in reconstruction was gained by using the methods of matching images and triangulation used in stereovision [29]. Scheme of algorithms pipeline with autocalibration and dense matching can be seen in Fig. 6.

An alternative solution could be to calculate the position changes for all the pixels in both images (an optical flow). When we know the optical flow vector field, camera internal parameters and the fundamental matrix we may aproximate dense scene's shape. The main problem is to calculate the vector field of the optical flow. The main trend of research in this field is to use variation methods [35, 48, 63, 80] (a more extended review may be found in [59]). An example of dense scene reconstruction which may be obtained on the basis of an optical flow and epipolar geometry rules can be seen in Fig. 7.

Characteristic for the described two-image algorithms are such that with increasing number of frames positioning error rises. It is a result of propagating errors in every iteration of the algorithm. A considerable error reduction can be achieved by using the three-image methods (tensor based methods), however, it does not eliminate the problem of error propagation. A better solution is to use multi-image methods, so called, factorisation methods.

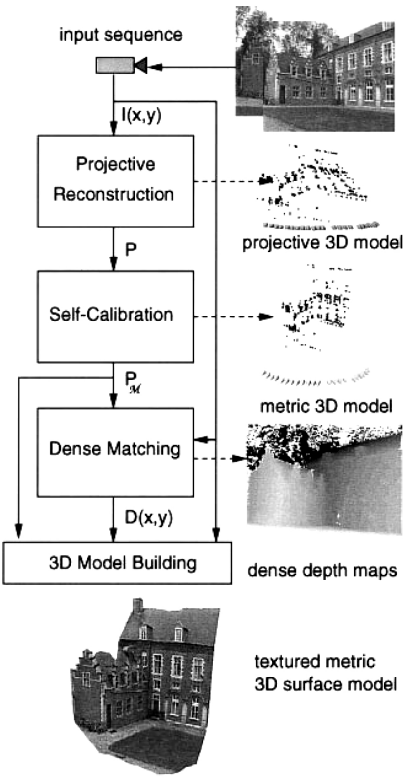


Fig. 6. A pipeline of 3D reconstruction algorithms, using a two-image SfM method, camera autocalibration and dense matching of images [29].

2.2.1. Factorisation

The first SfM algorithm which allowed to calculate the location of the camera and scene's structure by simultaneously analyzing all of the available scene's images was proposed by Tomasi and Kanade [8, 7]. The fulfillment of four assumptions are crucial in order to algorithm work correctly:

- the distance between the camera and the scene's objects is large, which allows to use an affine camera model with parallel projection (an orthogonal camera model)
- absence of pixels screw
- fixed and known internal parameters of camera
- all of the tracked points have to be visible in the whole sequence of images

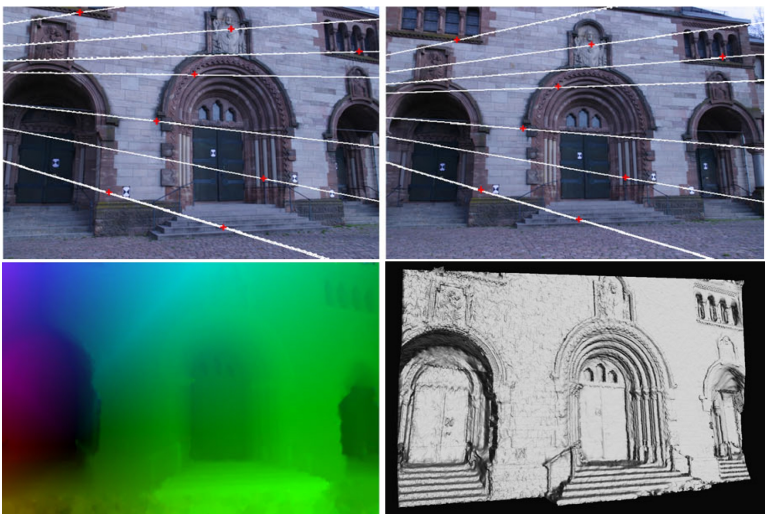


Fig. 7. An example of using two-image algorithm of dense reconstruction on the basis of an optical flow. At the top – a pair of input images with shown epipolar lines. At the bottom – a vector field of optical flow (left) and a dense reconstruction of the scene's shape (right) [84].

The orthogonal camera model without a pixel's screw:

$$M = \begin{bmatrix} \alpha & 0 & o_x \\ 0 & \beta & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_1^T & -r_1^T T \\ r_2^T & -r_2^T T \\ 0 & 1 \end{bmatrix} \tag{29}$$

where:

$M \in \mathbb{R}^{3\times 4}$	–	camera projection matrix
$r_1, r_2 \in \mathbb{R}^3$	–	rows of rotation matrix
$T \in \mathbb{R}^3$	–	camera translation vector
$\alpha, \beta \in \mathbb{R}$	–	pixel scaling factors
$o_x, o_y \in \mathbb{R}$	–	offsets of pixel coordinates system center

The point projection on the orthogonal cameras image plane:

$$p = \begin{bmatrix} \alpha & 0 \\ 0 & \beta \end{bmatrix} \left(\begin{bmatrix} r_1^T \\ r_2^T \end{bmatrix} P + \begin{bmatrix} -r_1^T T \\ -r_2^T T \end{bmatrix} \right) + \begin{bmatrix} o_x \\ o_y \end{bmatrix} \tag{30}$$

where:

$P \in \mathbb{R}^3$	–	3D point coordinates
$p \in \mathbb{R}^2$	–	pixel's coordinates

When the internal parameters of the camera are known, the point coordinates can be converted from the pixel coordinates into the image plane (normalized coordinates):

$$p^{'} = \left(p - \begin{bmatrix} o_x \\ o_y \end{bmatrix} \right) \begin{bmatrix} 1/\alpha & 0 \\ 0 & 1/\beta \end{bmatrix} \tag{31}$$

Due to the fact that many images and points are analyzed, we use indexation:

$$p_i^{(j)} = \begin{bmatrix} x_i^{(j)} & y_i^{(j)} \end{bmatrix}^T \tag{32}$$

where:
$p \in \mathbb{R}^2$ – normalized pixel's coordinates
$j \in \{1, 2, .., m\}$ – image number
$i \in \{1, 2, .., n\}$ – point number

Taking into consideration the change of coordinates and indexation we obtain:

$$p_i^{(j)} = \begin{bmatrix} r_1^{(j)T} \\ r_2^{(j)T} \end{bmatrix} P_i + \begin{bmatrix} -{r_1}^{(j)T} T_i^{(j)} \\ -{r_2}^{(j)T} T_i^{(j)} \end{bmatrix} \tag{33}$$

Putting together the characteristic points coordinates and the equations of their projections into one matrix:

$$\begin{bmatrix} p_1^{(j)} & ... & p_n^{(j)} \end{bmatrix} = \begin{bmatrix} r_1^{(j)T} \\ r_2^{(j)T} \end{bmatrix} \begin{bmatrix} P_1 & ... & P_n \end{bmatrix}_i + \begin{bmatrix} -{r_1}^{(j)T} T_i^{(j)} \\ -{r_2}^{(j)T} T_i^{(j)} \end{bmatrix} \tag{34}$$

Subsequently, we calculate all of the points' coordinates in a matrix in such a way so that the center of coordinates system is the centroid:

$$\bar{p} = \begin{bmatrix} r_1^{(j)T} \\ r_2^{(j)T} \end{bmatrix} \bar{P} + \begin{bmatrix} -{r_1}^{(j)T} T_i^{(j)} \\ -{r_2}^{(j)T} T_i^{(j)} \end{bmatrix} \tag{35}$$

where:
$\bar{p}$ – centroid's coordinates in the image plane
$\bar{P}$ – centroid's 3D coordinates in the external coordinates system

The centroid for the points in the image plane may be calculated using the formula:

$$\bar{p}^{(j)} = \frac{\sum_{i=1}^{n} p_i^{(j)}}{n} \tag{36}$$

The equation gets the following form:

$$\begin{bmatrix} p_1^{(j)} - \bar{p}^{(j)} & ... & p_M^{(j)} - \bar{p}^{(j)} \end{bmatrix} = \begin{bmatrix} r_1^{(j)T} \\ r_2^{(j)T} \end{bmatrix} \begin{bmatrix} P_1 - \bar{P} & ... & P_M - \bar{P} \end{bmatrix} \tag{37}$$

For n points in m images we build a registered measurement matrix:

$$W = \begin{bmatrix} p_1^{(1)} - \bar{p}^{(1)} & ... & p_n^{(1)} - \bar{p}^{(1)} \\ & ... & \\ p_1^{(m)} - \bar{p}^{(m)} & ... & p_n^{(m)} - \bar{p}^{(m)} \end{bmatrix} = \begin{bmatrix} r_1^{(1)T} \\ r_2^{(1)T} \\ ... \\ r_1^{(m)T} \\ r_2^{(m)T} \end{bmatrix} \begin{bmatrix} P_1 - \bar{P} & ... & P_n - \bar{P} \end{bmatrix} = RS \tag{38}$$

The authors of the method formulated a theorem that if there isn't any noise in measurements of points position then the matrix of registered measurements W has rank not greater than three. This is due to the fact that the ranks of the rotation matrix R and the structure matrix S is not greater than three. The presence of noise in the matrix W, built on the basis of real images, could cause the rank of W to be greater than three. The matrix rank reduction to three may be done by decomposing the matrix on the basis of Singular Value Decomposition.

$$W = U\Sigma V^T \tag{39}$$

where:
$W \in \mathbb{R}^{2m \times n}$ – registered measurements' matrix

$$\Sigma = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & .. & \\ & & & \lambda_n \\ & 0 & & \end{bmatrix}, \lambda_1 \geqslant \lambda_2 \geqslant ... \geqslant \lambda_n, \lambda_1, ..., \lambda_n \in \mathbb{R}$$

$U \in \mathbb{R}^{2m \times 2m}$,$V \in \mathbb{R}^{n \times n}$ – orthonormal matrices

From all values, the first three (the largest) are chosen with the corresponding left and right vectors and on their basis, a matrix is built which rank value is not greater than three.

$$W^{'} = \begin{bmatrix} U_1 & U_2 & U_3 \end{bmatrix} \begin{bmatrix} \lambda_1 & & \\ & \lambda_2 & \\ & & \lambda_3 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \\ V_3^T \end{bmatrix} = U^{'}\Sigma^{'}V^{'T} \tag{40}$$

We may factorize the W matrix into a camera rotation matrix and a scene structure matrix

$$W^{'} = R^{'}S^{'} \tag{41}$$

where:
$R^{'} = U^{'}\Sigma^{'}$
$S^{'} = V^{'T}$

The matrix $R^{'}$ should be a rotation matrix; unfortunately, the solution above guarantees only that the matrix will be a linear transformation of the rotation matrix, because there is a matrix which $R^{'}S^{'} = R^{'}QQ^{-1}S^{'} = \left(R^{'}Q\right)\left(Q^{-1}S^{'}\right) = \hat{R}\hat{S}$. Due to this fact to calculate the rotation matrix which makes an Euclidean transformation, it is necessary to calculate the correcting matrix. Since the matrix $\hat{R}$ has to be orthonormal, we can define the following system of equations:

$$\begin{aligned} \hat{r}_{2j-1}^{T}\hat{r}_{2j-1} = 1 = r_{2j-1}^{'T}QQ^{T}r_{2j-1}^{'} \\ \hat{r}_{2j}^{T}\hat{r}_{2j} = 1 = r_{2j}^{'T}QQ^{T}r_{2j}^{'} \\ \hat{r}_{2j}^{T}\hat{r}_{2j-1} = 0 = r_{2j}^{'T}QQ^{T}r_{2j-1}^{'} \end{aligned} \tag{42}$$

where:

j	=	$\overline{1,m}$
$\hat{r}_k$	–	k-th row of the matrix $\hat{R}$
$r_k^{'}$	–	k-th row of the matrix $R^{'}$

The matrix Q can be found by replacing matrix QQ^T with B and using Cholesky-Banachiewicz factorization. Non-linear method of least squares also could be used. After calculating the correcting matrix we obtain an Euclidean reconstruction and a rotation matrix. The solution is ambiguous because every initial matrix R_0 fulfills the equations:

$$\begin{aligned} \hat{R}^{'} = \hat{R}R_0 \\ \hat{S}^{'} = R_0^{T}\hat{S} \end{aligned} \tag{43}$$

It is arbitrarily selected that the initial rotation equals 0, so the first position of camera determine center and orientation of the coordinates system. Therefore:

$$\begin{bmatrix} \hat{r}_1^{'(1)T} \\ \hat{r}_2^{'(1)T} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{44}$$

$$R_0 = \begin{bmatrix} \hat{r}_1^{'(1)T} & \hat{r}_2^{'(1)T} & \hat{r}_1^{'(1)T} \times \hat{r}_2^{'(1)T} \end{bmatrix} \tag{45}$$

Finally the results are:

$$\begin{aligned} \hat{S}^{'} &= R_0^{T}\hat{S} \\ R^{(j)} &= \begin{bmatrix} \hat{r}_1^{'(j)} & \hat{r}_2^{'(j)} & \hat{r}_3^{'(j)} \end{bmatrix} \\ T^{(j)} &= \begin{bmatrix} \hat{r}_1^{'(j)} & \hat{r}_2^{'(j)} & \hat{r}_3^{'(j)} \end{bmatrix} \begin{bmatrix} \bar{p}^{(j)} \\ \alpha \end{bmatrix} \end{aligned} \tag{46}$$

where:

$\hat{S}'$	–	the matrix of 3D points' coordinates in the centroid coordinates system
$R^{(j)}$	–	camera's rotation matrix for image j, where $j = \overline{1,m}$
$T^{(j)}$	–	camera's translation vector for image j, where $j = \overline{1,m}$
$\alpha \in \mathbb{R}$	–	arbitrarily selected number

Results of the factorisation algorithm are shown in Fig. 8

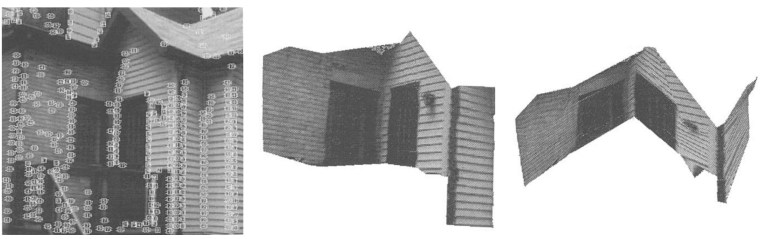


Fig. 8. The final results of the factorization algorithm's, (b) and (c) images show artificially generated 3D models, obtained on the basis of a sequence (a) of input images [8].

The algorithm in the presented form is much less complex than the two-image algorithm described before. The simplicity of this algorithm lies, however, in considerable simplification of camera model. The inadequacy problem about the whole range of the scene's objects distance has been solved in the next modifications of the method, subsequently introducing the following models: the paraperspective [9] and the perspective one [16, 17]. The number of 3D points which location can be determined considerably decreases during longer sequences, because it is required to possess the knowledge of the location of all the characteristic points in the entire sequence. There has been an attempt to solve the problem by approximating the points' location on the basis of the neighbouring points' location [8]. The factorization algorithm allows to make a 3D scene reconstruction and determine the location of cameras without the errors accumulation, however, as every linear method it is not greatly resistant to the errors resulting from the spatial image discretization and incorrect point matching. Unfortunately, on the grounds of the necessity of SVD of the huge matrix of registered measurements and the necessity to monitor all the points is not the optimal solution in most practical applications. The state of art in the field of accurate multi-image sparse SfM methods is the bundle adjustment algorithm. The method is base on a non-linear optimization method set in order to find a minimal error in adjusting the parameters of the 3D world model and cameras model to observations.

2.2.2. Bundle adjustment

The bundle adjustment is an iterative method of optimization, which minimizes the cost function in the shape of a square error between the observations and the theoretical camera and the 3D world models. Bundle adjustment is about using specific for SfM assumptions in order to effectively solve the optimization task. In contrast to the previously described methods, SfM does not introduce any limitations concerning the number or visibility of characteristic points. Below, we will describe an algorithm which requires known and fixed internal camera's parameters. By using bundle adjustment its possible to calculate all of the internal and external camera's parameters, however, in such a case, solutions space has more dimensions, which considerably reduces the speed of processing data by the algorithm.

The algorithm's input data:
$p_i^{(j)} \in \mathbb{R}^2$ – coordinates of i-th point in j-th image, where $j \in \{1,..,n\}, i \in \{1,..,m\}$
$z \in \mathbb{R}^5$ – vector of constant and known internal camera parameters

The theoretical pixel coordinates of characteristic points may be calculated using the following algorithm:

$$\begin{gathered} P_i \longmapsto \tilde{P}_i \\ \tilde{p}_i^{(j)} = M(z, C_j)\tilde{P}_i \\ \tilde{p}_i^{(j)} \longmapsto \bar{p}_i^{(j)} \end{gathered} \tag{47}$$

where:

$P_i \in \mathbb{R}^3$	–	Cartesian coordinates of i-th 3D point in external coordinates system
$\tilde{P}_i \in \mathbb{P}^3(\mathbb{R})$	–	homogeneous coordinates of i-th 3D point in external coordinates system
$\tilde{p}_i^{(j)} \in \mathbb{P}^2(\mathbb{R})$	–	expected homogeneous coordinates of i-th pixel corresponding to point P_i in j-th image
$\bar{p}_i^{(j)} \in \mathbb{R}^2$	–	expected Cartesian coordinates of i-th pixel corresponding to point P_i in j-th image
$C_j \in \mathbb{R}^6$	–	external camera parameters' vector (of rotation and translation) for j-th image
$M(z, C_j) \in \mathbb{R}^{3\times 4}$	–	camera projection matrix for j-th image

The algorithm above can be expressed as function $h : \mathbb{R}^6 \times \mathbb{R}^3 \to \mathbb{R}^2$ defined by $\bar{p}_i^{(j)} = h(C_j, P_i)$. When C_j, P_i are element of models parameters vector $x = [C_1^T, .., C_n^T, P_1^T, ..., P_m^T]^T \in \mathbb{R}^{6n+3m}$ mapping can be written as $h' : \mathbb{R}^{6n+3m} \times \{1,..,n\} \times \{1,..,m\} \to \mathbb{R}^2$ and $\bar{p}_i^{(j)}(x) = h'(x,i,j) = h(C_j, P_i)$. The error between the observation and the model can be defined as:

$$e_i^{(j)}(x) = p_i^{(j)} - \bar{p}_i^{(j)}(x) \tag{48}$$

Next, a global error function may be introduced:

$$f(x) = \sum_{j=1}^{n} \sum_{i \in W_j} \left(e_i^{(j)}(x) \right)^T e_i^{(j)}(x) \tag{49}$$

where:
$x = [C_1^T, .., C_n^T, P_1^T, ..., P_m^T]^T \in \mathbb{R}^{6n+3m}$ – vector of model's parameters
W_j – set of indexes of points seen in the i-image
The algorithm's result is the solution to the minimization task:

$$x^* = \arg\min_x f(x) \tag{50}$$

By linearization of error function in point x using Taylor's formula and omitting the rest of the second order and higher, we obtain:

$$e_i^{(j)}(x + \Delta x) \simeq e_i^{(j)}(x) + J_i^{(j)}(x)\Delta x \tag{51}$$

where:
$\Delta x \in \mathbb{R}^{6n+3m}$
$J_i^{(j)}(x) = \left[\frac{\partial e_i^{(j)}(x)}{\partial C_1}, ..., \frac{\partial e_i^{(j)}(x)}{\partial C_n}, \frac{\partial e_i^{(j)}(x)}{\partial P_1}, ..., \frac{\partial e_i^{(j)}(x)}{\partial P_m} \right] \in \mathbb{R}^{2\times(6n+3m)}$ – Jacobian

Due to the fact that partial derivatives are non-zero only for j-th camera and for j-th point (the error function depends on these vectors), the Jacobian matrix has the following form:

$$J_i^{(j)}(x) = \left[0, ..., 0, \frac{\partial e_i^{(j)}(x)}{\partial C_j}, 0, ..., 0, \frac{\partial e_i^{(j)}(x)}{\partial P_i}, 0..., 0 \right] \in \mathbb{R}^{2\times(6n+3m)} \tag{52}$$

The structure of the matrix created from all of the Jacobian matrices allows it to be split into two parts, one dependent only on parameters of the cameras and the second one on 3D points' coordinates.

$$J(x) = [J_C(x) | J_P(x)] \in \mathbb{R}^{2nm\times(6n+3m)} \tag{53}$$

The iterative algorithm of searching for the global minimum of error function depends on calculating a direction in which the error function decreases the fastest $\Delta x^* = \arg\min_{\Delta x} \|f(x) + J\Delta x\|^2$. In every iteration the vector is updated $x \leftarrow x + \Delta x^*$ if $f(x + \Delta x^*) < f(x)$. The stop condition is when the change in the values of the function is lower than the threshold $\|f(x) - f(x + \Delta x^*)\| < \xi$. Levenberg and Marquardt have developed the method of control the step length $\|\Delta x^*\|$ in every iteration, thanks to which, the number of iterations decreased considerably. In Levenberg and Marquardt's

method, the length of steps is controlled by element $\lambda\|D(x)\Delta x\|^2$, which is calculated for every iteration, therefore:

$$\Delta x^* = \arg\min_{\Delta x} \|f(x) + J(x)\,\Delta x\|^2 + \lambda\|D(x)\,\Delta x\|^2 \tag{54}$$

where:
$D(x) \in \mathbb{R}^{(6n+3m)\times(6n+3m)}$ – positive-semidefinite diagonal matrix
$\lambda \in \mathbb{R}$ – nonnegative coefficient

The optimal iterative step is the solution to the system of linear equations (the dependency from x has been omitted in order to simplify the notation):

$$\left(J^T J + \lambda D^T D\right)\Delta x = -J^T f \tag{55}$$

Basing on the known structure of the Jacobian matrix $J = [J_C|J_P]$ we may write down the equation in an alternative way:

$$\begin{bmatrix} H_{\lambda CC} & J_C^T J_P \\ J_P^T J_C & H_{\lambda PP} \end{bmatrix} \begin{bmatrix} \Delta x_c \\ \Delta x_P \end{bmatrix} = \begin{bmatrix} -J_C^T f \\ -J_P^T f \end{bmatrix} \tag{56}$$

where:
$H_{\lambda CC} = J_C^T J_C + \lambda D_C^T D_C$ – block-diagonal autocorrelation matrix of cameras
$H_{\lambda PP} = J_P^T J_P + \lambda D_P^T D_P$ – block-diagonal autocorrelation matrix of points

We may use the untypical structure of Hessian matrix in which the biggest element $H_{\lambda PP} \in \mathbb{R}^{n\times n}$ is the block-diagonal matrix with block size 3×3. Using Schur complement we obtain:

$$\begin{gathered} \left(H_{\lambda CC} - J_C^T J_P H_{\lambda PP}^{-1} J_P^T J_C\right)\Delta x_C = -J_C^T f + J_C^T J_P H_{\lambda PP}^{-1} J_P^T f \\ \Delta x_P = -H_{\lambda PP}^{-1}\left(J_P^T f + J_P^T J_C \Delta x_C\right) \end{gathered} \tag{57}$$

Block-diagonal structure of matrix allows to use algorithm of matrix inverse with linear computational complexity. Thanks to that we may calculate the vectors $-J_C^T f + J_C^T J_P H_{\lambda PP}^{-1} J_P^T f$, $-H_{\lambda PP}^{-1}\left(J_P^T f + J_P^T J_C \Delta x_C\right)$ and matrix $\left(H_{\lambda CC} - J_C^T J_P H_{\lambda PP}^{-1} J_P^T J_C\right)$ in an effective way. Unfortunately last matrix is an element of system of equations which solution is Δx_C. For a large number of images the matrix's size is large and typical methods for solving system of linear equations (for example, Cholesky-Banachiewicz decomposition) take too much time. The solution to this problem is using the iterative method of conjugated gradients. The method of conjugated gradients is based on the assumption that the solution to every system of linear equations $Ax = b$ can be presented in a form of a sum $x = \sum_{i=1}^{k} \alpha_i \varepsilon_i$, while the vectors ε_i on which space of the solutions is expanded are conjugated with respect to matrix A. The initial conditioning of matrices aims to improve convergence and numerical stability of the algorithm. Multiplying both sides by the matrix S^{-1} we obtain $S^{-1}Ax = S^{-1}b$, however, condition $\kappa(S^{-1}A) < \kappa(A)$

has to be fulfilled. For the bundle adjustment method, because its ease to inverse $S = H_{\lambda CC}$. For initial values $r_0 = b$, $z_0 = S^{-1}r_0$, $\varepsilon_0 = z_0$ in each algorithm's iteration we calculate:

$$\begin{aligned} \alpha_k &= \frac{r_k^T z_k}{\varepsilon_k^T A \varepsilon_k} \\ r_{k+1} &= r_k - A\varepsilon_k \\ z_{k+1} &= S^{-1} r_{k+1} \\ \varepsilon_{k+1} &= z_{k+1} + \frac{z_{k+1}^T r_{k+1}}{z_k^T r_k} \varepsilon_k \end{aligned} \tag{58}$$

and the result:

$$x_{k+1} = x_k + \alpha_k \varepsilon_k \tag{59}$$

The stop condition is $r_{k+1} < \xi$ where ξ is an acceptable error threshold. Performance of the algorithm comes from the fact that multiplication can be done without storing the whole matrix in memory, because when performing the operations in a correct order we multiply only vectors by vectors. For bundle adjustment algorithm:

$$A\varepsilon_k = \left(H_{\lambda CC} - J_C^T J_P H_{\lambda PP}^{-1} J_P^T J_C\right) \varepsilon_k \tag{60}$$

This equation can be transformed by expanding $H_{\lambda CC}$ and putting the brackets which force a specific order of calculations:

$$A\varepsilon_k = J_C^T \left(J_C p_k - J_P \left(H_{\lambda PP}^{-1} \left(J_P^T \left(J_C \varepsilon_k\right)\right)\right)\right) + \lambda D_C^T D_C \varepsilon_k \tag{61}$$

Operation on vector elements can be done in parallel; this allows to develop high performance implementations for GPU [82]. The main disadvantage of the described algorithm is that it does not guarantee finding the global minimum for the cost function. The result of the algorithm is the first found local minimum. The bundle adjustment algorithm is usually initiated by the result of one of the described two-image, three-image or factorization methods, thanks to which the probability of finding the global minimum increases considerably. High accuracy results obtained by method of adjusting parameters was motivation to use this method in many scientific as well as commercial projects. Spectacular examples of using the described algorithm are "Photo tourism" [57] and "Building Rome in a day" [78]. The example of reconstruction (di Trevi's fountain) on the basis of a web collection of pictures can be seen in fig 9.

Unfortunately, like with other methods based on the characteristic points and local feature descriptors with bundle adjustment we obtain only a sparse cloud of points. The literature about SfM is very exhaustive. This article presents only the key ideas which were most intensively studied and developed by researchers. A completely different source of information about structure of 3D scene's objects is the shape of their contours. The analysis methods of objects' contours in order to reconstruct their 3D shape are called "shape from silhouette".

Fig. 9. Cloud of points "Fontanna di Trevi" obtained using the bundle adjustment algorithm [57].

2.3. Shape from silhouette

The group of methods which allow to obtain a 3D shape of an object on the basis of their profiles, gained from moving the camera is called "shape from silhouette"(SfS) or "shape from profiles". The development of methods which base on the analysis of the silhouette's shape began in 1974 with publication of Baumgart [2]. He proposed a method of constructing an 3D model as an intersection of polyhedrons generated by the object's silhouette. Subsequent works from this field have input to SfS paradigm the definition of visual hull. According to [11, 42] the visual hull H_j for a consistent set of silhouette images $\left\{S_j^{(k)} : k = \overline{1, K}\right\}$ is intersection of K visual cones in which every one of them is formed by projecting the silhouette S_j^k into a 3D space from the central point of the camera C^k through the image plane Π^k. There is also an alternative definition [36], according to which H_j it is a figure of the largest possible capacity, explaining the shape of all the S_j^k where $k = \overline{1, K}$. Formation principle of visual hull is shown in Fig. 10.

Respectively to the definition we may distinguish two main methods of constructing the visual hull:

- surface-based
- volume-based

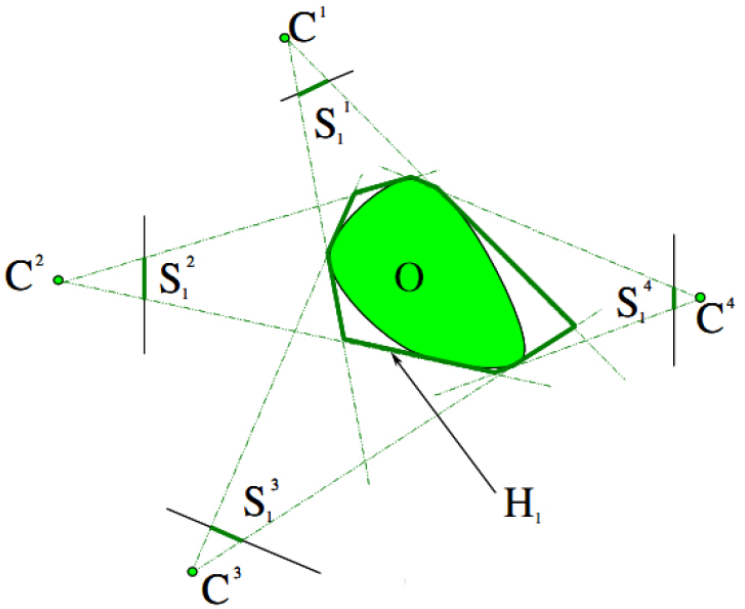


Fig. 10. The visual hull's shape is a result of intersecting the visual cones generated by the silhouettes and central points of cameras[42].

Regardless of the chosen method of constructing the visual hull, the error with which the real shape of the object approximates is highly dependent on the number of silhouettes. The reconstruction is possible when for every silhouette image a camera is calibrated. In most cases, the images of the objects are made without changing the internal camera's parameters. Therefore, it is assumed that the matrix of internal parameters is constant and known. The calibration of camera external parameters in case of SfS is aimed at calculating the rotation matrix and the translation vector in relation to the 3D object's coordinates system. Cameras' calibration methods depend on the used set of image acquisition in which we may determine:

- moving camera and static object
- static camera and the object placed on turntable
- object placed between static set of cameras

In case of a turntable, it is most commonly equipped with the calibration pattern facilitating precise calibration. When it comes to the static set of cameras, we use generally the disposable manual calibration procedure. When it comes to the mobile camera,

which moves along any trajectory around the object, we use the methods which are described in point "Shape from motion", mostly based on the epipolar geometry and bundle adjustment.

Another important SfS aspect for the final result is extraction of the silhouette. Separating the object's silhouette from the background is called segmentation and this problem is widely described in the literature and is too broad to be thoroughly explained in this article. The most frequent way to segment silhouettes is to use robust methods, such as active contours [5] and graph-cut [31, 73] (a review of different segmentation methods can be found in publications [85, 88, 89]). Segmentation is one of the most computationally expensive element of SfS. A particular case is an static set of cameras, because, in this case the background is constant. Automatic segmentation becomes then just background subtraction and binarization.

If we have a set of silhouettes and the corresponding internal and external cameras' parameteres, we may use one of the methods of visual hull construction.

2.3.1. Volume-based methods of visual hull construction

In terms of conception the most simple approach to SfS is subdivide of the 3D space on equivalence classes, called voxels and binary labeling each voxel. Labeling goal is to split voxels on those belonging to the visual hull and the other. The simplest volume-based SfS algorithm can be written as alg. 2.1.

The algorithm 2.1 is simple to implement, however, in general, it is rarely used. The problem lies in determining the minimal size of the voxel, because when there is a large number of voxels, the calculation time is unacceptable. When we enlarge the size of the voxel, the reconstruction quality drastically decreases, which is in part the result of the artifacts caused by aliasing. A sample reconstruction made using a method of filling the visual hull can be seen in Fig. 11

The basic improvement to this method is using octrees [10]. The structure of these

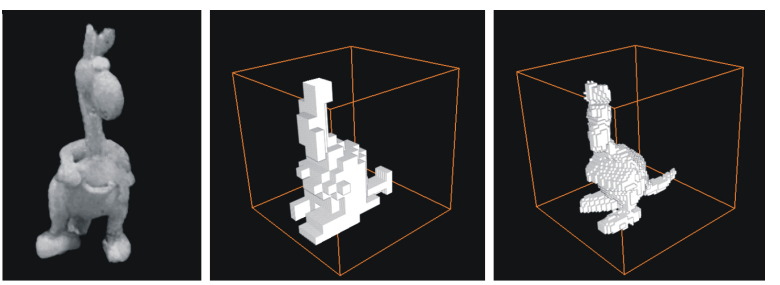


Fig. 11. A reconstruction of 3D object(a) obtained using a volume-based SfS (b)(c). The impact of the voxels' size on the quality of reconstruction is visible [33].

Algorithm 2.1 Volume-based SfS

1: Obtain set of silhouettes and camera's parameters
$\left\{\left\langle S^{(k)}, K^{(k)}, R^{(k)}, T^{(k)}\right\rangle \in \{0,1\}^{A\times B} \times \mathbb{R}^{3\times 3} \times SO(3) \times \mathbb{R}^3 : k = \overline{1,K}\right\}$
where:
$S^{(k)}$ – k-th silhouette image of sizes $A, B \in \mathbb{N}^+$
$K^{(k)}$ – internal camera parameters matrix for k-th silhouette
$R^{(k)}$ – camera rotation matrix for k-th silhouette
$T^{(k)}$ – camera translation vector for k-th silhouette
2: Subdivide 3D space into voxels of size $r[1,1,1]^T$,$r > 0$ hence $\vartheta = \{0,1\}^{X\times Y\times Z}$
where:
ϑ – voxels space
$X, Y, Z \in \mathbb{N}^+$ – size of voxels space
3: **for all** $\langle x, y, z\rangle \in \{1,..,X\} \times \{1,..,Y\} \times \{1,..,Z\}$ **do**
4: $\nu_{xyz} \leftarrow 1$ where $\nu_{xyz} \in \vartheta$ `/* initial value of voxel is 1 (full) */`
5: **for** $k \leftarrow 1$ **to** K **do**
6: Calculate the position of voxel's projection in k-th image (silhouette)

$$\begin{bmatrix} \alpha a \\ \alpha b \\ \alpha \end{bmatrix} = K^{(k)} \left[R^{(k)}| - R^{(k)}T^{(k)}\right] \begin{bmatrix} x - \frac{1}{2} \\ y - \frac{1}{2} \\ z - \frac{1}{2} \\ \frac{1}{r} \end{bmatrix}$$

7: **if** $< \lceil a \rceil, \lceil b \rceil > \notin \{1,..,A\} \times \{1,..,B\}$ **then** $\nu_{xyz} \leftarrow 0$ `/* out of image */`
8: **if** $S^{(k)} \ni s^{(k)}_{\lceil a \rceil \lceil b \rceil} = 0$ **then** $\nu_{xyz} \leftarrow 0$ `/* out of silhouette */`
9: **end for**
10: **end for**
11: **Result:** Visual Hull is created by all voxels $\nu \in \vartheta$ with assigned value (label) 1

trees allows size of voxels to be adjusted to the local number of details. The acceleration of the algorithm's operation is obtained using the fact that etiquetting voxels can be done completely parallely. Unfortunately, the Visual Hull (VH) representation, using voxels, does not allow one to generate photo-realistic images, which significantly limits its range of applications. The problem is solved by algorithms which transforms models from voxels into mesh or another form of object's surface representation (such as isosurface) [56, 66, 76]. An example of results of the the algorithm [66] can be seen in Fig. 12.

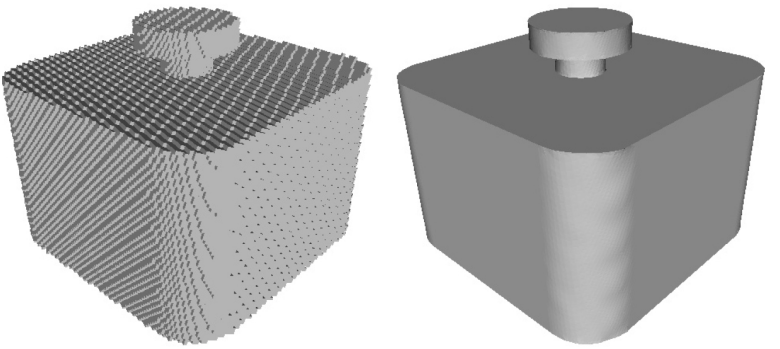


Fig. 12. The isosurface (on the right) generated on the basis of the model built from voxels (the object on the left) [66].

According to the basic definition, VH is an intersection of sets generated by silhouette's preimage in relation to the projection function. This is why an error in segmentation, even with just one silhouette, drastically affects the final result. An important field in the works on SfS is the improvement of the reconstruction quality by increasing the resistance to faulty segmentation. Most frequently used are probabilistic methods based on the analysis of cohesion with the image of neighbouring silhouettes [75, 83]. An important issue analyzed in publications [53, 55] is the poblem of partial object visibility and the possibility of using even a part of a silhouette as the information source about its 3D shape. An interesting field of the research is using probability theories in VH construction [43, 49, 86]. An example is the publication [49] in which an approach to the SfS problem is described as fussion of data from sensors. The authors proposed a probability sensor model known from robotics, corresponding to the set of video cameras in connection with SfS algorithm. The voxel's probability of belonging to VH is calculated as the conditional probability, dependent on the silhouettes, images (brightness, color

and pixels' color saturation) and the background images. Labeling is reduced to assigning label 1 for every voxel which has the probability of belonging to VH higher than the threshold. The effect of using the algorithm [49] can be seen in Fig. 13.

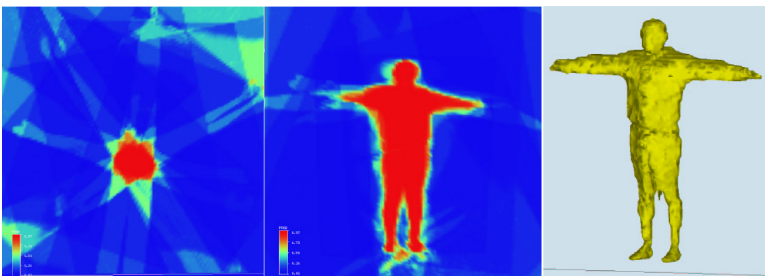


Fig. 13. The VH constrction on the basis of probability of voxel's occupancy, (a) – a view of one layer of the probability grid, (b) – a side view, (c) the visual hull obtained by binarizing the probability grid [49].

An alternative for methods based on building volume of the visual hull are surface-based methods of calculating its shape.

2.3.2. Surface-based methods

Surface-based SfS methods uses the information about contours of silhouette to calculate the approximated surface of VH. The surfaces are created from intersections of visual cones side surfaces (defined by the silhouettes outline and cameras central points). With regard to the way of calculating the intersections, we may introduce the division on:

- algorithms with calculating the intersection points in 3D
- algorithms with calculating the intersection points in 2D

The first group of methods derives from an idea proposed by Baumgart [2] of generating the visual hull's surface by calculating the primitives' intersections which form the visual cone in the 3D. Currently, one of the best algorithms regarding this field is the EPVH algorithm of Franko and Boyer [71]. The authors assume that the algorithm's input set is the set of silhouettes obtained from the calibrated cameras:

$$\left\{\left\langle S^{(k)}, K^{(k)}, R^{(k)}, T^{(k)}\right\rangle \in \{0,1\}^{A\times B} \times \mathbb{R}^{3\times 3} \times SO(3) \times \mathbb{R}^3 : k = \overline{1,K}\right\}$$

Just as with all the surface-based SfS methods, the silhouette is represented only by its contours. It is important for the described algorithm that the contours have an assigned direction. The contours direction inside the silhouette is opposite to that are outside. Therefore, the silhouette is represented by set $S^{(k)} = \left\{O^{(k)}, I^{(k)}\right\}$ where $O^{(k)}$ is the set of outer edges and $I^{(k)}$ is the set of inner edges. The authors assume that a closed contour

is used to describe the silhouette. It can be described as a set of normalized points on the image plane $\left\{\hat{p}_i \in \mathbb{R}^2 : i = \overline{1,N}\right\}$, which generates a set of 2D edges in the form of $\left\{\langle\hat{p}_1,\hat{p}_2\rangle, ..., \langle\hat{p}_{N-1},\hat{p}_N\rangle, \langle\hat{p}_N,\hat{p}_1\rangle\right\}$. Having the knowledge of internal and external camera parameters, we may convert the points' coordinates on the image plane to 3D points' coordinates in the object's coordinates system. Every two points, which form the 2D edge on the image plane, linked with the camera's central point $p_i^{'}$, $p_{(i \bmod N)+1}^{'}$, T defines the plane in 3D space ($n(i)$ means the next after i). For an ideally calibrated camera this plane is tangent in at least one point to the visual hull. Every 2D edge is the result of projecting tangent plane to object on the image's plane. In accordance with the rules of projective geometry, the 2D edges form half-plane limited by rays $\overrightarrow{Tp_i^{'}}$, $\overrightarrow{Tp_{n(i)}^{'}}$, called the strip. The surface of visual hull is created from segment of strips limited by the intersection edges with other strips. The main problem of SfS, based on the VH surface construction, is the effective calculating of planes intersections in the 3D space. The main idea of the described algorithm's is building VH surfaces, basing on the points which determine the beginnings and endings of viewing edges. Viewing edges are half lines $\overrightarrow{Tp_i^{'}}$, $\overrightarrow{Tp_{n(i)}^{'}}$ belonging to the VH surface. After calculating the 3D location and the length of the viewing edges, there is the heuristic procedure of calculating intersection of three strips (the triple points) and then of the edges and the VH surface. The key steps of the algorithm can be seen in Fig. 14.

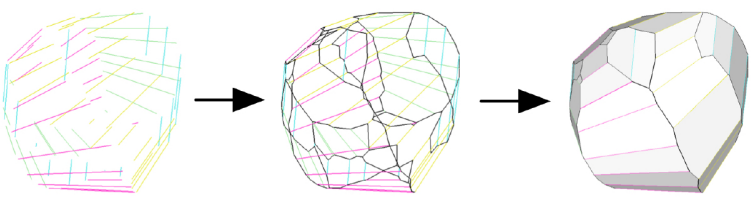


Fig. 14. The key steps of EPVH algorithm: (a) compute the viewing edges, (b) cones intersections and triple points, (c) faces [71].

The first step is to calculate the length and direction of the viewing VH edges. Each normalised point $\hat{p}_i^{(s)}$ which belongs to the closed contour of the silhouette $O^{(s)}$ and the camera's central points $T^{(s)}$ forms a projection line $\ell_i^{(s)}$ in the 3D space. Projection of the lines in the other silhouettes' images are epipolar lines which form the following set: $\left\{\hat{l}_i^{(s,k)} \in \mathbb{R}^3 : \hat{l}_i^{(s,k)} = E^{(s,k)}\hat{p}_i^{(s)}, \hat{p}_i^{(s)} \in O^{(s)}, E^{(s,k)} \in \mathbb{R}^{3\times3}, k \in \{1,..,K\}\backslash s\right\}$, where $E^{(s,k)}$ is the fundamental matrix describing the relation between the silhouettes image s and k. Epipolar line $\hat{l}_i^{(s,k)}$ which goes through the silhouette's interior creates intersection points with contours $O^{(k)}$ and $I^{(k)}$ which determines line segments that do not belong to $\ell_i^{(s)}$. At the end, the 3D points' coordinates of the beginning and end of every line

segment are calculated. The result of the first step of the algorithm is a 3D VH skeleton composed of unlinked viewing edges $E = \left\{\langle v_{2t-1}, v_{2t}\rangle \in \mathbb{R}^3 \times \mathbb{R}^3 : t \in \mathbb{N}^+\right\}$. The next step of the algorithm is to calculate the missing edges which determine the intersection with the visual cones. Each of the initial and final points of the viewing edges forms a polygon's vertex $v_i \in \mathbb{R}^3$, forming the VH surface. The vertices which belong to the viewing edges are not sufficient enough to describe the VH surface. The other vertices (called triple points) are placed where more than two strips intersect. Each vertex v_i generates a left and right edge. The direction of the edges are calculated on the basis of the normal vectors of strips. The sense of vectors is known, because its formed on the basis of contours with specified direction. For an edge which starts in vertex v_i, we calculate its maximal length, limited by the nearest vertex through which it passes. Afterwards, edge $\langle v_i, v_j\rangle$ is projected on all silhouettes $S^{(k)}$ for $k = \overline{1, K}$, calculating the closest point in which the silhouette's edge intersects the contour v_i. If such point exists, then, to the set of triple points vertex v_k is added and $\langle v_i, v_k\rangle$ is added to the set of edges. Otherwise, if such a point does not exist, $\langle v_i, v_j\rangle$ is added to the set of edges. Edges are generated for all of the vertices and triple points, and in result a three-dimensional grid of VH edges is formed. The last step of the algorithm is generating surfaces. For the surfaces limited by the previously calculated edges normal vectors of planes are determined. The planes orientation is calculated on the basis of the direction of the contours edges. The result of the algorithm can be seen in Fig. 15.

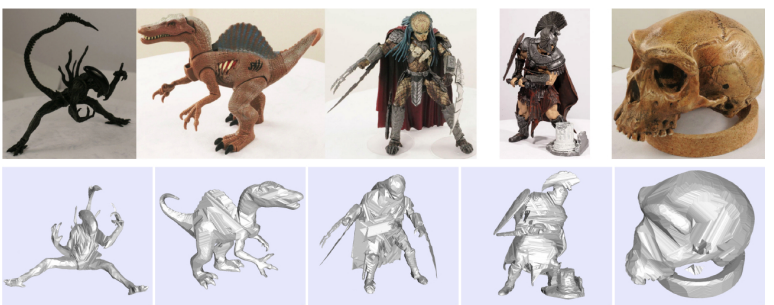


Fig. 15. The results of the EPVH algorithm [71].

The algorithm described above is highly robust to faults in the cameras calibration. The main computational cost of the algorithms is calculating intersections of the visual cones in the 3D space. An important innovation which allows to reduce the computational complexity is calculating the intersections in the 2D space, instead of the 3D space. The description of an effective algorithm which uses the epipolar geometry rules to simplify the problem of calculating the intersection points can be found in [39].

An important hybrid solution, linking the volume-based and surface-based SfS is

presented under the name of marching intersections [40] and marching cubes [51]. Instead of straightforward calculation of points in which the visual cones intersect, we calculate the points in which the cones intersect with the 3D homogeneous grid. The grid lines are parallel to the coordinates axes and equidistant. Just as with previously described SfS algorithms, it is assumed that internal and external cameras parameters are known for every silhouette. The marching intersections algorithm can be split into two stages. The first stage is calculating intersection coordinates of silhouettes visual cones and grid lines, subsequently intersection coordinates are stored in a special data structure – MI. The projection of the grid lines on the silhouette image is done using camera projection matrix, formed on the basis of known internal and external parameters. The rule of determination the points in which the silhouette intersects with the grid lines can be seen in Fig. 16.

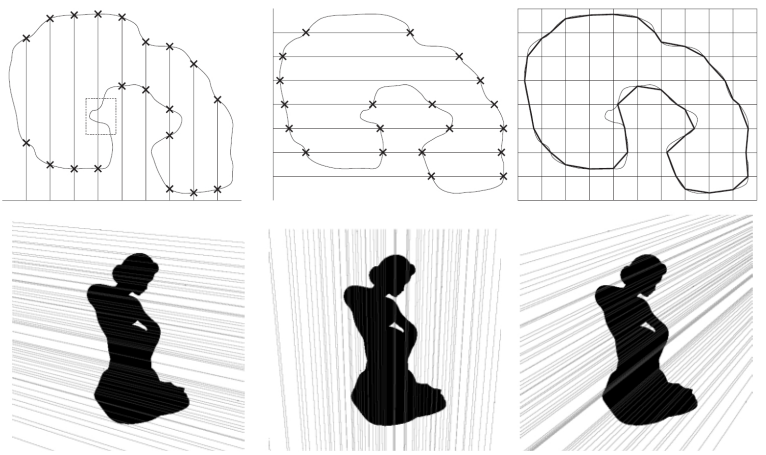


Fig. 16. The marching intersections: upper row – rule of determination the points in which the silhouette intersects with the grid lines (an example of a 2D grid), the bottom row – projection of the 3D grid's line on the silhouettes images[40].

MI structure stores information for every node (a point in which the grid's line is intersected) binary label (inside or outside VH) and the distance of the intersection point for each of coordinate axes. The information is only saved in nodes in which the intersection took place between the directly adjacent nodes. The points in which they intersect are determined, by projecting the grid's line on the silhouette. Subsequently moving along the line, for every intersection the distance between intersection and node is calculated (after perspective projection correction). The second step of the algorithm is the analysis of data stored in MI structures. For every node in which the information

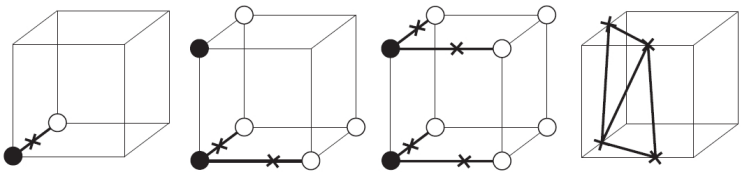


Fig. 17. For black nodes (which are placed inside the VH) with direct neighbour of white nodes (outside the VH) undergo a procedure of surface construction [40].

about the intersection points is saved, on base of heuristic rules surface is constructed from triangles (Fig. 17).

As result of the analysis of whole MI structure we obtain the visual hull's mesh. The algorithm is relatively simple in implementation and may be parallelized. Unfortunately, like in the case of the volume-based SfS, the problem in choosing the right size of the grid is a compromise between the reconstruction's quality and the computational time. The MI algorithm's effect can be seen in Fig. 18.

The "shape from silhouette" methods are a numerous group of reconstruction methods of the 3D scene. Apart from the basis methods there exists a substantial number of hybrid methods with connect SfS with many other methods, in particular with the stereo-vision [42, 46, 69, 70]. The described SfS methods all have a common fault, they

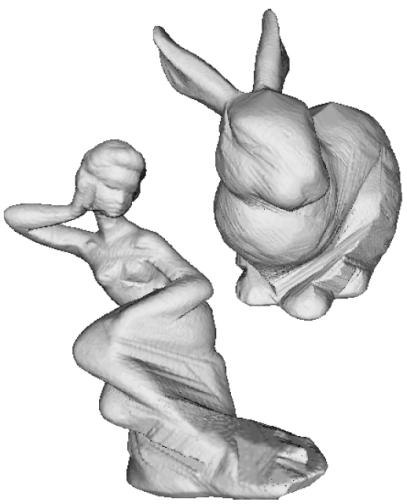


Fig. 18. Example of results obtained from marching intersections algorithm [40].

only allow to obtain the object's model and not the whole scene. Reconstructed object should be placed in center of the scene and additionally it should be a convex object. The "shape from photo-consistency" methods is the SfS generalisation, which allows to reconstruct the scene's image of every shape.

2.4. Shape from photo-consistency

The basis of described SfS methods is the analysis of the binary images coherence (consistent) with the generated VH. If even one projection on the image plane of "potentially full" voxel does not belong to the silhouette what means that is inconsistent with the silhouette's image. An incoherent voxel cannot belong to the VH, therefore it is labeled as "empty". Exactly the same reasoning is basis for the SfPC method (Shape from Photo-Consistency). The color of every images pixels has to be consistent with the shape and color (radiance) of the scene. The key concept that lays the foundations for the method is the notion of photo-consistency. We may define three different types of photo-consistency [30].

Definition 1. (Point Photo-Consistency) Let S be an arbitrary subset of $\mathbb{R}^3$. A point $p \in S$ that is visible from c is photo-consistent with photograph at c if p does not project to a background pixel, and the color at p's projection is equal to $rad_p(\overrightarrow{pc})$. If p is not visible from c, it is trivially photo-consistent with the photography at c.

Definition 2. (Shape-Radiance Photo-Consistency) A shape-radiance scene description is photo-consistent with the photograph at c if all points visible from c are photo-consistent and every non-background pixel is the projection of a point in V.

Definition 3. (Shape Photo-Consistency) A shape V is photo-consistent with a set of photographs if there is an assignment of radiance functions to the visible points of V that makes the resulting shape-radiance description photo-consistent with all photographs.

On the basis of the definitions above, the notion of the test of fulfilling the consistency criterion is introduced. Test is done using the function $\text{consist}_K() : Col^K \times \left\{\mathbb{R}^3\right\}^K \rightarrow \{0,1\}$ which parameters are composed of the pixels' colors $col_1, .., col_K \in Col$ and the vectors pointing the location of the camera $\xi_1, .., \xi_K \in \mathbb{R}^3$. In case of more accurate models of light reflection parameters set contains light's sources' position. The maximum K index is not greater than the number of input images. The function returns value 1 only in case when it is possible for a point on the surface to reflect light of color col_i in the direction ξ_i simultaneously for every $i = \overline{1, K}$. The last element needed to write down the reconstruction algorithm is definition of set of images' for which a point is visible. If p is point on shape surface V, then $p \in \text{Surf}(V)$. Set $\text{Vis}_V(p)$ is the subset of input images in which V does not occlude p. On the basis of the definitions above, we may write down the general algorithm of 3D scene's shape reconstruction, called Space Carving.

The practical implementation of the Space Carving algorithm requires solving the

Algorithm 2.2 Space Carving

Initialize space $V = \{0,..,X\} \times \{0,..,Y\} \times \{0,..,Z\}$ containing the whole reconstructed scene
Initialize $found \leftarrow 1$
while $found = 1$ `/* Found voxel` $v \in V$ `non shape photo-consistent with projections of` Surf(V) `*/` **do**
    $found \leftarrow 0$
    for all $v \in V$ **do**
        $p \leftarrow$ coordinates of center of v
        for $j \leftarrow 1, K$ **do**
            $\xi_j \leftarrow p\vec{C}_j$, where C_j is center of camera j
            if $j \in \mathrm{Vis}_V(p)$ **then**
                $col_j \leftarrow \mathrm{Col}(p, j)$, where $\mathrm{Col}(p, j)$ is color of the projection of point p on the image plane of camera j
            else
                $col_j \leftarrow none$
            end if
        end for
        if $\mathrm{consist}_K(col_1, ..., col_K, \xi_1, ..., \xi_K) \neq 1$ **then**
            $found \leftarrow 1$
            $V \leftarrow V \backslash \{v\}$
        end if
    end for
end while
Result: Set of photo-consistent voxels V

problem of effective calculating set $\mathrm{Vis}_V(p)$ for every voxel. The solution is to force the sequence of processing voxels in accordance with the "plane sweep" algorithm [15, 19] or its extension "multi plane sweep" [30]. They key idea of the plane sweep algorithm (see alg. 2.3) is to divide the analysed space into parallel slices, which are intersections of plane and the voxels grid. The plane is perpendicular to one of the axes of coordinate system, for example, to X. The analysis starts from the marginal slice and by increasing value of one coordinate (such as X) it moves inside the scene. When we know the video camera's location and their orientation we may calculate which video camera can see a specific voxel (Fig. 19).

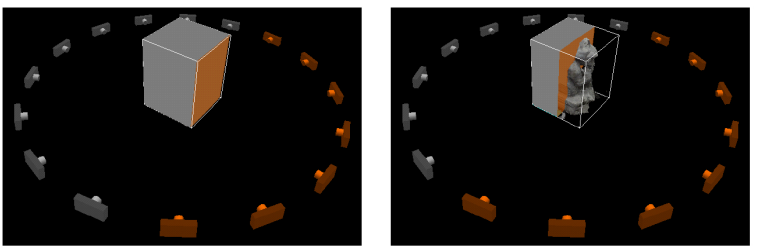


Fig. 19. Plane sweep algorithm. When voxels slice moves from the initial positon (the image on the left) inside the scene (the image on the right) the number of cameras (which take part in determining photo-consistency with scenes shape) increases [40].

The multi-sweep space carving algorithm is different from the plane sweep one, because instead of moving in one direction, the surface moves in six different directions (increasing and decreasing the values in X, Y, Z). The multi plane sweep algorithm can be written as alg. 2.4 [30].

Algorithm 2.3 Plane Sweep

Step 1: Get an initial volume V, initialize the sweep plane Π such that V lies below Π (ie. swept towards V)
Step 2: Interesect Π with the current shape V
Step 3: For each surface voxel v on Π :
a. let $C_1, .., C_j$ be the cameras above Π for which v projects to an *unmarked* pixel;
b. determine the photo-consistency of v using $\mathrm{consist}_K(col_1, ..., col_j, \xi_1, ..., \xi_j)$;
c. if v is inconsistent then set $V \leftarrow V \backslash \{v\}$, otherwise mark the pixels to with v projects.
Step 4: Move Π downward one voxel width and repeat **Step 2** until V lies above Π.
Result: Set of photo-consistent voxels V

Algorithm 2.4 Multi-Sweep Space Carving

Step 1: Initialize V to be a superset of the true scene
Step 2: Apply the Plane Sweep Algorithm in each of the six principle directions and update V accordingly.
Step 3: For every voxel in V whose consistency was evaluated in more than one plane sweep:
a. let $C_1, .., C_j$ be the cameras that participated in the consistency check of v in *some* plane sweep during **Step 2**;
b. determine the photo-consistency of v using $\text{consist}_K(col_1, ..., col_j, \xi_1, ..., \xi_j)$;
Step 4: If no voxels were removed from V in **Step 2** and **Step 3**, set $V* \leftarrow V$ and terminate; otherwise, repeat **Step 2**.
Result: Set of photo-consistent voxels $V*$

The reconstruction time of the scene's 3D shape, on the basis of the algorithm described above depends mainly on the voxels' grid size. Similarily to the volume-based SfS and Space Carving methods described earlier it is based on a full review of voxels. Their size and quantity are a compromise between reconstruction time and reconstruction quality. Run time of the multi-sweep algorithm may be considerably reduced by using parallel processing and hardware acceleration of mapping textures on graphic cards. Sample results of the multi-sweep space carving algorithm's can be seen in Fig. 20

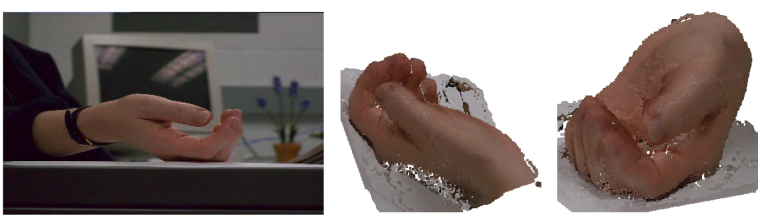


Fig. 20. The results of SfPC multi-sweep algorithm (middle, right) for a sequence of a hundred input images (left) [30].

The quality of the reconstruction depends significantly on the algorithm of determining value of the function consist_K. Most frequently the algorithm uses simple model of the image formation. Simple models do not takes into account phenomena such as specular highlight, mirror reflections or glass transparency. As in the case of SfS, lack of precision in calibration of even one camera has a significant impact on the reconstruction quality. Every pixel which was wrongly verified as inconsistent removes the voxel. It is the main disadvantage of this method. Because of noise in brightness function and

calibration errors, with the increase of the number of images increases risk of erroneous voxel removal.

2.4.1. Shape from photo-consistency and stereovision

There is a very interesting solution to the problem of finding the photo-consistent 3D shape and its images proposed in the hybrid solution linking SfPC with stereovision [65]. The key idea of the described algorithm is to determine photo-consistent voxels by voting of cameras. The proposed measure of the points photo-consistency is defined in the following way:

$$\rho(x) = \exp\left(-\mu \sum_{i=1}^{K} vote_i(x)\right) \tag{62}$$

where:
$x \in \mathbb{R}^3$ -point in 3D space
$\mu \in \mathbb{R}^+$ -constant
$K \in \mathbb{N}$ -number of input images
$vote_i : \mathbb{R}^3 \to \mathbb{R}$ -voting function of i-th camera

It is assumed that for every input image the internal and external camera's parameters are known (projection function $\Pi()$ for each camera is defined). The algorithm of calculating the function's $vote_i(x)$ value was described as alg. 2.6 and subalgoritm alg. 2.5.

Algorithm 2.5 Function $score_i$

Input: x – point position in 3D space
Input: d – distance from point x
1: $score_i \leftarrow 0$
2: Calculate the function $o_i : \mathbb{R} \to \mathbb{R}^3$ which determines point lying on straight line passing through the point x and the i-th camera central point c_i: $o_i(d) = x + (c_i - x)d$
3: $p_i \leftarrow \Pi_i(x)$ `/* where` Π_i `- projection function of i-th camera*/`
4: **for all** $j \in N(i)$ `/* where` $N(i)$ `- set of nearest cameras*/` **do**
5: $p_j \leftarrow \Pi_j(o_i(d))$
6: $score_i \leftarrow score_i + \frac{1}{|N(i)|} NCC_{ij}(p_i, p_j)$ `/* where` $NCC_{ij}(p_i, p_j)$ `- Normalized Cross Correlation of i-th images brightness function` $I_i()$ `in window surrounding pixel` p_i `and j-th images brightness function` $I_j()$ `in window surrounding pixel` p_j `*/`
7: **end for**
Result: $score_i$

Algorithm 2.6 Function $vote_i$

Input: x – point position in 3D space
1: Initialize V to be a superset of the true scene
2: $vote_i \leftarrow score_i(x, 0)$
3: Calculate the parameters of the function $o_i : \mathbb{R} \rightarrow \mathbb{R}^3$ which determines point lying on straight line passing through the point x and the i-th camera central point c_i: $o_i(d) = x + (c_i - x)d$
4: **for all** $d \in \{d : o_i(d) \in V\}$ **do**
5: **if** $score_i(x, 0) < score_i(x, d)$ **then**
6: $vote_i \leftarrow 0$ `/*only if` $score_i$ `has global maximum in d=0 function returns` $score_i(x, 0)$ `otherwise 0*/`
7: **end if**
8: **end for**
Result: $vote_i$

Key equation of alg. 2.5 can be written in another notation as:

$$score_i(d) = \frac{1}{|N(i)|} \sum_{j \in N(i)} NCC_{ij}\,(d) \tag{63}$$

This simple solution is not much immune to the errors in determining the correlation, resulting from occlusion and noises. A much better solution is to calculate the local maxima of the Normalized Cross Correlation function, according to universal rule $\frac{\partial NCC_{ij}}{\partial d}\,(d_k) = 0$ and $\frac{\partial^2 NCC_{ij}}{\partial d^2}\,(d_k) > 0$. Afterwards, using Parzen window method with the kernel W calculate correlation score function as:

$$score_i(d) = \sum_{j \in N(i)} \sum_{k} NCC_{ij}\,(d_k) W(d - d_k) \tag{64}$$

The practical implementation of the alg. 2.6 requires discretization of the 3D space V, using a homogeneous grid of voxels. The result of undergoing the voting procedure is assigning every voxel a measure of photo-consistency (exactly photo-inconsistency). The obtained grid is the starting point of the shape's approximation of the scene's surface. The shape of the scene S should minimise the functional:

$$E\,(S) = \iint_S \rho\,(x)\,dA - \lambda \iiint_{V(S)} dV \tag{65}$$

where:
$V(S)$ – volume enclosed by the surface S

The negative coefficient $-\lambda$ causes with an unchanged consistency cost, such a scene's shape is favored which encloses largest possible volume (so called, ballooning effect).The authors proposed a method of optimization the energy $E(S)$, based on the minimal graphs cut. The graphs construction consists in forming an vertex for every voxel and linking the vertex with edges in such a way so that the vertex (voxel) is connected symmetrically with six surrounding nearest neighbours. The weight function of the edges between the vertex x_i, x_j for voxel size h can be calculated as:

$$w_{ij} = \frac{4\pi h^2}{3} \rho \left(\frac{x_i - x_j}{2} \right) \tag{66}$$

The results of the described algorithm are shown in Fig. 21

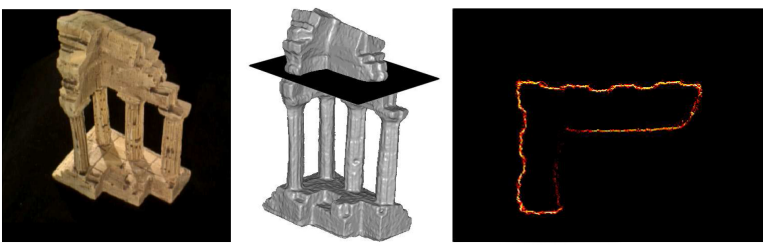


Fig. 21. The results of the algorithm [65], (left) -– one of sequence of 312 input images, (middle) – a 3D model generated on the basis of the input images, (right) — the accumulated value $vote_i$ for voxels in the slice denotated as plane in middle image.

A important drawback of the SfPC algorithms described so far is the long operation time. It is due to the fact that these methods are based on scenes voxel full review. A considerable execution time reduction of SfPC based method can be obtained by using calculus of variations.

2.4.2. Shape from photo-consistency – calculus of variations

The shape from photo-consistency theory assumes that there is a function consist_K which allows to check photo-consistency of the scene's shape with the input images. The problem in using this method is a relatively long time needed for checking if all regions of the observed surface are consistent with the images. Limited reconstruction resolution determined by the minimal size of a voxel is also problematic. A very promising direction in this research is using variational methods based on the photo-consistency. Instead of a function consist_K, there is an functional that defines energy of adjustment the shape of the scene to its images. A discrete voxel representation of the scene's shape is replaced by a continuous function of scenes surface. Variational methods allows to find the scenes shape which adjustment energy is the smallest [64, 77].

Stuhmer et al. in work [64] describe the methods which makes it possible to obtain a dense scene's reconstruction in real time. An algorithm's input is a set of images obtained from a moving video camera $\{I_i : \Omega_i \to \mathbb{R}\}$, $i = \overline{1,N}$.The result of the algorithm's operation is assigning every pixel of I_0 a distance value, that is a depth function $h : \Omega_0 \to \mathbb{R}$. For presenting the method's details it is necessary to enter a few additional denotations. Mapping between points coordinates in external three-dimensional coordinates system into coordinates in camera's pixel coordinates system is called a projection function $\pi : \mathbb{R}^3 \to \mathbb{R}^2$. The image points coordinates are denoted as homogeneous coordinates $x = [x_1, x_2, 1]^T \in \Omega_0$. The coordinates in 3D space are calculated from depth function as $X(x,h) = h(x1, x2) \cdot x$. Projecting the 3D points that has coordinates X on Ω_i a different than Ω_0 image plane may be calculated by using formula $\pi(\exp(\hat{\xi}_i) \cdot X)$, where $\xi \in \mathbb{R}^6$ are camera's "twist" coordinates. Operator $\wedge : \mathbb{R}^6 \to se(3)$ allows to present the "twist" coordinates as linear (matrix) Lie's representation. Matrix exponential of $\hat{\xi}_i$ it's transformation matrix between coordinates system of I_0 and I_i with the assumption of rigid body motion. Best way to show principle of algorithm operation is to start from case of only two camera's locations. The algorithm's aim is to find h function, for which the value of the energy functional is minimal:

$$E(h) = \lambda \int_{\Omega_0} \left| I_1\left(\pi\left(\exp\left(\hat{\xi}_i\right) \cdot X(x,h)\right)\right) - I_0(x)\right| d^2x + \int_{\Omega_0} |\nabla h| \, d^2x \tag{67}$$

The first integral is a data term, the second is a regularization term. The data term in this case is the difference between the second image's points' brightness and the brightness of first image points projection on second image for assumed shape of the distance function. Energy minimization of only data term could lead to unrealistic results as form of very irregular surface, particularly in the presence of noise. Regularization is implemented to enforce smoothes of surface. The regularization term is total variation of h function. In order to simplify the notation, a denotation $I_1(x,h)$ is introduced for $I_1\left(\pi\left(\exp\left(\hat{\xi}_1\right) \cdot X(x,h)\right)\right)$.

Linear approximation of $I_1(x,h)$ by using Taylor's formula leads to a new shape of the energy functional:

$$E(h) = \int_{\Omega_0} \lambda \underbrace{\left| I_1(x,h_0) + (h-h_0)\frac{d}{dh} I_1(x,h)|_{h_0} - I_0(x)\right|}_{\rho_1(x,h)} d^2x + \int_{\Omega_0} |\nabla h| \, d^2x \tag{68}$$

Generalizing for many images we obtain:

$$E(h) = \lambda \int_{\Omega} \sum_{i \in \Im(x)} \rho_i(x,h) d^2x + \int_{\Omega} |\nabla h| \, d^2x \tag{69}$$

where:

$\Im(x)$ – the set of images indexes for which projection $\pi\left(\exp\left(\hat{\xi}_i\right)\cdot X(x,h)\right)$ is in boundaries of the image

$\rho_i(x,h) = I_i(x,h_0) + (h-h_0)\frac{d}{dh}I_i(x,h)|_{h_0} - I_0(x)$

Finding the minimal value of the functional (69) is difficult because of the discontinuous differentiability. The solution proposed by the authors is to enter supplementary function u, which separates the data term from the regularization term and allows to build a functional which is a convex approximation 69.

$$E_\theta = \int_\Omega \left\{ |\nabla u| + \frac{1}{2\theta}(u-h)^2 + \lambda \sum_{i\in\Im(x)} |\rho_i(x,h)| \right\} d^2x \tag{70}$$

where: θ – small constant

The functional minimum can be determined by using minimization scheme of functionals based on more than one functionals. The problem is decomposes into single functional minimization and solved in iterative way (thresholding scheme). Proposed by authors method is to use primal-dual algorithm for minimization of ROF energy. For defined h, we calculate the energy's minimum which has the following form:

$$\min_u \int_\Omega \left\{ |\nabla u| + \frac{1}{2\theta}(u-h)^2 \right\} d^2x \tag{71}$$

The solution is $u = h - \theta p$ function, where $p = (p_1, p_2)$ is the vector field which fulfils the differential equation $\nabla(\theta \mathrm{div} p - h) = |\nabla\theta \mathrm{div} p - h|\, p$. The vector field above can be calculated using the iterative method, using the formula:

$$[p^{k+1} = \frac{p^k + \tau\nabla\left(\mathrm{div}\ p^k - h/\theta\right)}{1 + \tau\nabla\left|\mathrm{div}\ p^k - h/\theta\right|} \tag{72}$$

where: $p^0 = 0$, $\tau < \frac{1}{8}$

For defined u, there is a functional's minimum:

$$\min_h \int_\Omega \left\{ \frac{1}{2\theta}(u-h)^2 + \lambda \sum_{i\in\Im(x)} |\rho_i(x,h)| \right\} d^2x \tag{73}$$

The functional (73) cannot be directly minimized by using the gradient methods, because it is not continuously differentiable in the whole interval. The fact that discontinuous places exist results from the used norm L_1. The critical points in which the

functional is indifferentiable are places in which the function $\rho_i(x,h)$ goes through zero and change its sign. The critical points can be calculated using formula:

$$t_i = -\frac{I_i(x,h_0) - h_0 I_i^h(x) - I_0}{I_i^h(x)}, i \in \Im(x) \tag{74}$$

where:
$I_i^h(x) \triangleq \frac{d}{dh} I_i(x,h)\Big|_{h_0}$

The image's indexes are chosen in such a way so that the critical points are sorted in a non-decreasing order $t_i \leqslant t_{i+1}$. Moreover, there are two points added: $t_0 = -\infty$ and $t_{|\Im(x)|+1} = +\infty$. The shape of h function which guarantees the functional's minimum, can be found using the following strategy:

if the stationary point

$$h_1 = u - \lambda\theta \left(\sum_{i \in \Im(x): i \leqslant k} I_i^h(x) - \sum_{j \in \Im(x): i > k} I_j^h(x) \right) \tag{75}$$

belongs to interval (t_k, t_{k+1}) for same $k \in \Im(x)$, then the solution is $h = h_1$, otherwise:

$$h = \arg \min_{h_2 \in \{t_i\}} \left(\frac{1}{2\theta}(u - h_2)^2 - \lambda \sum_{i \in \Im(x)} |\rho_i(x, h_2)| \right) \tag{76}$$

The implementation of the algorithm on GPU makes it possible to make a dense scene's shape reconstruction in real time. The image sequence of resolution 640×480 pixels, using GPU Nvidia GTX 480, was processed at speed of 11 frames per second. The quality of the obtained reconstruction depends on the size and number of input images. Moreover, the level of texturing has also got a considerable impact. An example of such reconstruction obtained using the method described above can be seen in Fig. 22.

The methods described so far were based on a relation analysis between the 3D point's location and the position of their projections on the image plane, with defined location of the cameras. A completely different carrier of information about the distance is used by shape from defocus methods.

2.5. Shape from defocus

The last of the presented grup of algorithms of passive 3D scene's reconstruction is the shape from defocus (SfD). It is characteristic for this method to use the imperfection's model of the camera's optical system to calculate the distance. The image's blurring is

Fig. 22. The results of SfPC algorithm [77]. The top line shows the obtained 3D model and the impact of an increasing number of images on the quality of mapping the real scene. The bottom line shows selected frames from a sequence of input images.

function of distance from focal point of the camera's optical system and image plane. The blurring is measured by the radius of the circle of confusion, which can be calculated using formula:

$$b = \frac{D\nu}{2}\left|\frac{1}{F} - \frac{1}{\nu} - \frac{1}{s}\right| \tag{77}$$

where:

b – radius of circle of confusion
D – radius of lens
F – focal distance
ν – distance between image plane and principal point of lens
s – distance between scene's point and principal point of lens

The image formed on the image plane is ideally sharp only for 3D points which lies on plane to the image parallel for which the circle of confusion radius equals zero. The image is out of focus for all the points which do not lie on this plane. Acquisition of a single image happens with constant parameters F,ν, D. However s may be different for every point of image plane. Therefore, the image's model forming, which includes the blurring phenomenon can be written in the following form:

$$I(y) = \int_{\Omega} h(y, x, s(x); F, D, \nu)\, r(x)\, dx \tag{78}$$

where:

$\Omega \subseteq \mathbb{R}^2$	–	image space
$x, y \in \Omega$	–	points of image
$s(x)$	–	distance from image plane
$r(x)$	–	radiance
$h(\cdot)$	–	kernel

The SfD is an inverse problem, its calculating $s(x)$ value for every $y \in \Omega$ on the basis of image I. Method of directly calculating the scene's shape on the basis of the general model (78) is unknown. The SfD algorithms described in the literature are based on simplified models. The most common simplification used is defining the point's local surroundings as a surface parallel to the image plane (equifocal assumption) [6, 12, 14, 18, 23, 27, 37]. It allows to write down the image point's brightness equations as a convolution of kernel and the surface radiation function (for notation simplification the constants F, D, ν were omitted):

$$I(y) = \int_{\Omega} h(y, x) * r(x)\, dx \tag{79}$$

The kernel is called the Point Spread Function. The shape of PSF changes depending on the surface's point distance from the image surface. Most commonly in the literature there are two PSF forms – the Pillbox function and Gaussian function. The measurement of defocus is adjustment of PSF's model parameters (for example, the standard deviation of Gaussian function). On the basis of PSF model's parameters we may calculate the distance from the image surface. The main problem is that the Point Spread Function has to be isolated from the image brightness function by using deconvolution. Unfortunately, apart from PSF the radiance of scene is also unknown. Therefore, the task is the blind deconvolution of two integrated signals, which is ill-posted inverse problem. Because the general method of solving this class of problems is unknown, many algorithms have been developed that allows to calculate the approximated PSF parameters. In the context of this article, we are going to describe only the methods which are based on more than one image. At this point we must differentiate the shape from defocus (SfD) methods from the shape from focus (SfF) methods. The SfF are methods of calculating the scene's shape by obtaining a series of images, which differ in the distance of plane for which all of the points are ideally focused (focal plane). Subsequently by measuring the defocus and selecting points with minimal defocus we determine the intersection points of scenes surfaces and focal plane. The isolines obtained as the result determine the approximated shape of the scene. The reconstruction's quality strongly depends on the number of images and the differences in the camera focal settings. The necessity of obtaining a large number of images from static camera, causes the SfF methods to have limited practical use. The SfD methods are based on the analysis of two

images. Most frequently the basis of determining the scene's shape is the ratio of the defocusing measure (relative blur). The differentiator of the SfD methods is the way of measuring defocus. Considering the domain in which the analysis is carried out, the methods can be divided into frequency methods and spatial methods.The foundation of the methods which are based on Fourier transform is the fact that the convolution with PSF works as low-pass filter of characteristics dependent on defocus. Apart from Fourier transform, authors of algorithms use a wide range of different mathematical tools, such as convolutions, orthogonal functions in Hilbert space [37], S-transform of the images approximated by a polynomial [12], Markov Random Fields [21], the global and local optimisation methods [60]. At the end, the result of every algorithm is a distance map, which is obtained by converting the defocus measure into distance. Often, an additional step of the algorithm is to smooth the distance map, using filters (such as a bilateral filter). There is one main problem of methods which are based on equifocal assumption. Error of defocus measure depends on window size of local planar surface approximation. Enlarging the window's size enables to calculate estimators of the PSF parameters with less variation, but causes also an increase in the error of mapping the scenes shape (in reality the scene is not a parallel to the image plane). Therefore there is a forced compromise between the precision and robustness (independence from the type of the scenes texturing).

The new approach allows modeling the process of forming the image without the equifocal assumption (non equifocal model). To present the issues and concepts connected with SfD an algorithm which belongs to the group of variation methods will be accurately described. The publication [67] contains descriptions of SfD methods which are based on analogy between image blurring and process of heat diffusion. The algorithm's input data are two images of the same scene , obtained using the same camera at static position and orientation but with different parameters of the optical system. It is assumed that the radius of CoC (blur) is regulated by changing the distance of the image plane from the lens principal point (ν_1, ν_2) with the unchanged focal length (F) and the lens sizes (D). Understanding the algorithm requires the introduction of few concepts. The first is relative blur. In the simplest case in which the PSF is Gaussian function, which shape depends only on the point's distance and does not depend on their location (shift invariant) and the assumption that the image I_2 is more out of focus than the image I_1, we may write down the equation:

$$I_2(y) = \int \frac{1}{2\pi\sigma_2^2} e^{-\frac{\|x-y\|^2}{2\sigma_2^2}} r(x)\,dx = \int \frac{1}{2\pi\Delta\sigma^2} e^{-\frac{\|x-y\|^2}{2\Delta\sigma^2}} I_1(x)\,dx \tag{80}$$

where:

$\Delta\sigma^2 \triangleq \sigma_2^2 - \sigma_1^2$	–	relative blur
σ_1^2, σ_2^2	–	variations accordingly of the first and second image

The introduction of the relative blur eliminates the scene's radiance r from the equation which is unknown. The equation (80) can also be interpreted as the heat diffusion equation. Introducing a time variable we may write down the equation (80) in the form of $u(y,t_1) = I_1(y)$ and $u(y,t_2) = I_2(y)$, $\forall y \in \Omega$. For $\sigma_2 > \sigma_1$ we can form the following system of equations:

$$\begin{cases} \dot{u}\,(y,t) = c\boldsymbol{\Delta} u\,(y,t) & c \in [0,\infty) \\ u\,(y,t_1) = I_1(y) & \forall y \in \Omega \end{cases} \tag{81}$$

where:
$\dot{u} \triangleq \frac{\delta u}{\delta t}$ – derivative of function u with respect to time t
$\boldsymbol{\Delta}$ – laplacian

A case when $\sigma_2 < \sigma_1$ can be solved by interchanging $I_1(y)$ and $I_2(y)$. To simplify the analysis, variable t is introduced as the increase of time in relation to $t_1 = 0$. Therefore $u(y,0) = I_1(y)$, $u(y,\Delta t) = I_2(y)$, $\forall y \in \Omega$. We may write down the relation between the relative blur and time increment as:

$$\Delta\sigma^2 = 2\Delta t c \tag{82}$$

for $c = \frac{\gamma^2(b_2^2 - b_1^2)}{2\Delta t}$ where $b_i = \frac{D\nu_i}{2}\left|\frac{1}{F} - \frac{1}{\nu_i} - \frac{1}{s}\right|$, $i = 1,2$, γ is constant greater than zero.

According to the assumptions above, we may write down the following system of equations:

$$\begin{cases} \dot{u}\,(y,t) = \nabla \cdot (c\,(y)\,\nabla u\,(y,t)) & t \in (0,\infty) \\ u\,(y,0) = I_1(y) & \forall y \in \Omega \\ c\,(y)\,\nabla u\,(y,t) \cdot n\,(y) = 0 & \forall y \in \delta\Omega \\ u\,(y,\Delta t) = I_2(y) & \forall y \in \Omega \end{cases} \tag{83}$$

where:
$\nabla\cdot$ – divergence
∇ – gradient
Ω – space of image
$\delta\Omega$ – boundaries of image
n – unit normal vector at boundaries of image

Solution to the system of equations for a real pair of images does not guarantee obtaining correct results, because the earlier assumption for the whole image $\sigma_2^2 > \sigma_1^2$ (or $\sigma_2^2 < \sigma_1^2$) does not have to be fulfilled. Practically, every pair of images contains areas in which $\sigma_2^2 > \sigma_1^2$ and areas in which $\sigma_2^2 < \sigma_1^2$. Additionally, in case the assumption is not fulfilled, the algorithm is no longer numerically stable. It is crucial to develop the algorithm by forcing the one-way diffusion in which $c\,(y) > 0$ is in the whole domain of analysis (forward diffusion). The proposed solution is to split the images on two separate areas:

$$\Omega_+ \triangleq \{y \in \Omega : c\,(y) > 0\} \tag{84}$$

and

$$\Omega_- \triangleq \{y \in \Omega : c(y) \leqslant 0\} \tag{85}$$

Therefore, the system of equations obtains a new form:

$$\begin{aligned}
\dot{u}(y,t) &= \begin{cases} \nabla \cdot (c(y) \nabla u(y,t)) & \forall y \in \Omega_+, t \in (0,\infty) \\ \nabla \cdot (-c(y) \nabla u(y,t)) & \forall y \in \Omega_-, t \in (0,\infty) \end{cases} \\
u(y,0) &= \begin{cases} I_1(y) & \forall y \in \Omega_+ \\ I_2(y) & \forall y \in \Omega_- \end{cases} \\
c(y) \nabla u(y,t) \cdot n(y) &= 0 \qquad \forall y \in \delta\Omega_+ = \delta\Omega_- \\
u(y,\Delta t) &= \begin{cases} I_2(y) & \forall y \in \Omega_+ \\ I_1(y) & \forall y \in \Omega_- \end{cases}
\end{aligned} \tag{86}$$

Diffusion coefficient $c(y)$ is a function dependent on distance map $s(y)$ by the relation:

$$c(y) = \frac{\gamma^2 D^2}{8\Delta t} \left(\nu_2^2 \left(\frac{1}{F} - \frac{1}{\nu_2} - \frac{1}{s(y)} \right)^2 - \nu_1^2 \left(\frac{1}{F} - \frac{1}{\nu_1} - \frac{1}{s(y)} \right)^2 \right) \tag{87}$$

Therefore, diffusion coefficient $c(y) = 0$ when $s(y) = \frac{(\nu_1+\nu_2)F}{\nu_1+\nu_2-2F}$ or $s(y) = F$. It makes it possible to precisely define the boundaries:

$$\delta\Omega_+ = \delta\Omega_- = \left\{ y : s(y) = \frac{(\nu_1+\nu_2)F}{\nu_1+\nu_2-2F} \vee s(y) = F \right\} \tag{88}$$

and the areas:

$$\Omega_+ = \left\{ y : 0 < s(y) < F \vee s(y) > \frac{(\nu_1+\nu_2)F}{\nu_1+\nu_2-2F} \right\} \tag{89}$$

$$\Omega_- = \left\{ y : F < s(y) < \frac{(\nu_1+\nu_2)F}{\nu_1+\nu_2-2F} \right\} \tag{90}$$

The boundaries $\delta\Omega_+$, $\delta\Omega_-$ are areas which on both images have the same level of blurring. Their shape depends directly on the scene's shape. The scene's shape has an impact on function u by coefficient $c(y)$ and can be calculated by minimizing the function:

$$\hat{s} = \arg\min_s \left\{ \begin{array}{l} \int H(c(y)) |u(y,\Delta t) - I_2(y)|^2 dy + \\ + \int H(-c(y)) |u(y,\Delta t) - I_1(y)|^2 dy + \\ + \alpha \|\nabla s\|^2 + \kappa \|s\|^2 \end{array} \right\} \tag{91}$$

where:

$H(\cdot)$	–	Heaviside step function
$\alpha > 0$	–	constant
$\kappa > 0$	–	constant

To simplify the notation, the following is introduced:

$$E(s) \triangleq E_1(s) + E_2(s) + E_3(s) \tag{92}$$

where:
$E_1(s) \triangleq \int H(c(y)) |u(y, \Delta t) - I_2(y)|^2 dy$
$E_2(s) \triangleq \int H(-c(y)) |u(y, \Delta t) - I_1(y)|^2 dy$
$E_3(s) \triangleq \alpha \|\nabla s\|^2 + \alpha\kappa \|s\|^2$

Sum $E_1(s) + E_2(s)$ defines data term and $E_3(s)$ defines regularization term of the functional $E(s)$. The structure of regularization term $\alpha\|\nabla s\|^2 + \alpha\kappa\|s\|^2$ for a very small κ causes that energy $E(s)$ is smaller for small, smooth surfaces. Finding $\hat{s} = \arg\min_s E(s)$ is possible by using the standard method of steepest descent. The changes in the surface shape are indexed by pseudo-time variable, hence:

$$\frac{\partial s}{\partial \tau} \triangleq -E'(s) \tag{93}$$

According to previously established denotations $E'(s) = E_1'(s) + E_2'(s) + E_3'(s)$. Using the chain rule for a data term we obtain ($\tilde{E}_i$ denotes E_i functional to avoid ambiguity):

$$E_i'(s) = \tilde{E}_i'(c(s)) c'(s), \ i = 1, 2 \tag{94}$$

where

$$c'(s) = \frac{\gamma^2 D^2 (\nu_2 - \nu_1)}{4 s^2 \Delta t} \left[(\nu_2 + \nu_1) \left(\frac{1}{F} - \frac{1}{s} \right) - 1 \right] \tag{95}$$

At last, the gradients have the following form:

$$\begin{aligned} \tilde{E}_1'(c(s))(y) = -2H(c(y)) \int_0^{\Delta t} \nabla u(y, t) \cdot \nabla w_1(y, \Delta t - t)\, dt \\ + \delta(c(y)) (u(y, \Delta t) - I_2(y))^2 \end{aligned} \tag{96}$$

$$\begin{aligned} \tilde{E}_2'(c(s))(y) = -2H(-c(y)) \int_0^{\Delta t} \nabla u(y, t) \cdot \nabla w_2(y, \Delta t - t)\, dt \\ + \delta(-c(y)) (u(y, \Delta t) - I_1(y))^2 \end{aligned} \tag{97}$$

where:
$\delta(\cdot)$ – Dirac delta function
$w_1 : \Omega_+ \times [0, \infty) \to \mathbb{R}$
$w_2 : \Omega_+ \times [0, \infty) \to \mathbb{R}$

Funtions w_1,w_2 fulfill the following systems of equations:

$$\begin{cases} \dot{w}_1(y,t) = \nabla \cdot (c(y) \nabla w_1(y,t)) & t \in (0,\infty) \\ w_1(y,0) = u(y,\Delta t) - I_2(y) & \\ c(y) \nabla w_1(y,t) \cdot n(y) = 0 & \forall y \in \delta\Omega_+ \end{cases} \tag{98}$$

and

$$\begin{cases} \dot{w}_2(y,t) = \nabla \cdot (-c(y) \nabla w_2(y,t)) & t \in (0,\infty) \\ w_2(y,0) = u(y,\Delta t) - I_1(y) & \\ c(y) \nabla w_2(y,t) \cdot n(y) = 0 & \forall y \in \delta\Omega_- \end{cases} \tag{99}$$

The gradient of the regularization term has the following form:

$$E_3'(s)(y) = -2\alpha\Delta s(y) + 2\alpha\kappa s(y) \tag{100}$$

Theoretically, calculating the direction of the gradient's functional could be the last element needed to implement an iterative method of calculating the minimum of $E(s)$. The authors, however, propose additional step of preconditioning, thanks to which algorithm's numerical stability will improve and in consequence the robustness of method. Pre-conditioning aims to reduce the impact of values of the functions $u(y,\Delta t)$,$u'(y,\Delta t)$ on the result of the algorithm. Pre-conditioning involves replacing the equation (93) by

$$\frac{\partial s}{\partial \tau} \triangleq -M(s) E'(s) \tag{101}$$

where:
$M(s)$ – positive definite operator
The proposed pre-conditioning operator is:

$$(M(s)\phi)(y) = \frac{\phi(y)}{2\left[H(c(y)) I_2(y) + H(-c(y)) I_1(y)\right] |u'(y,\Delta t)|} \tag{102}$$

where:
$u'(y,\Delta t)$ – derivative of functional respect to s
Pre-conditioning of the gradient of the functional's $E(s)$ data term has the form:

$$\begin{aligned} (M(s) E'_1(c(s)))(y) = \; & H(c(y)) \left(\tfrac{u(y,\Delta t)}{I_2(y)} - 1\right) \tfrac{u'(y,\Delta t)}{|u'(y,\Delta t)|} \\ & + \tfrac{1}{2}\delta(c(y)) \tfrac{(u(y,\Delta t) - I_2(y))^2}{I_2(y)|u'(y,\Delta t)|} \end{aligned} \tag{103}$$

and

$$\begin{aligned} (M(s) E'_2(c(s)))(y) = \; & H(-c(y)) \left(\tfrac{u(y,\Delta t)}{I_1(y)} - 1\right) \tfrac{u'(y,\Delta t)}{|u'(y,\Delta t)|} \\ & - \tfrac{1}{2}\delta(c(y)) \tfrac{(u(y,\Delta t) - I_1(y))^2}{I_1(y)|u'(y,\Delta t)|} \end{aligned} \tag{104}$$

Preconditioning of regularization term is unnecessary, hence finally:

$$\begin{aligned}\frac{\partial s}{\partial \tau} &\triangleq -M(s)\left(E'_1(s)+E'_2(s)+M(s)^{-1}E'_3(s)\right)= \\ &= -M(s)\left(E'_1(s)+E'_2(s)\right)-E'_3(s)\end{aligned} \tag{105}$$

All of the notions and and formulas described so far are needed to formulate the scene's reconstruction algorithm on the basis of the shape from defocus. The proposed iterative algorithm was written as alg. 2.7.

Algorithm 2.7 Shape from defocus via relative diffusion

Step 1: For two images I_1,I_2 of the scene obtain information about the parameters of the camera's optical system. Required calibration parameters: ν_1, ν_2, F, D, γ. Determine the value of regularization parameters α and κ. Determine threshold value ε of change E for stop condition.
Step 2: Initialize the depth map with a plane at depth $s_0 = \frac{(\nu_1+\nu_2)F}{\nu_1+\nu_2-2F}$
Step 3: Calculate the diffusion coefficient c via eq.(87), compute the partition $\{\Omega_+, \Omega_-\}$ via eq.(89) and eq.(90)
Step 4: Simulate (i.e. numerically integrate) eq.(81) and eq.(83)
Step 5: Using solution obtained at **Step 4** simulate eq.(98) and eq.(99)
Step 6: Compute the gradient of u and w and evaluate eq.(96), eq.(97), eq.(100), eq.(102)
Step 7: Update the depth map s by performing a time step of $\frac{\partial s}{\partial \tau} \triangleq -M(s)\left(E'_1(s)+E'_2(s)\right)-E'_3(s)$, with precomputed right-hand side.
Step 8: Return to **Step 3** until $\left\|E'_1(s)+E'_2(s)+M(s)^{-1}E'_3(s)\right\| \leqslant \varepsilon$ otherwise stop
Result: Depth map s

An example results of algorithm can be seen in Fig. 23. The visible edges' roundings are caused by used L_2 norm for the regularization term. In the picture results of gradient pre-conditioning are also visible.

The newer publication [79] contains the description of a more complex, two-step SfD method, which allows obtaining a dense depth map in real time. The first stage is space discretization and approximation of its shape at certain distances from the camera (space slicing by equifocal planes). The second stage is to use the calculus of variations to obtain a dense and continuous approximation of the scene's shape. The result of the first stage of the algorithm is the starting point for the variational method. The precise initialization is required, because of the fact that the proposed by the authors functional is not convex. An important characteristic of the described two-step method is the

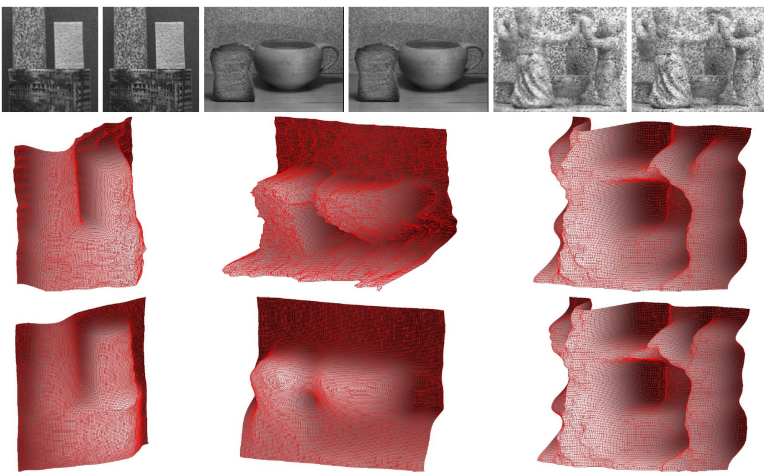


Fig. 23. The results of the Shape from Defocus via relative diffusion method [67] for three pairs of input images (top row). The scene's model is obtained without pre-conditioning conditioning (middle row) and with pre-conditioning (bottom row)

possibility of parallelizing most calculations. Thanks to the effective implementation on GPU (Nvidia GTX 460, 1 GB), the total time of the algorithm's functioning is below 0,25 s for images of 640×480 resolution in pixels. It is worth to point out that the common feature of SfD methods is low precision in the reconstruction of scenes which do not have textures of high constituent frequencies.

3. The summary

Even though there are clear differences between the algorithms we may point out a general trend of passive methods' development, which consists transition from two-image methods to multi-image methods. Reconstruction algorithms based on common information contained in the multiple images (bundle adjustment, shape from photo-consistency - calculus of variations) are generally more accurate and more resistant to disadvantageous phenomenon occurring in the image (such as noises and occlusions). Less computationally expensive methods have been successfully implemented on mobile devices. Stereo-vision and shape from motion often form the basis for navigation in a low-cost mobile robots. Directions of development of the 3D reconstruction algorithm determines to a large extent available computational power. A skillful use of GPU allowed obtaining

a dense depth map in real time for the whole visible scene. Implementation of variational methods using mass parallel processing is currently the main direction of research on passive 3D reconstruction methods.

The computer vision and passive 3D scene's reconstruction methods are an intensively developing discipline of knowledge. The article contains only reviews of a few chosen methods and key ideas which appeared in the publications about the three-dimensional modeling of the scene. The presented details of algorithms allows to understand the problems regarding the 3D reconstruction and point out methods on how to solve them. The article's aim was to acquaint the reader with the basic notions and key concepts which appeared in the literature. We hope that the information which is contained here will constitute a good starting point for further, more thorough, independent analysis of the issues described in this article.

References

1913

[1] Kruppa E.: Zur Ermittlung eines Objektes aus zwei Perspektiven mit innerer Orientierung. math.naturw.Abt.IIa, 122:1939-1948.

1974

[2] Baumgart B. G.: Geometric Modeling for Computer Vision. Stanford University.

1981

[3] Fischler M. A : Bolles R. C., Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM, 24:381-395.

[4] Longuet-Higgins H. C.: A computer algorithm for reconstructing a scene from two projections. Nature, 1981, 133-135.

1988

[5] Kass M., Witkin A., Terzopoulos D.: Snakes: Active contour models. International Journal of Computer Vision, 1:321-331.

[6] Subbarao M.: Parallel Depth Recovery By Changing Camera Parameters. Second International Conference on Computer Vision, 1988, p. 149-155.

1992

[7] Tomasi C., Kanade T.: Shape and motion from image streams: a factorization method: full report on the orthographic case. Cornell University.

[8] Tomasi C., Kanade T.: Shape and motion from image streams under orthography: a factorization method. International Journal of Computer Vision, 9:137-154.

1993

[9] Poelman C. J., Kanade T.: A paraperspective factorization method for shape and motion recovery. DTIC Document.

[10] Szeliski R. : Rapid octree construction from image sequences. CVGIP: Image Understanding, 58:23-32.

1994

[11] Laurentini A. : The visual hull concept for silhouette-based image understanding. IEEE Transactions on Pattern Analysis and Machine Intelligence, 16:150-162.

[12] Subbarao M., SURYA G. : Depth from defocus: a spatial domain approach. International Journal of Computer Vision, 13:271-294.

1995

[13] Hartley R. I. : In defence of the 8-point algorithm. Proc. 5th International Conference on Computer Vision, p. 1064-1070.

[14] Nayar S. K., Watanabe M., Noguchi M. : Real-time focus range sensor. Proc. 5th International Conference on Computer Vision, p. 995-1001.

1996

[15] Collins R. T. : A space-sweep approach to true multi-image matching. Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR'96, p. 358-363.

[16] Sturm P., Triggs B. : A factorization based algorithm for multi-image projective structure and motion. Proc. European Conference on Computer Vision ECCV'96, p. 709-720.

[17] Triggs B. : Factorization methods for projective structure and motion. Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR'96, p. 845-851.

[18] Watanabe M., Nayar S. K. : Minimal operator set for passive depth from defocus. Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR'96, p. 431-438.

1997

[19] Seitz S. M., Dyer C. R. : Photorealistic scene reconstruction by voxel coloring. Proc. Conference on Computer Vision and Pattern Recognition, p. 1067-1073.

[20] Torr P. H. S., Murray D. W. : The development and comparison of robust methods for estimating the fundamental matrix. International Journal of Computer Vision, 24:271-300.

1998

[21] Rajagopalan A. N., Chaudhuri S. : Optimal recovery of depth from defocused images using an MRF model. Proc. 6th International Conference on Computer Vision, p. 1047-1052.

[22] Tomasi C., Manduchi R. : Bilateral filtering for gray and color images. Proc. 6th International Conference on Computer Vision, p. 839-846.

[23] Watanabe M., Nayar S. K. : Rational filters for passive depth from defocus. International Journal of Computer Vision, 27:203-225.

[24] Zhang Z. : Determining the epipolar geometry and its uncertainty: A review. International Journal of Computer Vision, 27:161-195.

1999

[25] Hartley R. I. : Theory and practice of projective rectification. International Journal of Computer Vision, 35:115-127.

[26] Isgro F., Trucco E. : On robust rectification for uncalibrated images. Proc. International Conference on Image Analysis and Processing, p. 297-302.

2000

[27] Favaro P., Soatto S.: Shape and radiance estimation from the information divergence of blurred images. Proc. European Conference on Computer Vision ECCV 2000, p. 755-768.

[28] Fusiello A., Trucco E., Verri A.: A compact algorithm for rectification of stereo pairs. Machine Vision and Applications, 12:16-22.

[29] Koch R., Pollefeys M., Van Gool L.: Realistic surface reconstruction of 3D scenes from uncalibrated image sequences. The Journal of Visualization and Computer Animation, 11:115-127.

[30] Kutulakos K. N., Seitz S. M.: A theory of shape by space carving. International Journal of Computer Vision, 38:199-218.

2001

[31] Boykov Y., Jolly M. P.: Interactive graph cuts for optimal boundary & region segmentation of objects in ND images. Proc. 8th IEEE International Conference on Computer Vision, ICCV 2001, 1, p. 105-112.

[32] Boykov Y., Veksler O., Zabih R.: Fast approximate energy minimization via graph cuts. IEEE Transactions on Pattern Analysis and Machine Intelligence, 23:1222-1239.

[33] Kuzu Y., Rodehorst V.: Volumetric modeling using shape from silhouette. Proc. 4th Turkish-German Joint Geodetic Days, p. 469-476.

[34] Oram D.: Rectification for any epipolar geometry. Pro. British Machine Vision Conference BMVC 2001, 1, p. 653-662.

[35] Weickert J., Schnorr C.: Variational optic flow computation with a spatio-temporal smoothness constraint. Journal of Mathematical Imaging and Vision, 14:245-255.

2002

[36] Cheung G. K. M.: Visual hull construction, alignment and refinement across time. Technical Report CMU-RI-TR-02-05, Carnegie Mellon University.

[37] Favaro P., Soatto S.: Learning shape from defocus. Proc. European Conference on Computer Vision ECCV 2002, p. 823-824.

[38] Jin H., Favaro P.: A variational approach to shape from defocus. Proc. European Conference on Computer Vision ECCV 2002, p. 18-30.

[39] Matusik W., Buehler C., Mcmillan L., Gortler S. J.: An efficient visual hull computation algorithm. MIT LCS Technical Memo 623, MIT Laboratory for Computer Science, Cambridge.

[40] Tarini M., Callieri M., Montani C., Rocchini C., Olsson K., Persson T.: Marching Intersections: An Efficient Approach to Shape-from-Silhouette. Proc. Vision, Modeling, and Visualization Conference VMV 2002, p. 283-290.

2003

[41] Cheung K. M. G., Visual hull construction, alignment and refinement for human kinematic modeling, motion tracking and rendering. Citeseer.

[42] Cheung G. K. M, Baker S., Kanade T.: Visual hull alignment and refinement across time: A 3D reconstruction algorithm combining shape-from-silhouette with stereo. Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2, p. 375-382.

[43] Grauman K., Shakhnarovich G., Darrell T.: A bayesian approach to image-based visual hull reconstruction. Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1, p. 187-195.

[44] Hartley R., Zisserman A.: Multiple view geometry in computer vision. Cambridge University Press.

[45] Hemayed E. E., A survey of camera self-calibration. Proc. IEEE Conference on Advanced Video and Signal Based Surveillance, p. 351-357.

2004

[46] Hernandez E. C., Schmitt F. : Silhouette and stereo fusion for 3D object modeling. Computer Vision and Image Understanding, 96:367-392.

[47] Nister D. : An efficient solution to the five-point relative pose problem. IEEE Transactions on Pattern Analysis and Machine Intelligence, 26:756-770.

2005

[48] Bruhn A., Weickert J., Feddern C., Kohlberger T., Schnorr C. : Variational optical flow computation in real time. IEEE Transactions on Image Processing, 14:608-615.

[49] Franco J. S., Boyer E. : Fusion of multiview silhouette cues using a space occupancy grid. Proc. 10th IEEE International Conference on Computer Vision ICCV 2005, 2, p. 1747-1753.

[50] Mallon J., Whelan P. F. : Projective rectification from the fundamental matrix. Image and Vision Computing, 23:643-650.

[51] Mercier B., Meneveaux D.: Shape from silhouette: Image pixels for marching cubes. Vaclav Skala-UNION Agency.

2006

[52] Rzeszotarski D., Strumillo P., Pelczynski P., Wiecek B., Lorenc A. : Stereovision system for 3D reconstruction of image sequneces (in Polish). Zeszyty Naukowe Elektronika, 10:165-184.

[53] Guan Li, Sinha S., Franco J. S., Pollefeys M. : Visual hull construction in the presence of partial occlusion. Proc. 3rd International Symposium on 3D Data Processing, Visualization, and Transmission, p. 413-420.

[54] Kangni F., Laganiere R. : Projective rectification of image triplets from the fundamental matrix. Proc. IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP 2006, 2, p. II

[55] Michoud B., Guillou E., Bouakaz S. : Shape from silhouette: Towards a solution for partial visibility problem. Proc. of Eurographics, p. 13-16

[56] Montenegro A. A., Velho L., Carvalho P. C. P., Sossai J. : Polygonization of volumetric reconstructions from silhouettes. Proc. 19th Brazilian Symposium on Computer Graphics and Image Processing, SIBGRAPI 06, p. 11-18.

[57] Snavely N., Seitz S. M., Szeliski R. : Photo tourism: exploring photo collections in 3D. ACM transactions on graphics (TOG), 25:835-846.

[58] Szeliski R., Zabih R., Scharstein D., Veksler O., Kolmogorov V., Agarwala A., Tappen M., Rother C. : A comparative study of energy minimization methods for markov random fields. Proc. European Conference on Computer Vision ECCV 2006, p. 16-29.

[59] Weickert J., Bruhn A., Brox T., Papenberg N. : A survey on variational optic flow methods for small displacements. In: Mathematical models for registration and applications to medical imaging, O. Scherzer (Ed.). Springer. p. 103-136.

2007

[60] Favaro P. : Shape from focus and defocus: Convexity, quasiconvexity and defocus-invariant textures. Proc. IEEE 11th International Conference on Computer Vision ICCV 2007, p. 1-7.

[61] Hallmann I.: Lokalizacja robota mobilnego względem automatycznie wybieranych obiektów. Ph.D. Thesis. IPPT PAN, Warszawa.

[62] Kolmogorov V., Rother C. : Minimizing nonsubmodular functions with graph cuts-a review. IEEE Transactions on Pattern Analysis and Machine Intelligence, 29:1274-1279.

[63] Mileva Y., Bruhn A., Weickert J. : Illumination-robust variational optical flow with photometric invariants. Proc. 29th DAGM Symposium on Pattern Recognition, Lecture Notes in Computer Science Vol. 4713, p. 152-162.

[64] Pons J. P., Keriven R., Faugeras O. : Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. International Journal of Computer Vision, 72:179-193.

[65] Vogiatzis G., Hernandez C., Torr P. H. S., Cipolla R. : Multiview stereo via volumetric graph-cuts and occlusion robust photo-consistency. IEEE Transactions on Pattern Analysis and Machine Intelligence, 29:2241-2246.

2008

[66] Chica A., Williams J., Andujar C., Brunet P., Navazo I., Rossignac J., Vinacua A. : Pressing: Smooth isosurfaces with flats from binary grids. Computer Graphics Forum, 27:36-46.

[67] Favaro P., Soatto S., Burger M., Osher S. J. : Shape from defocus via diffusion. IEEE Transactions on Pattern Analysis and Machine Intelligence, 30:518-531.

[68] Fusiello A., Irsara L. : Quasi-euclidean uncalibrated epipolar rectification. Proc. 19th International Conference on Pattern Recognition ICPR 2008, p. 1-4.

[69] Kolev K., Cremers D. : Integration of multiview stereo and silhouettes via convex functionals on convex domains. Proc. European Conference on Computer Vision ECCV 2008, p. 752-765.

[70] Lin H.-Y., Wu J.-R. : 3d reconstruction by combining shape from silhouette with stereo. Proc. 19th International Conference on Pattern Recognition ICPR 2008, p. 1-4.

2009

[71] Franco J. S., Boyer E. : Efficient polyhedral modeling from silhouettes. IEEE Transactions on Pattern Analysis and Machine Intelligence, 31:414-427.

[72] Liansheng S., Jiulong Z., Duwu C. : Image rectification using affine epipolar geometric constraint. Journal of Software, 4:27.

2010

[73] Campbell N. D. F., Vogiatzis G., Hernandez C., Cipolla R. : Automatic 3D object segmentation in multiple views using volumetric graph-cuts. Image and Vision Computing, 28:14-25.

[74] Favaro P. : Recovering thin structures via nonlocal-means regularization with application to depth from defocus. Proc. IEEE Conference on Computer Vision and Pattern Recognition CVPR 2010, p. 1133-1140.

[75] Haro G., Pardas M. : Shape from incomplete silhouettes based on the reprojection error. Image and Vision Computing, 28:1354-1368.

[76] Lempitsky V. : Surface extraction from binary volumes with higher-order smoothness. Proc. IEEE Conference on Computer Vision and Pattern Recognition CVPR 2010, p. 1197-1204.

[77] Stuhmer J., Gumhold S., Cremers D.: Real-time dense geometry from a handheld camera. Proc. 32nd DAGM Symposium on Pattern Recognition, Lecture Notes in Computer Science Vol. 6376, p. 11-20.

2011

[78] Agarwal S., Furukawa Y., Snavely N., Simon I., Curless B., Seitz S. M., Szeliski R.: Building Rome in a day. Communications of the ACM, 54:105-112.

[79] Ben-Ari R., Raveh G.: Variational depth from defocus in real-time. Proc. IEEE International Conference on Computer Vision Workshops ICCV Workshops 2011, p. 522-529.

[80] Brox T., Malik J.: Large displacement optical flow: descriptor matching in variational motion estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 33:500-513.

[81] Delong A.: Advances in graph-cut optimization: multi-surface models, label costs, and hierarchical costs (Spine title: Advances in Graph-Cut Optimization). Citeseer.

[82] Wu Changchang, Agarwal S., Curless B., Seitz S. M.: Multicore bundle adjustment. Proc. IEEE Conference on Computer Vision and Pattern Recognition CVPR 2011, p. 3057-3064.

2012

[83] Haro G.: Shape from Silhouette Consensus. Pattern Recognition, 45:3231-3244.

[84] Valgaerts L., Bruhn A., Mainberger M., Weickert J.: Dense versus sparse approaches for estimating the fundamental matrix. International Journal of Computer Vision, 96:212-234.

2013

[85] Khan W.: Image Segmentation Techniques: A Survey. Journal of Image and Graphics, 1(4):166-170.

[86] Kim D., Ruttle J., Dahyot R.: Bayesian 3D shape from silhouettes. Digital Signal Processing, 23:1844-1855.

[87] Nieradka G.: Dopasowanie obrazów pary stereoskopowej z wykorzystaniem logiki rozmytej. Politechnika Warszawska, Wydział Elektroniki i Technik Informacyjnych.

[88] Peng B., Zhang L., Zhang D.: A survey of graph theoretical approaches to image segmentation. Pattern Recognition, 46:1020-1038.

[89] Singh M., Misal A.: A Survey Paper on Various Visual Image Segmentation Techniques. International Journal of Computer Science and Management Research, 2:1282-1288.

DYNAMIC PATH PLANNING WITH REGULAR TRIANGULATIONS

Petr Brož, Michal Zemek, Ivana Kolingerová, Jakub Szkandera
Faculty of Applied Sciences, University of West Bohemia, Pilsen, Czech Republic

Abstract. Path planning is a well known problem that has been extensively studied in many scientific disciplines. In general, it defines a task of finding a path between two given spots in an abstract environment so that the path satisfies certain criterion of optimality. Although there are many methods solving this objective, they usually assume the examined space does not change in runtime. Modern applications, however, do not have to meet these requirements, especially in case of virtual reality or computer games. Therefore, we propose a general model for real-time path planning in dynamic environment where the obstacles can nondeterministically appear, disappear, change the position, orientation or even shape. The model uses a triangulation for dynamic space subdivision among bounding spheres of the obstacles and a heuristic algorithm to repair an already found path after any change of the scene. The presented solution is the first one using regular triangulation. At the price of the suboptimal result, it provides an efficient and fast way to plan a path with the maximal clearance among the moving and changing obstacles. In comparison to raster based techniques and methods using the Delaunay triangulation (Voronoi diagram), it requires less time to preprocess and generates paths with a larger clearance.

Key words: path planning, path finding, motion planning, virtual reality, robotics, proteins, suboptimality, gaps filling.

1. Introduction

Path planning, a general problem of finding a path between two spots in a certain environment in such a way that the path satisfies one or more given objectives, is often considered to be one of the basic tasks in the computer science. However, it has been extensively studied much earlier as one of the fundamental and best known problems in the graph theory. There are therefore many sophisticated and efficient methods nowadays to solve this task in various applications, e.g., in computer networks, optimalization theory or design of data structures. In the context of the computational geometry, path planning usually involves additional procedures and characteristics such as definition, creation and maintenance of a certain representation of the environment, considering shape of the navigated entity, etc.

As there are many scientific disciplines involving the theory of path planning, various terms such as *path finding*, *route planning* or *motion planning* are sometimes used and distinguished, too. For example, the term path planning may refer to the procedure of planning an overall motion trajectory whereas the term path finding then defines the process of carrying out the path with a certain feedback for the planner. In our paper, all these terms refer to the same global problem in terms of the computational geometry.

Although the graph theory provides subtle fundamentals for many path planning algorithms,

these often suffer from serious disadvantages when they are to be used in modern applications, e.g., in the computer games or in the virtual reality (*VR*) applications. The conventional methods usually assume that the examined environment does not change in time and that its complete and detailed overview is available in advance. In the case of the modern and often indeterministic applications, it is essential to modify the existing path planning approaches or to define new ones. To provide a better solution for the mentioned type of applications, we introduce a new approach for a real-time path planning in a dynamic environment. The proposed approach is designed to suit the following requirements and characteristics:

- In terms of the dynamic environment, the new approach assumes that the obstacles can change their position, orientation or even shape in time and it is also able to register newly added or removed obstacles.
- Moreover, the behaviour of the obstacles can be either deterministic or indeterministic.
- Sometimes it is not necessary to find an optimal solution. Papers regarding the artificial intelligence actually suggest the so-called *tolerate imperfection* approach to simulate results more similar to those created by a human. On top of that, such an acceptance of non-optimal results obviously leads to more efficient solutions.

According to these specifications, the described solution is suited for a pseudooptimal navigation in an abstract environment with both static and dynamic obstacles where the obstacles may appear and disappear in time. To ensure the space subdivision among all obstacles, we use the regular triangulation with the possibility to add and remove generating points after its creation (see Section 4.1). A dual structure of the triangulation, the so-called *power diagram* (see Section 4.2), is then used to navigate among the obstacles.

The rest of the paper is organized as follows. Section 2 introduces the theoretical background of path planning and Section 3 then outlines both fundamental path planning methods and the most relevant approaches for the modern applications. Section 4 describes our proposed solution and provides a brief theoretical introduction to triangulations as they are part of the solution. In Section 5, the results of the proposed technique are presented and compared to the conventional methods. Finally, Section 6 surveys the most important characteristics of the presented algorithm and outlines possible ways of the future research.

2. Theory

As mentioned in Section 1, the term path planning in general defines a problem of finding a path between two given spots in a certain abstract environment representation [8]. In the context of the virtual reality, the term *avatar* is often used for the navigated entity.

Environment

The environment is represented in various ways (see further in this section), but all these can be generalized in the so-called *configuration space* or *c-space*, in other words, a graph $G(V,E)$ with a set of nodes V, $|V| = n$, representing all available states (in our case, positions in 2D/3D space)

and a set of edges E, $|E| = m$, defining all possible transitions between these states. The graphs are often differentiated according to various characteristics:

- Edges can be either non-directed or directed. The directed edges allow only for transitions in the given direction and non-directed edges allow for passing in both directions.
- Evaluation of the graph is ensured with an assessing function w giving each node or edge its weight.

$$w : V \to \mathbb{R} \text{ for node-weighted graphs}$$

$$w : E \to \mathbb{R} \text{ for edge-weighted graphs}$$

- Connection density – represents the number of edges in the graph. It is a very important characteristic, especially with regard to the computational complexity which can be $O(m)$ or $O(m^2)$. The graphs with a high number of edges are called *dense graphs*, whereas the graphs with a low amount of edges are usually called *sparse graphs*.
- Evaluation homogenity – distinguishes the graphs with a uniform weight distribution (the so-called *homogeneous edge costs*) and the graphs with a non-uniform weight distribution (the so-called *irregular edge costs*).

Path

The path is an acyclic sequence of the graph nodes. It is said to be optimal by satisfying one or more given objectives of optimality, e.g., the shortest path, the cheapest path or the path with the maximal clearance among all surrounding obstacles. The objectives usually result from the particular graph evaluation. For instance, the edges can be evaluated according to the time needed to traverse between their nodes or the nodes can be evaluated according to their distance from the closest obstacle.

In the area of path planning, the graphs usually have weighted and directed edges that do not form any cyclic closed path. These are called *directed acyclic graphs*.

Environment representation

The abstract environment representation for the path planning algorithms is divided into the following classes:

Graph representation – describes the given environment with a set of states and edges defining the possible transitions between the states (see Fig. 1(a)). In this case, the states represent discrete points in an n-dimensional space, e.g., the position coordinates in the 3D space.

Grid/raster representation – represents the environment with a final set of values in a matrix. Its dimension corresponds to the dimension of the represented environment. In this case, the path planning algorithms usually traverse the cells of the matrix (see Fig. 1(b)).

However, neither the virtual reality applications nor the computer games provide an environment representation immediately prepared for the path planning process. In general, the only information provided is a list of obstacles with their shapes. It is then necessary to construct the final environment representation in a certain way.

Graph representation

In the case of graph representation, the so-called *visibility graphs* (*VG*) are often used. The nodes of such a graph correspond to the vertices of all obstacles in the environment[1] and the edges connect the nodes that can "see" each other. In other words, the edges connect every pair of nodes where the line connecting these nodes does not intersect any edge (in 2D space) or face (in 3D space) of any obstacle, see Fig. 1(a). Apart from the visibility graph, other techniques and structures are used such as the *quadtrees*, the *kD-trees*, the *BSP trees* or various triangulations as we will see in Section 4.1.

Raster representation

The raster representation provides several ways to interpret the values in the matrix. The simplest way is the use of logical values to define whether there is any obstacle presented in the spot of the corresponding cell. In a more frequently used technique, the so-called *image distance transform*, the values in the matrix are specified according to the distance of a particular cell from the nearest obstacle. An information hereby received, the so-called *potential field*, then faciliates to find a path with the maximal clearance among the surrounding obstacles.

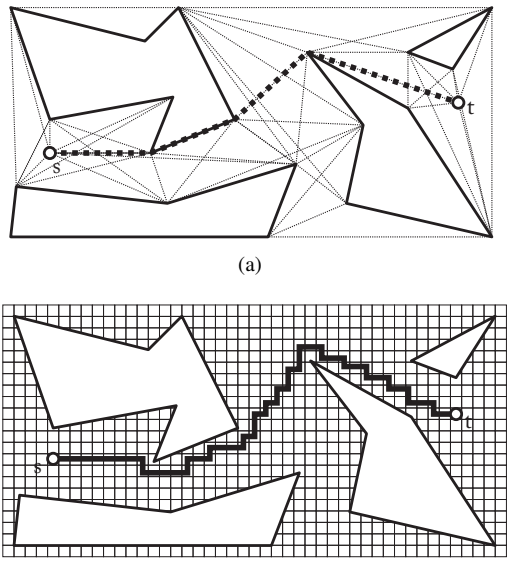


Fig. 1. (a) An example of the visibility graph. (b) An example of the grid path planning.

[1]The nodes of the visibility graph also contain the end-points of the sought path as shown in Fig. 1(a).

3. State of the art

3.1. Classification of the algorithms

As the basic path planning problem is well known since the beginning of the graph theory, there are many algorithms and techniques that need to be categorized in some way. The following list presents the most important characteristics and properties for this classification:

Extent of the desired results – determines whether a particular algorithm is supposed to return the very single path between the two given nodes (*single-pair* algorithms), to find all paths from the given source node (*single-source* algorithms) or to provide all possible paths among the graph nodes (*all-pairs shortest path* algorithms).

Environment characteristics – form the second most important factor in the classification of path planning algorithms. These characteristics are, e.g., the knowledge of the environment (known, partially-known or unknown environment), the already mentioned environment representation, its preliminary depiction etc. In case of using the c-space representation, other characteristics arise such as the possibility of a negative edge evaluation.

Avatar specification – is also a notable factor as some path planning approaches assume the navigated avatar to be represented only with a point, whereas other techniques consider its size, shape or amount of avatars at the same time.

Finding only a part of the path – is another option to be considered while planning a path in an environment that is very likely to change in time.

Suboptimal path planning – represents an approach for the applications prioritizing speed over quality.

3.2. Conventional methods

Single source algorithms

The single source (SS) algorithms are used to find optimal paths from the given source node to all other nodes of the graph. Such a solution is often obtained with the so-called *minimum spanning tree* (MST). The basic approach to construct the MST is called a *relaxation* [15].

Although the computational complexity of the MST construction is $O(2^m)$, other algorithms, obviously more efficient ones, are based just on this technique. E. W. Dijkstra formalized the problem of finding an optimal path in graph in 1956 and presented its solution for the graphs with positive edge evaluation. The solution was then generalized in 1986 into the so-called *path planning algorithm prototype*. The prototype uses a list of open nodes, that means, a list of the nodes intended for the processing. In the beginning of the prototype algorithm, the list is empty and the only node inserted is the start node of the path. As long as there are any nodes in the list, the algorithm chooses one of them (to be more precise, the node is removed from the list) and checks whether this node provides a shorter path to any of its adjacent nodes. The approach is outlined in Algorithm 3.1 where the implementation of methods `Insert` and `Select` is subject to specific modifications in the particular techniques.

Algorithm 3.1 Path planning algorithm prototype

Input: *graph* contains a final set of nodes and edges {Each node contains a reference to the preceding node in the searched path (*node.parent*) and a value defining its minimum distance from the starting point of the path to the current node (*node.distance*); each edge contains reference to its end-points (*edge.start* and *edge.end*)}
Input: *root* is one of the nodes in *graph*
Input: *list* is an empty list of nodes

```
list.Insert(root)
while list is not empty do
  node ← list.Select() {The selected node is removed from the list}
  for all edge in graph.edges do
    dist ← edge.start.distance + distance_between(edge.start, edge.end)
    if edge.end.distance > dist then
      edge.end.parent ← edge.start
      edge.end.distance ← dist
      list.Insert(edge.end)
    end if
  end for
end while
```

Dijkstra's algorithm does not need the `Insert` method and uses the so-called *greedy* approach to select the node to be processed next – the algorithm always uses the node with the minimal distance from the source node. If a sequential search is used to find the closest node, the overall computational complexity is $O(n^2)$. Therefore, various data structures have been presented to speed up the technique, e.g., the *d-Heaps* [5]. For the graphs where the negative evaluation is also allowed, the Bellman-Ford's algorithm is used [9] with the complexity $O(mn)$. It however tends to the complexity $O(m^3)$ for very dense graphs according to the definition from Section 2.

All-pairs shortest path algorithms

The all-pairs shortest path (*APSP*) algorithms find the optimal paths between every pair of nodes in the graph. As the construction of a MST for every single node would be very inefficient, the so-called *all-pairs shortest distance* algorithms are rather used to provide only the lengths of the optimal paths between each pair of nodes. It is possible to apply the Dijkstra's algorithm or the Bellman-Ford's algorithm, respectively, for each node of the graph, however, with the overall computational complexity $O(n^3)$, $O(n^4)$, respectively. A better solution for this task, Floyd-Warshall's algorithm [9], works with the so-called *distance matrix* and *matrix of precursors*.

It is known that the Floyd-Warshall's algorithm works with the complexity $O(n^3)$. It is also possible to use this approach for the graphs with negative evaluation of the edges but without any cycles. However, improving the computational complexity turned out to be a significant

problem. An algorithm combining the Dijkstra's and Bellman-Ford's algorithm has been presented in 1977. The improvement works in $O(n^2 \log n + mn)$ under the assumption of a sparse and oriented graph without any cycles. Without this assumption, the only way to break the cubic complexity is an approximation of the desired path, often denoted as the *t-optimal path*. Such a path can be at most n-times worse than the optimal solution. For instance, Don, Halperin and Zwick presented an efficient algorithm for finding a 3-optimal path [12].

Single pair shortest path algorithms

The single pair shortest path (*SPSP*) algorithms form the basic class in the whole problematics of path planning. In this case, only a single path between the two given nodes is required. The best known techniques in this area [9] are the *depth first search* (*DFS*) and the *breadth first search* (*BFS*). From the theoretical point of view, the single source algorithms and the single pair algorithms have the same computational complexity. However, the single pair algorithms are usually more efficient as the single solution is found earlier.

A* algorithm [15], the most significant representative in this class of algorithms, uses certain heuristics to underestimate the nodes selected for the next processing. In this case, the heuristic function is defined as $f(u) = g(u) + h(u)$, where $g(u)$ is the minimum distance between the start node and the currently selected node $u \in V$, $h(u)$ is the heuristic defined by the Euclidean distance from the end node to the currently selected node $u \in V$. A node that is closer to the destination has a bigger chance of being selected for the next processing. The A* algorithm is also generalized into the so-called *best-first search*.

3.3. Advanced methods

In this section, we briefly describe the most important representatives from the wide range of advanced path planning applications available nowadays.

D* algorithm

The D* algorithm [6] modifies the A* algorithm and is suited for graphs whose evaluation may change in time of its traversal. It is also suited for unknown, partially known or varying environments. The D* algorithm uses the same list of open nodes as the single source techniques and propagates every change of evaluation throughout this list. Each node keeps an information about the smallest estimated distance to the destination node called *key function*, which is also used to sort the nodes in the list. Any evaluation change is then propagated to all appropriate key functions and causes reordering of the nodes in the list. The algorithm provides an optimal solution by planning a new path after each change in the environment. Although this approach corresponds to the brute-force solution, it is more effective.

Path planning system for strategic games

A good example of a path planning approach for navigating several avatars at the same time has been presented in [14]. It distinguishes static obstacles, avatars waiting for the orders and avatars that are currently in motion. In the preprocessing time, the visibility graph (see Fig. 1(a))

is created for the static obstacles. The waiting avatars are gathered into clusters and the initial path found in the graph is then adjusted to avoid the collision with any of these clusters. In the time of executing the movement, possible collisions between the moving entities are avoided by temporarily stopping one of these entities until its path is free again.

Space discretization for efficient human navigation

The approach proposed in [11] uses the rasterization technique described in Section 2. The input scene is discretized into a certain matrix consisting of cells of the same size, the so-called *uniform cells*. Inside these cells, the optimal path is then found using the mentioned A* algorithm. As the desired path should serve for human-like avatars, the basic criterion for the resulting path is its accessibility for the humans. In other words, the path cannot lead through the air. An example of such a path is shown in Fig. 2(a).

Probabilistic roadmaps approach

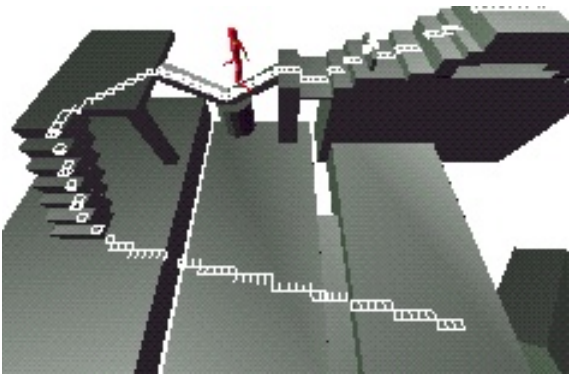
The probabilistic roadmap [16] is one of the frequently used techniques nowadays. The so-called *probabilistic roadmap planner* randomly samples the input scene for collision-free positions and adds these positions as nodes to a specific roadmap graph structure (see Fig. 2(b)). In the following step, the planner chooses various pairs of *promising nodes* and attempts to connect them with a local planner until the c-space is fully covered. The exact implementation details (e.g., the space sampling, the selection of promising nodes) are then subject to particular variants of this path planning approach.

Voronoi-based route graphs

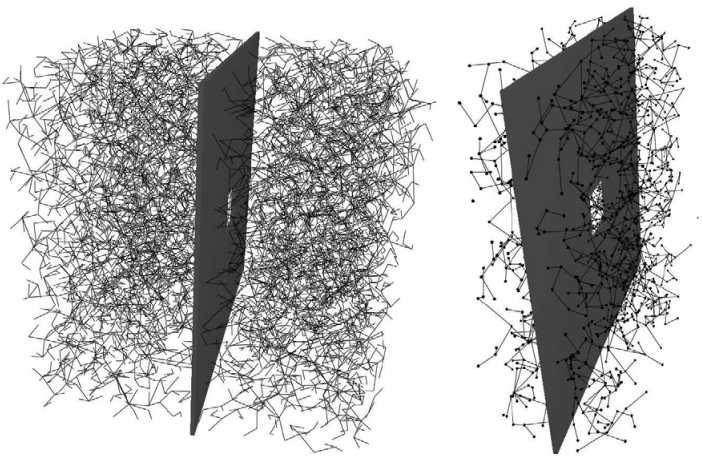
The hierarchical Voronoi-based route graph [20] defines a spatial representation of the environment for the purposes of robot navigation. For any input scene consisting of polygonal obstacles, a *generalized Voronoi diagram* (*GVD*) is constructed as shown in Fig. 3 and its edges are then used for the navigation along the path with the maximal clearance among all the obstacles. Such a construction usually consists of creating a particular *Voronoi diagram* (*VD*) for the vertices of every single obstacle in the scene and consequently removing all the unnecessary edges.

Grid and graph representation

As the last example of modern solutions in the area of path planning, we present our previous research [23] which is compared to the newly proposed solution in Section 5. The approach developed in the previous research combines both raster and graph representations of the environment. While the raster defines a certain danger (in this case, the danger is represented by the proximity to the nearest obstacle and therefore by the probability of a collision) of the scene in particular positions (see Figs 4(a), 4(b), the graph as an adaptive spatial structure refines or coarses itself according to this danger in the corresponding areas (see Fig. 4(c)). The adaptive graph structure then represents the c-space and allows the approximate navigation along the least dangerous path.



(a)



(b)

Fig. 2. (a) A discretized path planning for human-like avatars [11]. (b) An example of a space sampling using the probabilistic roadmaps [16].

4. Proposed solution

In the case of dynamic applications of virtual reality or computer games, not even the modern path planning methods often provide a sufficient solution, mainly because of the following reasons:

- in our focus, we assume the navigation to be executed in E^3 space. This assumption may

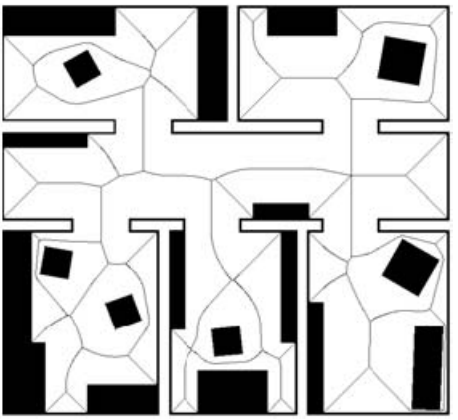
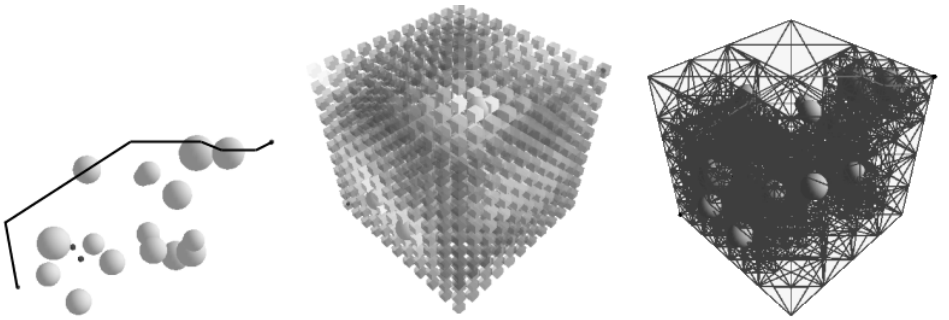


Fig. 3. Generalized Voronoi diagram for an example scene [20]



(a) (b) (c)

Fig. 4. (a) An example scene. (b) Its raster representation. (c) An adaptive mesh refined according to the raster representation in Fig. 4(b).

cause serious complications in the techniques based on a discrete representation, especially in the terms of memory consumption;

- the modern algorithms based on the graph representation usually construct the c-space according to the exact shapes of the obstacles. As it may significantly affect the computational complexity, we prefer to provide an efficient solution at the cost of the optimality;
- some of the methods described in Section 3.3 are not able to operate in an undeterministic environment which is obviously a very frequent case in the VR applications.

Therefore, we propose a solution that uses a triangulation to navigate among bounding

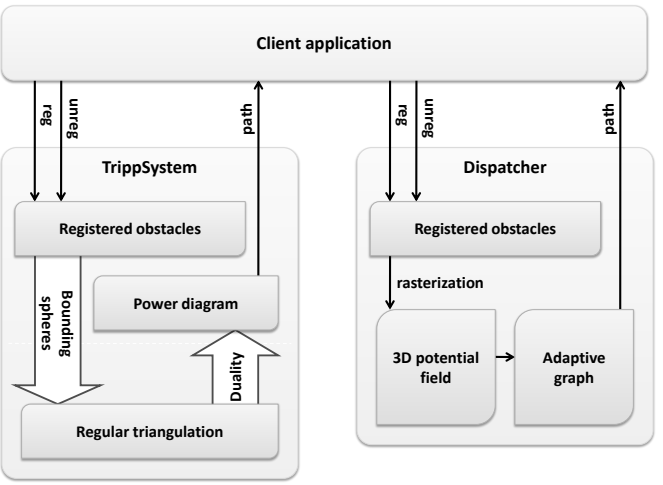


Fig. 5. A design of the proposed path planning solution.

spheres of the obstacles in the scene. It is a prime approach in using the *regular triangulation* (*RT*) and the correspondent dual structure, the *power diagram* (*PD*), instead of the Voronoi diagram. As we will show in Section 4.1, the RT makes it posible to define certain weights of the points that generate the triangulation. In our proposal, the generating points and their weights are defined by each bounding sphere. The corresponding PD then defines the c-space and allows us to plan a path with the maximal clearance among the bounding spheres. Moreover, the implemented RT provides the possibility to insert, remove and change the generating points in the runtime, thus allowing us to react to any change in the scene. The overall design of this approach is outlined in Fig. 5. The block labeled *TrippSystem* represents the new approach based on the triangulations (triangulation based path planning system), whereas the second block labeled *Dispatcher* represents our previous solution combining raster and graph representations of the environment, see the last paragraph in Section 3.3.

The main process of planning a path in the c-space is ensured by the so-called *gaps filling algorithm* that will be described in Section 4.3. After any change in the scene, the gaps filling algorithm repairs the previously found path instead of planning a new one from the scratch.

We provide a brief theoretical background for the regular triangulations and related structures (Sections 4.1, 4.2) and describe the heuristic algorithm used to maintain a suboptimal path in the varying environment (Section 4.3).

4.1. Regular triangulation

The regular triangulation (RT) is a generalization of well-known Delaunay triangulation (DT). Each point in the regular triangulation is associated with a real number – a weight of the point. If the weights of the points are equal, then the regular triangulation and the Delaunay triangulation of this set of points are identical, otherwise the triangulations can be different. Further we give a definition of the regular triangulation (see [4]).

Triangulation – Given a set of points S in E^3, the triangulation $T(S)$ of this set of points is a set of tetrahedra such that:

- a point $p \in E^3$ is a vertex of a tetrahedron in $T(S)$ only if $p \in S$;
- the intersection of two tetrahedra of $T(S)$ is either empty or it is a shared face, a shared edge or a shared vertex;
- the union of all tetrahedra in $T(S)$ entirely fulfills the convex hull of S.

Note that in this definition of $T(S)$ we do not require each $p \in S$ to be a vertex of $T(S)$.

Weighted point – A point $p \in E^3$ with an associated weight $w_p \in \mathbf{R}$ is called a weighted point. If the weight w_p is non-negative, then p can be interpreted as a sphere centered at the point p with a radius $\sqrt{w_p}$.

Power distance – A power distance of a weighted point p from a point $x \in E^3$ (no matter whether x is weighted or unweighted) is defined as $\pi_p(x) = |px|^2 - w_p$, where $|px|$ denotes the Euclidean distance between the points p and x. The power distance $\pi_p(x)$ can be interpreted as a square of length of a tangent from the point x to a sphere centered at p and with the radius $\sqrt{w_p}$ (if x lies outside this sphere), see Fig. 6a.

Orthogonal points – Two weighted points p and q are said to be orthogonal if $|pq|^2 = w_p + w_q$, i.e. $\pi_p(q) = w_q$, see Fig. 6b.

Orthogonal center – Let a, b, c, d be non-coplanar weighted points. A weighted point z is an orthogonal center of a tetrahedron $abcd$ if z is orthogonal to the points a, b, c, d.

Global regularity – Let z be the orthogonal center of a tetrahedron $abcd$. The tetrahedron $abcd$ is globally regular with respect to a set of weighted points S if $\pi_z(p) > w_p$ for each point $p \in S - \{a, b, c, d\}$.

Regular triangulation – A triangulation $T(S)$ is a regular triangulation of S ($RT(S)$) if each tetrahedron in $T(S)$ is globally regular with respect to S.

Redundant point – A point p, $p \in S$, is called a redundant point if no globally regular tetrahedron $pabc$ exists in $RT(S)$. Redundant points are not vertices of any tetrahedron in $RT(S)$, therefore, the vertex set of $RT(S)$ is generally only a subset of S.

Basic characteristics

Similarly to the Delaunay triangulation, the regular triangulation of S is unique if the points of S are in a general position. In contrast to DT, it is possible that some point $p \in S$ is not a vertex of $RT(S)$. From the given definition of $T(S)$ it follows that each point $p \in S$ lying on the convex hull $CH(S)$ of S is a vertex of $RT(S)$ – otherwise the union of tetrahedra would not fulfill $CH(S)$.

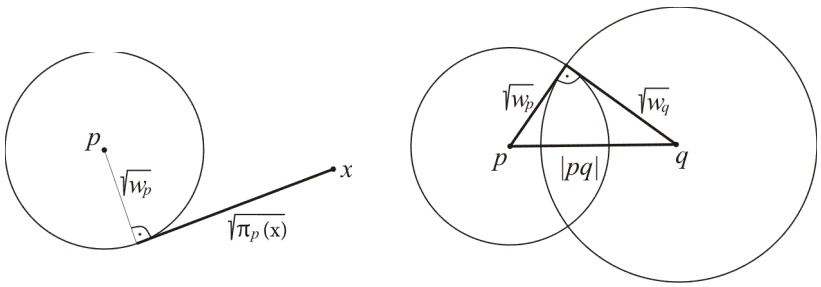


Fig. 6. (a) The power distance $\pi_p(x)$. (b) Two orthogonal weighted points.

Shewchuk [19] proved that in the worst case the number of tetrahedra of $RT(S)$ is $O(n^2)$, where $n = |S|$. This happens if the points of S lie on (or nearby) two nonintersectors. If the points of S are uniformly or nearly uniformly distributed, the expected number of tetrahedra grows with n nearly linearly.

Dynamic triangulation

As already mentioned, our intended scene is dynamic, thus obstacles can appear, disappear, discretely change their size or position. And an underlying regular triangulation must handle such events. It is not necessary to rebuild a whole triangulation after each change. All the four types of changes of an obstacle (a weighted point) in the triangulation can be implemented as an insertion, deletion or a combination of both (when an obstacle changes its size or position, it is first removed from the triangulation and then re-inserted). Therefore we use an algorithm of an incremental insertion (see e.g. [4, 1]) for the triangulation construction. Here, points are inserted into the regular triangulation one by one and the regularity of the triangulation is restored after each insertion. The time complexity of this algorithm is $O(n^2)$ in the worst case. In the average case, the time complexity is of our implementation is $O(n^{5/4})$.

For the deletion of a point from a triangulation, we employ the algorithm described in [18]. This algorithm removes all tetrahedra incident to a point to be deleted. This creates a cavity - a star-shaped polyhedron P. The cavity is then retriangulated by a successive cutting of ears of the P. The time complexity of the deletion of one point is $O(kg)$, where k is a number of vertices of P and g is a number of tetrahedra created to retriangulate P. Note that g is $O(k^2)$ in the worst case and $O(k)$ in the average case. Other applicable algorithms of a point removal are described in [17, 26, 22].

4.2. Power diagram

Power cell – Given a set of weighted points S in E^3. For each weighted point $p \in S$, its power cell is defined as $power\ cell(p) = \{x \in E^3 | \forall q \in S - \{p\} : \pi_p(x) \leq \pi_q(x)\}$. The point p is

the so-called *generator* of *power cell*(p). Observe that *power cell*(p) is a convex polyhedron and the union of all *power cells*(p), $p \in S$, covers E^3. An intersection of two or more power cells is either empty or forms:

- a planar convex polygon – a face of $PD(S)$. A face of $PD(S)$ is the intersection of two power cells;
- a line segment or a half line – an edge of $PD(S)$. An edge of $PD(S)$ is the intersection of at least three power cells;
- a point – a vertex of $PD(S)$. A vertex of $PD(S)$ is the intersection of at least four power cells and the sphere orthogonal to the generators of these power cells is centered in this vertex of $PD(S)$.

Power diagram – The power diagram of S is a collection of all power cells, their faces, edges and vertices.

Thanks to the duality, power diagrams could be seen as another representation of regular triangulations and vice versa. Also each of these structures can be derived from the other in linear time. Any computation performed on one of these structures can be also done on the other structure and with the same time complexity. For example, a path in $PD(S)$ – a sequence of vertices and edges – can be represented as a sequence of tetrahedra and faces in $RT(S)$.

From the point of view of the path planning, the power diagrams (in comparison with the ordinary Voronoi diagrams) offer one big advantage. Let the power cells of two weighted points $p,q \in S$ share a face. If we interpret the points p,q as spheres with the radii $\sqrt{w_p}, \sqrt{w_q}$, respectively, and these spheres do not intersect each other, then the shared face does not intersect any sphere from S.

Power diagram and c-space

We interpret a power diagram $PD(S)$ as a c-space $G(V,E)$ as follows:

- each edge $h \in PD(S)$ corresponds to one node $v_h \in V$;
- two nodes $v_h, v_k \in V$ are connected by an edge $e \in E$ if and only if the two corresponding edges $h,k \in PD(S)$ share a common vertex;
- a weight w of a node of $v_h \in V$ is computed as

$$w = \frac{1}{1+d},$$

where d is a euclidean distance between the edge $h \in PD(S)$ and the closest weighted point $p \in S$, i.e.,

$$d = min_{x \in h, p \in S}\{|xp| - \sqrt{w_p}\}.$$

Therefore, $w \in \langle 0,1 \rangle$ and a value of w corresponds to a probability of a collison. An exact computation of the distance d is not trivial and in the worst case can take $O(n)$ time, therefore we approximate d by a radius of a so-called bottleneck sphere of h. A bottleneck sphere of h is the biggest sphere, which can travel along h from one its endpoint to the other without intersecting

generators of power cells forming h (for details, see [27]). The computation of a bottleneck sphere is a $O(1)$ operation. The drawback is that a bottleneck sphere still can intersect some generators from S. But sizes of such intersections tend to be small. Note that if the approximate distance is positive, the exact distance must be also positive.

4.3. Gaps filling

The gaps filling algorithm [24] provides a fast way to find a suboptimal path in graphs with varying topology or evaluation. Instead of finding a new path after any change in the graph, this approach repairs the last found path along the nodes that have been removed and the nodes whose evaluation has been made worse. An example of such a procedure is presented in Fig. 7. First, a new path was found between the nodes s,t using the standard Dijkstra's algorithm and in the following iteration, after worsening of the evaluation in the nodes a,b, these nodes have been omitted and the resulting path segments have been connected again using the Dijkstra's algorithm. The procedure is outlined in Algorithm 4.1.

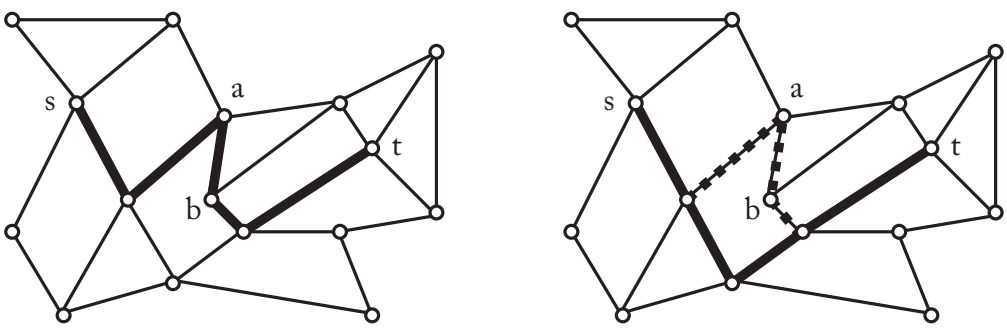


Fig. 7. An example of the gaps filling approach [24].

At the cost of the suboptimal solution, the algorithm offers a significant speed-up in terms of the number of processed nodes during the search. Tab. 1 shows the speed-up compared to the standard Dijkstra's algorithm – in a planar graph of 32x32 nodes which are randomly connected to some of the neighbouring nodes, the evaluation has been changed in a certain percentage of the nodes. The second column shows the percentage of nodes processed for re-planning the path by the gaps filling algorithm in comparison with the Dijkstra's algorithm. The last column then specifies the percentual change of the overall path quality, an average of weights of all nodes on the path where the minimum weight is the goal.

Algorithm 4.1 Gaps filling algorithm

Input: a graph $G(V,E)$, a path $P \subset V$, ending nodes $s,t \in P$
Output: new path P'

```
P' ⇐ {s}
u ⇐ NULL {Auxiliary node}
for all n ∈ P − {s} do
  if n was removed ∨ n has worse evaluation then
    if u == NULL then
      u ⇐ predecessor of the node n on the path P
    end if
  else
    if u == NULL then
      P' ⇐ P' ∪ {n}
    else
      P' ⇐ P' ∪ path(u,n) {Finding a path from u to n using Dijkstra's algorithm}
      u ⇐ NULL
    end if
  end if
end for
```

Tab. 1. Speed-up of the gaps filling algorithm.

% nodes changed	% nodes processed	% weight of path
25	13.70	126.26
50	37.75	115.76
75	85.67	108.67

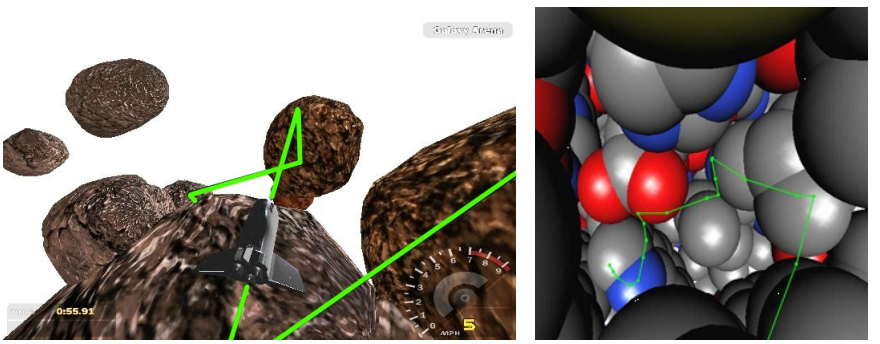
5. Experiments and results

5.1. Implementation

The proposed path planning solution was implemented as a dynamically linked library in the C# language in order to enable its easy use in modern applications while allowing us to continue with efficient development and debugging. For testing purposes, a simple game application was created, too, using the XNA Game Studio. The demo simulates a spaceship flying through a swarm of asteroids where some of these asteroids are moving, see Fig. 8(a). The ship may be controlled by the user or by an automatic navigation, in other words, by our implemented approach. In addition to the visual presentation, the game application was also used to measure

and gather various characteristics of the path planning algorithm for different types of behavior of the environment. The characteristics are presented and described in the following sections.

The proposed method was also tested in the context of real biochemistry data. To be more precise, the technique described in this paper was used to plan a path in a dynamic environment of protein molecules, see Fig. 8(b). In this case, movement of the proteins was defined in a discrete way by providing their location in different time slices. Results of this measurement are presented in Section 5.5.



(a) (b)

Fig. 8. (a) A screenshot from the testing application Galaxy Wars. (b) An example of a path in a protein molecule.

In case of the path planning algorithm itself, we have tested several options (with the correspondent notation in the following charts):

- Type of the triangulation for the space subdivision:
  - regular triangulation (denoted with *RT*);
  - Delaunay triangulation (denoted with *DT*).
- Accuracy of the so-called *in-circle test* used in the triangulation alteration:
  - exact arithmetics (denoted with *EA*);
  - non-exact arithmetics (denoted with *NEA*).

Finally, the following list summarizes the most important characteristics we have observed during the tests:

Preprocessing time – It is often possible and very efficient to preprocess the available information about the environment and we have considered this property for all the tested algorithms.

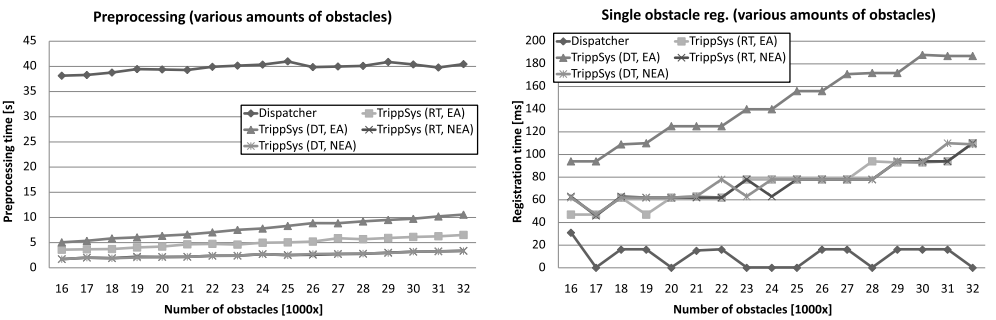
Single registration time – In our proposal, the adaptation to the changes of the scene is ensured by re-inserting the generating point of the regular triangulation (see Section 4). We have therefore measured the time needed to insert a single point into the triangulation.

Path weight – An average path weight is defined as the arithmetic mean of all weights of the path nodes. Weight of any node represents its proximity to the nearest obstacle, higher weight

means a higher danger of collision with the obstacle. The maximum danger value of 1.0 then defines a collision state with the obstacle.

5.2. Various numbers of obstacles

Dependence of the speed and quality of the provided results on the number of obstacles is obviously the most important characteristic in the context of path planning algorithms. In Figs 9(a) and 9(b), we present the preprocessing and registration times for various configurations of the proposed solution (TrippSys) together with the solution based on graph and grid representation of the environment (Dispatcher, see Section 3.3). It can be seen that the triangulation based techniques provide shorter preprocessing times than the conventional approach, even for higher numbers of obstacles. On the other hand, the time needed to insert an obstacle to the triangulation that already consists of a given number of generating points is greater than in the conventional case. In Dispatcher, every single obstacle is projected on the grid structure and consequently causes the adjustment of the adaptive graph. The time needed to insert an obstacle in such a manner therefore does not change with the increasing number of already registered obstacles.

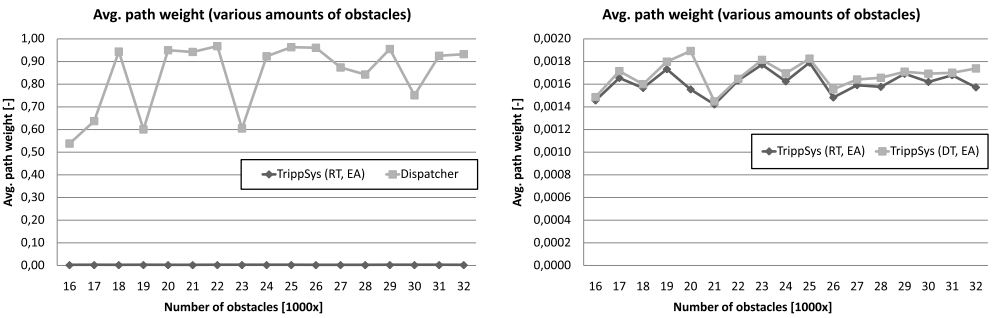


(a) (b)

Fig. 9. (a) Preprocessing times for various numbers of obstacles in the scene. (b) Single obstacle registration times for various numbers of obstacles already in the scene.

As for the quality of the found path, Figs 10(a) and 10(b) show the average path weight for the increasing number of obstacles registered in the system. Here we can see the most significant advantage of the triangulation based path planning algorithms. The conventional approach based on the combination of graph and raster representation navigates along the edges of the adaptive graph that does not provide many possibilities of movement, thus providing a less safe path. On the other hand, the edges of the power diagram are always safe in terms of the distance from the surrounding obstacles. Fig. 10(b) also shows that the Delaunay triangulation (its dual structure, Voronoi diagram, respectively) provides slightly worse results – it only

considers the position of the generating points, whereas the regular triangulation also considers their weight represented by the radius of the bounding spheres. Therefore, the path provided by the dual structure of the regular triangulation has the maximal clearance among the obstacles.

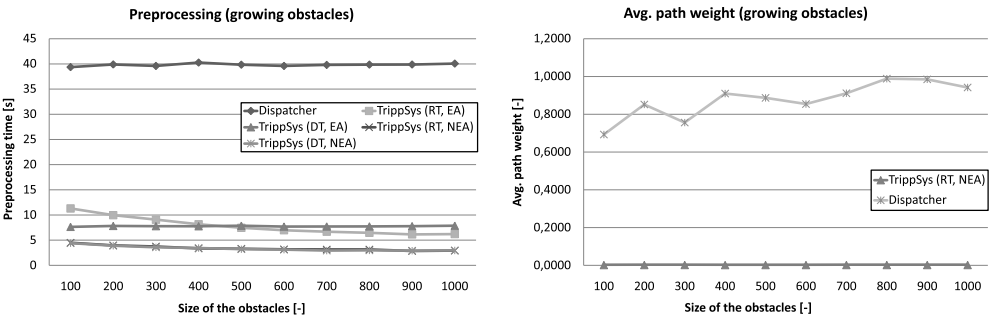


(a) (b)

Fig. 10. (a) Average path weights for various numbers of obstacles in the scene (compared to the raster based algorithm). (b) Average path weights for various numbers of obstacles in the scene (using the triangulation based algorithms).

5.3. Growing obstacles

In the following set of tests, we have observed the behavior of the chosen approaches in an environment consisting of a constant number of obstacles whose radii have been increased in several steps. In the last step, the obstacles almost fully covered the examined space. For the environment of 24000 obstacles with various radii, Fig. 11(a) shows the preprocessing times and Fig. 11(b) then outlines the average weight of the found path in each technique. It can be seen that the change of the size does not affect the characteristics in any significant manner. The triangulation based approaches again demand more time for the registration of a single obstacle but provide significantly better results in terms of the path quality. The average weight of the path found by the conventional algorithm Dispatcher higly varies due to the fact that this approach depends strongly on the coverage of the space with the obstacles. For instance, Fig. 11(b) shows a worsening of the path in the beginning (size 100 to 200) immediately followed by an improvement (size 200 to 300) – as the obstacles increase their radii, bigger part of the examined space is covered which in consequence causes a refinement of the adaptive graph structure in that location, thus providing a finer and safer path.



(a) (b)

Fig. 11. (a) Preprocessing times for growing obstacles. (b) Average path weights for growing obstacles.

5.4. Obstacles in a dilating cluster

In case of a single growing cluster consisting of a constant number of obstacles, the characteristics of the preprocessing and registering time were again very similar to the previous measurements. Fig. 12 however shows an interesting dependence of the path quality while increasing the radius of the cluster of 24000 obstacles. With higher coverage of the space with the obstacles, the conventional solution provides worse results whereas the triangulation based algorithm provides slightly better results. The explanation again lies in the behavior of the space division structures used in both approaches. If only a small part of the space is covered with a cluster of obstacles, the raster based method chooses a path around this cluster. On the other hand, the triangulation based technique has to follow the edges of the power diagram leading through the cluster. By increasing the radius of the cluster, the safe surroundings for the conventional algorithm disappear whereas the edges of the power diagram leading among the obstacles gain higher freedom.

5.5. Protein molecules

In the last step of our measurement, we have observed the path planning systems in an environment consisting of real protein data. The proposed approach proved to be an efficient solution in terms of the preprocessing time and the resulting path. Figs 13(a) and 13(b) present the particular results for various time slices of real protein data and various numbers of molecules.

6. Conclusion

The presented path planning approach is the first one using the regular triangulation for the space subdivision. In comparison to the raster based techniques and the methods using the Delaunay

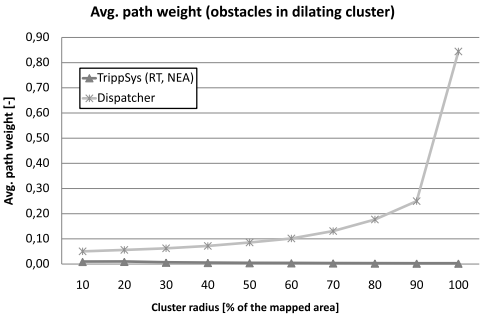
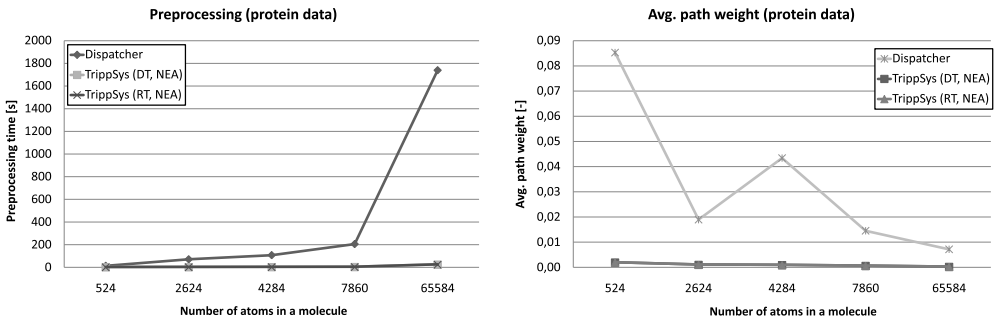


Fig. 12. Average path weights for a dilating cluster of obstacles



(a) (b)

Fig. 13. (a) Preprocessing times for the protein data. (b) Average path weights for the protein data.

triangulation (Voronoi diagram), it requires less time to preprocess and provides significantly safer path (in terms of the distance from the surrounding obstacles) at the cost of suboptimal requirements for the additional registration of a single obstacle. Unlike many path planning techniques, the proposed solution is able to adapt to any change in the scene – insertion/removal of any obstacle, position change, shape change etc. – by only recalculating the correspondent bounding sphere and reinserting the correspondent generator in the triangulation.

Moreover, the proposed gaps filling algorithm (Section 4.3) is able to substantially speed up the main process of replanning the path while providing reasonable suboptimal results. In an environment of 2^{16} randomly positioned and moving obstacles, the gaps filling approach provided a slightly suboptimal solution (1.1-optimal on the average) while taking only 20% of the time needed by the Dijkstra's algorithm.

For the continuing research, we have pointed out the most important and most interesting ways of the improvement:

- as stated in Section 4, the current solution finds a path among the bounding spheres of the obstacles. In the future research, it is our intention to consider this path as the first estimation and locally refine it according to the particular shapes of the surrounding obstacles;
- the proposed technique does not consider any shape of the navigated entity so far. One of the future aims would therefore be the retrieval of a certain tunnel instead of a path for navigating an avatar with concrete proportions.

Acknowledgments

This work was supported by the Ministry of Education, Youth and Sports of Czech Republic - project Kontakt No. LH11006, by the UWB grant SGS-2013-029 – Advanced Computer and Information Systems, and by the Ministry of Education, Youth and Sport of Czech Republic – University spec. research – 1311.

References

1981

[1] Watson D. F.: Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes, The Computer Journal, 2:167-172.

1986

[2] Fortune S.: A sweepline algorithm for Voronoi diagrams, SCG'86: Proceedings of the second annual symposium on Computational geometry, p. 313-322.

[3] Aurenhammer F.: Power Diagrams: Properties, Algorithms and Applications, SIAM Journal on Computing, 16:78-96.

1992

[4] Edelsbrunner H., Shah N. R.: Incremental Topological Flipping Works for Regular Triangulations, ACM Annual Symposium on Computational Geometry, 8:43-52.

1994

[5] Eppstein D.: Finding the k-Shortest Paths, 35th Annual Symposium on Foundations of Computer Science, p. 154-165.

[6] Stentz A.: Optimal and Efficient Path Planning for Partially-Known Environments, Proceedings IEEE International Conference on Robotics and Automation, San Diego, California, USA, p. 3310-3317.

1996

[7] Barber C. B., Dobkin D. P., Huhdanpaa H.: The quickhull algorithm for convex hulls, ACM Trans. Math. Softw., 22:469-483.

1997

[8] Goodman J. E., O'Rourke J.: Handbook of Discrete and Computational Geometry, CRC Press LLC.

[9] Skiena S. S.: The Algorithm Design Manual, Springer-Verlag, New York.

[10] Cignoni P., Montani C., Scopigno R. : DeWall: A fast divide and conquer Delaunay triangulation algorithm in E^d, Computer-Aided Design, 30:333-341.

1998

[11] Bandi S., Thalmann D.: Space Discretization for Efficient Human Navigation, Computer Graphics Forum, 17(3):195-206.

2000

[12] Dor D., Halperin S., Zwick U.: All-Pairs Almost Shortest Paths, SIAM Journal on Computing, 29(5):1740-1759.

[13] Shewchuk J. R.: Sweep algorithms for constructing higher-dimensional constrained Delaunay triangulations, SCG '00: Proceedings of the sixteenth annual symposium on Computational geometry, p. 350-359.

2001

[14] Arikan O., Chenney S., Forsyth D. A.: Efficient Multi-Agent Path Planning, Computer Animation and Simulation, p. 151-162.

[15] Champandard A. J.: Path-Planning from Start to Finish (Internet source: `http://www.ai-depot.com/BotNavigation`).

2002

[16] Geraerts R., Overmars M. H.: A Comparative Study of Probabilistic Roadmap Planners, Proc. Workshop on the Algorithmic Foundations of Robotics (WAFR'02), p. 43-57.

[17] Vigo M., Pla N., Cotrina J.: Regular Triangulations of Dynamics Sets of Points. Computer Aided Geometric Design 19:127-149.

2003

[18] Devillers O., Teillaud M.: Perturbations and Vertex Removal in a 3D Delaunay Triangulation. Proceedings 14th ACM-SIAM Symposium on Discrete Algorithms, Baltimore, MD, USA, p. 313-319.

2004

[19] Shewchuk J.R.: Stabbing Delaunay tetrahedralizations, Discrete Comput. Geom., 32:343.

2005

[20] Wallgrün J. O.: Autonomous Construction of Hierarchical Voronoi-Based Route Graph Representations, Spatial Cognition IV. Reasoning, Action, and Interaction, 3343:413-433.

[21] Beyer T., Schaller G., Deutsch A., Meyer-Hermann M.: Parallel dynamic and kinetic regular triangulation in three dimensions, Computer Physics Communications, 172:86-108.

[22] Ledoux H., Gold C.M., Baciu G.: Flipping to Robustly Delete a Vertex in a Delaunay Tetrahedralization. ICCSA (1):737-747.

2007

[23] Brož P., Kolingerová I., Zitka P., Apu R. A., Gavrilova M. : Path planning in dynamic environment using an adaptive mesh, Proceedings of the SCCG, p. 172-178.

[24] Brož P. : Exact and heuristic path planning methods for a virtual environment, Proceedings of the CESCG.

[25] Medek P., Beneš P., Sochor J. : Computation of tunnels in protein molecules using Delaunay triangulation. Proceedings of the WSCG.

2009

[26] Zemek M., Kolingerová I. : Hybrid algorithm for deletion of a point in regular and Delaunay triangulation. Spring Conference on Computer Graphics, Budmerice, Slovakia, p. 149-2156.

[27] Zemek M. : Regular triangulation in 3D and its applications. Technical report, University of West Bohemia, Faculty of Applied Sciences, Department of Computer Science and Engineering, (Internet source: `https://www.kiv.zcu.cz/site/documents/verejne/vyzkum/publikace/technicke-zpravy/2009/tr-2009-03.pdf`).

Instruction for Authors

Submission of a paper for publication in MG&V is taken to imply that the paper was not previously published (with possible exception of selected conference papers), and neither has been or is being submitted for publication elsewhere. In addition, it is also taken to imply that permission for publication, if needed, has been granted by the appropriate sources. If the manuscript contains any "copyrighted material", the task of obtaining the necessary permissions from the copyright owner(s) is the author(s)'s responsibility.

No short communications are accepted, as MG&V publishes full papers only.

The author actually submitting the manuscript must enclose a note (cover letter) saying that he is submitting a paper with the listed co-authors. Such note is taken to imply that he has obtained their consent for submitting the paper. One of the authors (e.g., the submitting one) should be indicated as the contact person. Otherwise, all correspondence will be sent to the author listed as the first one.

The language of the journal is English.

The papers should be submitted to be considered for publishing in electronic form in one of commonly used formats (.pdf, .doc, .docx, .odt), directly to the Editorial Office e-mail address: MGV@sggw.pl.

Preparation of manuscripts

The layout of the text should contain sufficient spacing and margins. Figures and photos should be included in the text. Colour figures and photos are admissible, provided that colour is essential for the presentation,

The number of pages is limited to 25. The Editorial Office may relax the limit in justified cases.

The paper's title, the authors' names, affiliations and addresses, as well as possible thanks and information about funding, should be placed on a separate page, and the text proper should not contain any such elements in page headers or footers. All the necessary additional elements will be added to the paper's final version following acceptance.

Refereeing process

All papers will be evaluated by one of the Editors, and refereed by three specialists in the given area. The refereeing process ensures anonymity of both referees and authors. Suitability for publication will be judged on the basis of the paper's relevance to the overall aims and scope of MG&V, as well as its quality: originality, technical quality, importance and readability.

Acceptance for publication and preparation of the final version

The authors are informed about the evaluation results and the Editors' decision by e-mail.
The acceptance is conditional on fulfilling further requirements concerning preparation of the final version.

The authors informed about the conditional acceptance are asked to submit the final version of the paper prepared with LaTeX according to the instructions available in the MG&V website: `http://mgv.wzim.sggw.pl`. The bibliography style should also be compliant with the MG&V standard described there. If the text requires nonstandard LaTeX styles or TeX macro definitions, they should be provided too. Nonstandard fonts should not be used without prior arrangement with the Editor. The figures should be sent in separate files, in the .EPS format. It is important that black and white, grey-level and colour figures are prepared in their respective formats, so that no ambiguity arises whether a given figure should be printed in colour or not. In addition, the authors should include a hardcopy-equivalent version of the paper in the .ps or .pdf format for reference purposes.

Together with the final version, the authors should also send their short scientific CVs and photographs (300x400 pixels at a minimum) – or declare explicitly that they do not wish to publish the above.

All the files are to be sent by e-mail to `MGV@sggw.pl`. In addition, a filled-in and signed copyright transfer form supplied by MG&V (together with confirmation that the paper has not been published elsewhere) should be sent back by ordinary mail.

Each paper undergoes a final correction of English in a limited extent. However, it is the authors' responsibility to make sure that the manuscript is written in clear, correct English. In case of a very low standard of English, the paper will be sent back to the authors for correction, which will considerably delay its publication.

The Editorial Office reserves the right to introduce certain changes aimed at preserving a uniform style of the journal's contents. The authors do not receive any galley proof for acceptance.

Published MG&V issue

After the issue containing the paper is published, each author obtains the author's copy. For this purpose, all authors are requested to provide the Editorial Office separately with the mail address for sending the copy, otherwise all authors' copies will be sent to the first author. The Editorial Board does not send any preprints or electronic versions.

The list of contents of the issue is placed in the MG&V website. Selected papers can also be placed in this website, according to the Editor's policy.

Machine GRAPHICS & VISION

CONTENTS Vol. 23, No. 3/4, 2014

Printed in Poland **ISSN 1230-0535**

Abstracted/indexed in:
ACM Computing Reviews,
ACM Guide to Computing Literature,
BazTech,
COMPENDEX (Engineering Information),
EBSCO Computers & Applied Sciences Complete,
INSPEC Information Services,
PASCAL,
SCOPUS (Elsevier),
VINITI Abstracts Journal (Referativnyi Zhurnal),

International Directory of Design (Penrose-Press).