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.

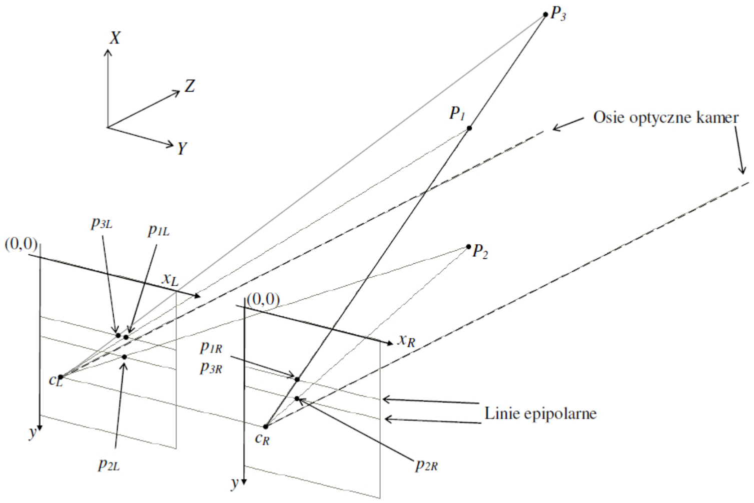

Fig. 1. Canonical stereo-vision arrangement [52].

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):

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

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)

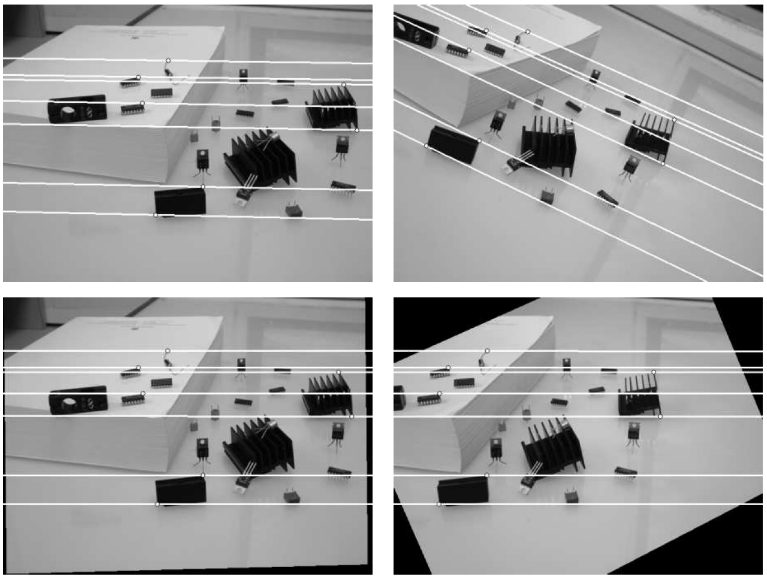

Fig. 2. The upper line – the images before rectification, the bottom line – the images after rectification [50].

- 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):

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

- 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):

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

where:

$C(d_p)$	–	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].

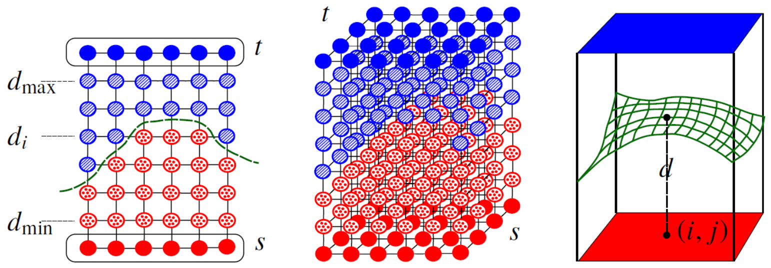

Fig. 3. A cut of one of the 3D graph's surfaces (left), the graphical graph's representation (middle) , the surface obtained in the result of cutting the graph (right) [50].

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

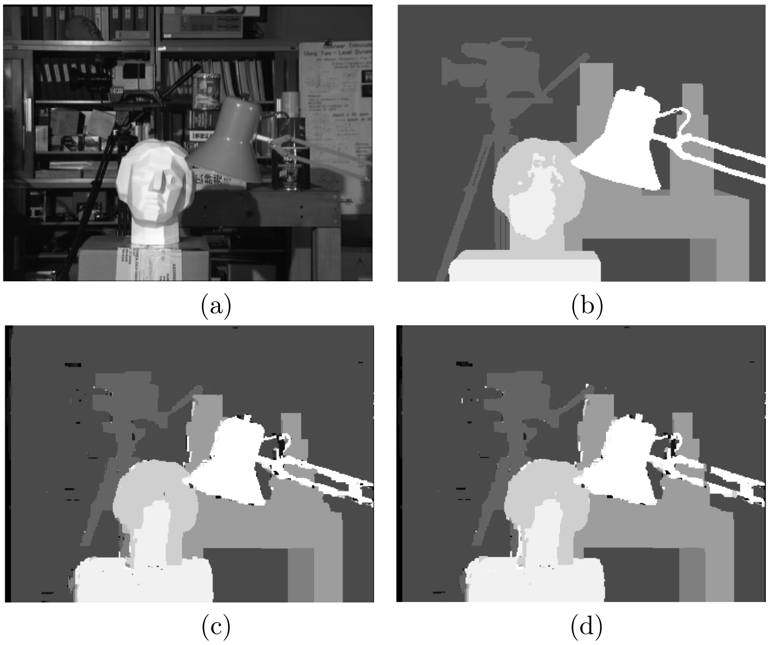

Fig. 4. One of the two stereo-vision images (a), ground truth disparity (b), disparity map obtained by $\alpha\beta$-swap (c), disparity map obtained by α-expansion (d) [32].

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

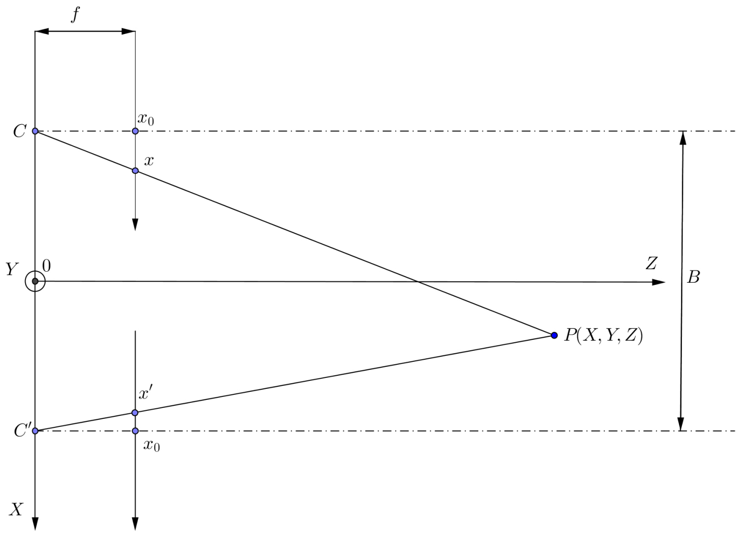

Fig. 5. Geometrical canonical model of the stereo-vision set [52].

∘ 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 $K, K' \in \mathbb{R}^{3\times 3}$ 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:

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

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:

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

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

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:

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

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 $E = Udiag\left\{1, 1, 0\right\} V^T$

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

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

where:

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

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:

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

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.

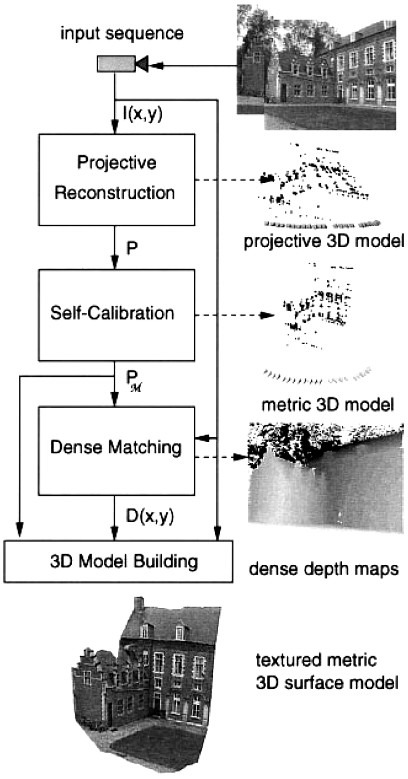

Fig. 6. A pipeline of 3D reconstruction algorithms, using a two-image SfM method, camera autocalibration and dense matching of images [29].

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

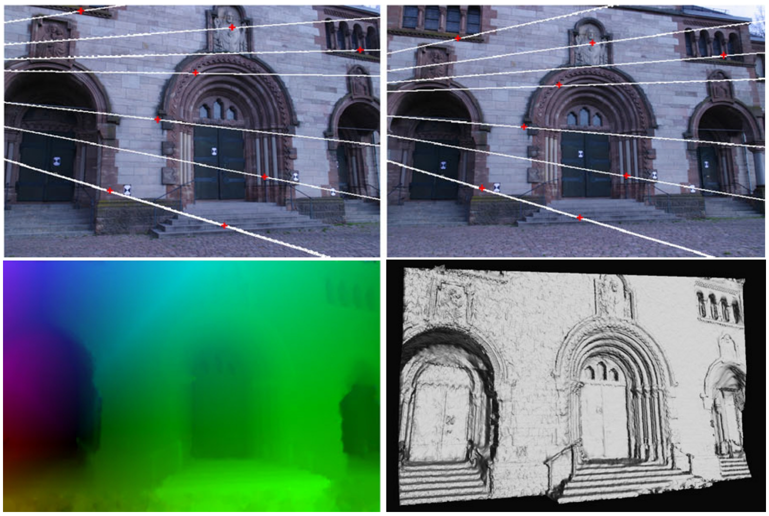

Fig. 7. An example of using two-image algorithm of dense reconstruction on the basis of an optical flow. At the top – a pair of input images with shown epipolar lines. At the bottom – a vector field of optical flow (left) and a dense reconstruction of the scene's shape (right) [84].

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:

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

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:

$$\begin{aligned} R' &= U'\Sigma' \\ S' &= V'^T \end{aligned}$$

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

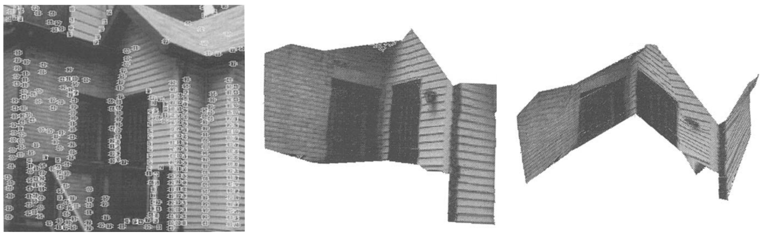

Fig. 8. The final results of the factorization algorithm's, (b) and (c) images show artificially generated 3D models, obtained on the basis of a sequence (a) of input images [8].

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
$M(z, C_j) \in \mathbb{R}^{3\times4}$	–	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".

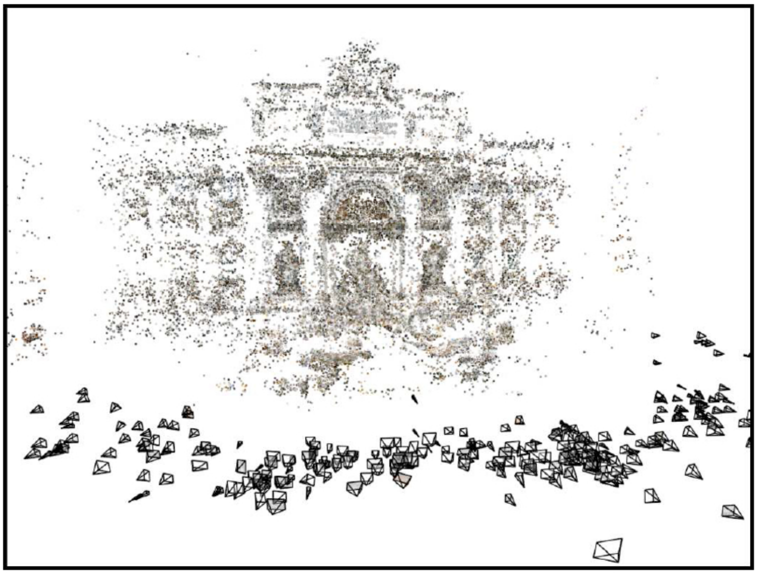

Fig. 9. Cloud of points “Fontanna di Trevi” obtained using the bundle adjustment algorithm [57].

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

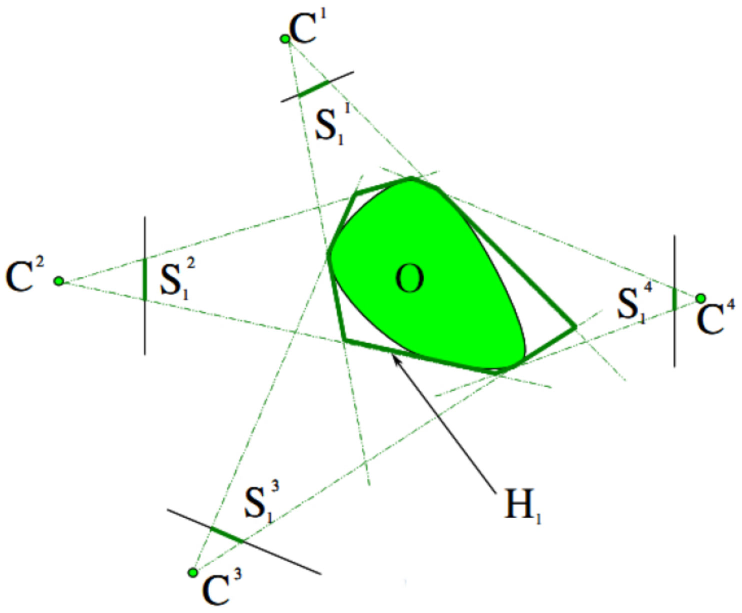

Fig. 10. The visual hull's shape is a result of intersecting the visual cones generated by the silhouettes and central points of cameras[42].

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

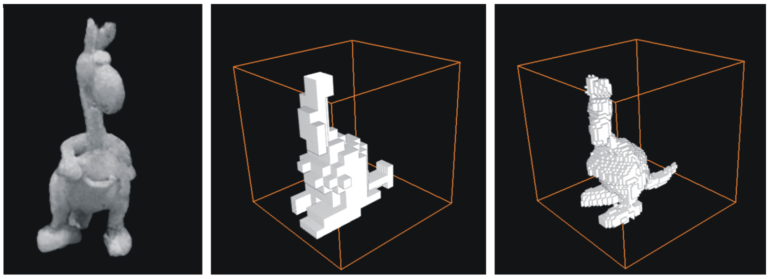

Fig. 11. A reconstruction of 3D object(a) obtained using a volume-based SfS (b)(c). The impact of the voxels’ size on the quality of reconstruction is visible [33].

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:
   $\vartheta$ – 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.

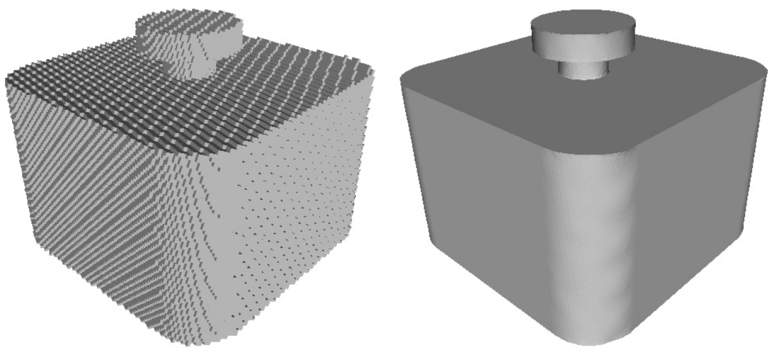
Fig. 12. The isosurface (on the right) generated on the basis of the model built from voxels (the object on the left) [66].

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.

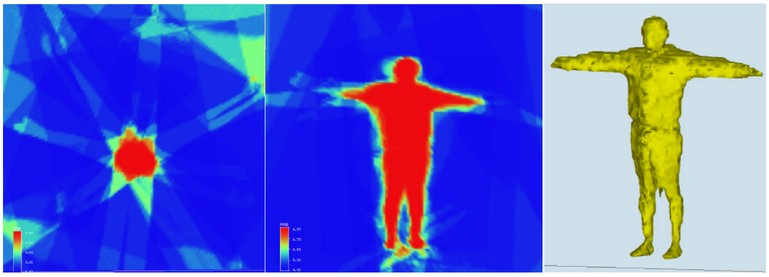

Fig. 13. The VH constrction on the basis of probability of voxel's occupancy, (a) – a view of one layer of the probability grid, (b) – a side view, (c) the visual hull obtained by binarizing the probability grid [49].

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.

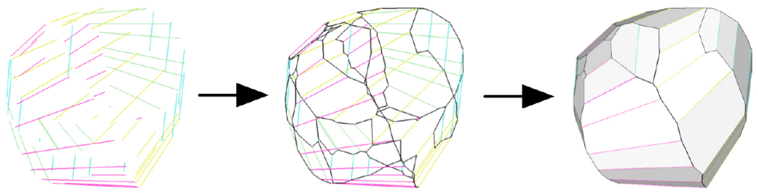

Fig. 14. The key steps of EPVH algorithm: (a) compute the viewing edges, (b) cones intersections and triple points, (c) faces [71].

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.

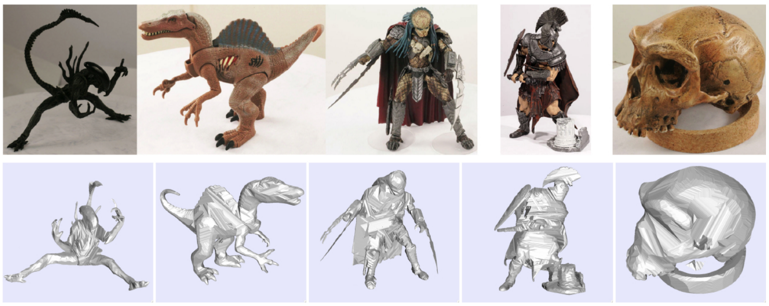
Fig. 15. The results of the EPVH algorithm [71].

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.

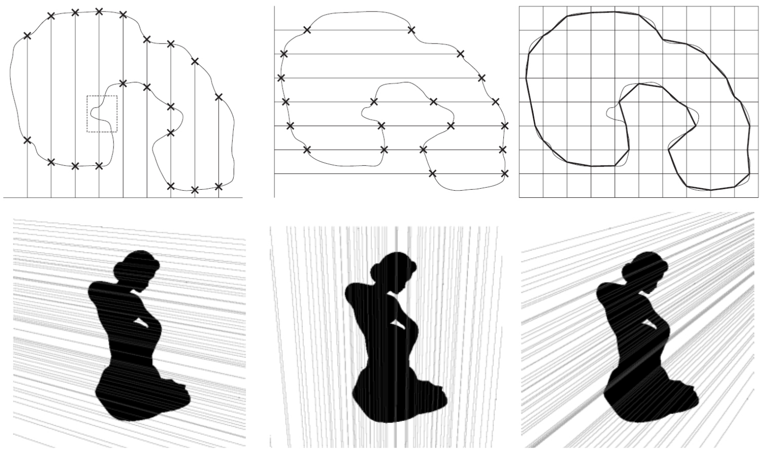
Fig. 16. The marching intersections: upper row – rule of determination the points in which the silhouette intersects with the grid lines (an example of a 2D grid), the bottom row – projection of the 3D grid's line on the silhouettes images[40].

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

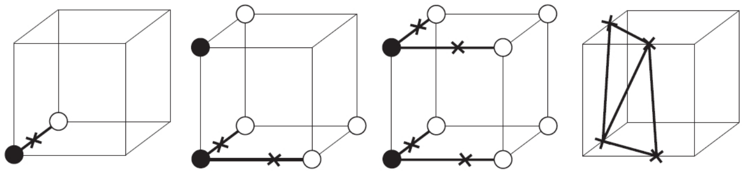

Fig. 17. For black nodes (which are placed inside the VH) with direct neighbour of white nodes (outside the VH) undergo a procedure of surface construction [40].

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

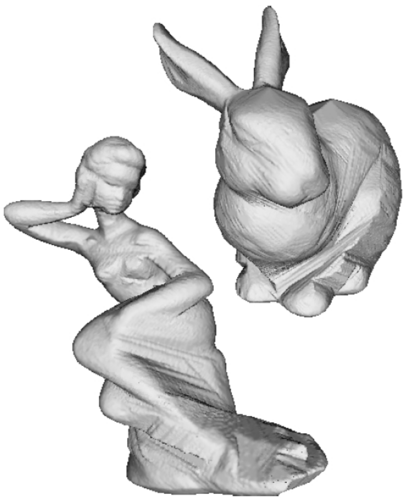

Fig. 18. Example of results obtained from marching intersections algorithm [40].

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 $\mathrm{consist}_K() : Col^K \times \left\{\mathbb{R}^3\right\}^K \rightarrow \{0,1\}$ 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 $p \in \mathrm{Surf}(V)$. Set $\mathrm{Vis}_V(p)$ 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

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

problem of effective calculating set $\text{Vis}_V(p)$ 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).

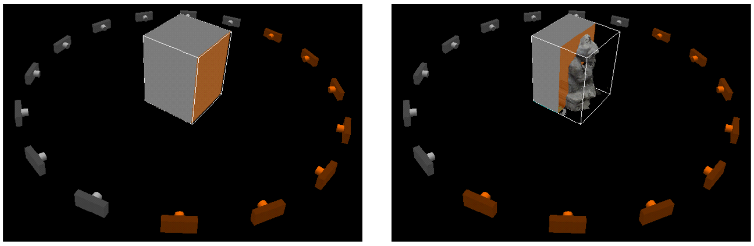
Fig. 19. Plane sweep algorithm. When voxels slice moves from the initial positon (the image on the left) inside the scene (the image on the right) the number of cameras (which take part in determining photo-consistency with scenes shape) increases [40].

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

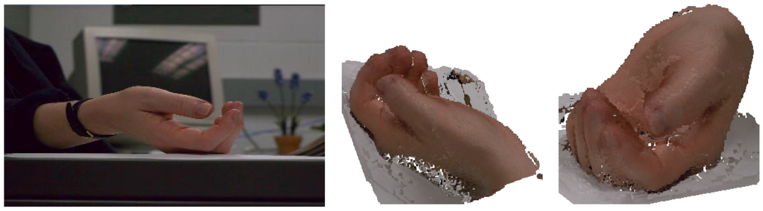
Fig. 20. The results of SfPC multi-sweep algorithm (middle, right) for a sequence of a hundred input images (left) [30].

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 $\Pi_i$ - 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

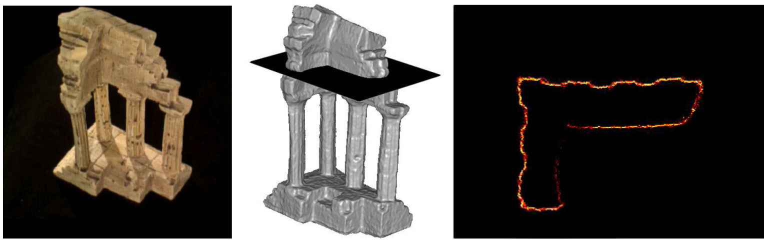

Fig. 21. The results of the algorithm [65], (left) -– one of sequence of 312 input images, (middle) – a 3D model generated on the basis of the input images, (right) — the accumulated value $vote_i$ for voxels in the slice denotated as plane in middle image.

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:

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

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:

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

Generalizing for many images we obtain:

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

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.

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

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:

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

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

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

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

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

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

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

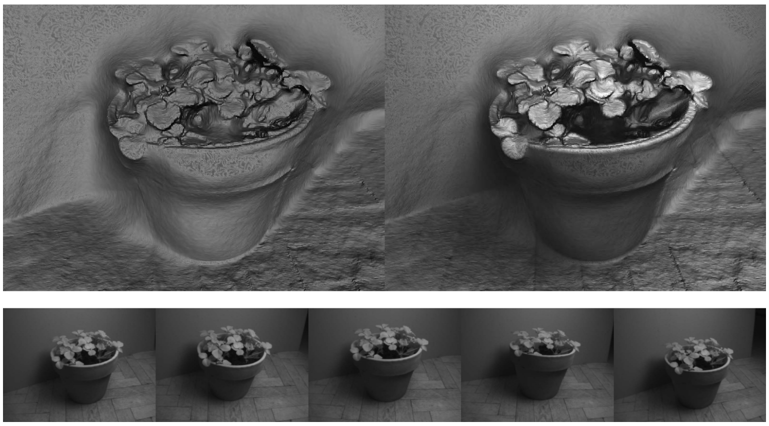
Fig. 22. The results of SfPC algorithm [77]. The top line shows the obtained 3D model and the impact of an increasing number of images on the quality of mapping the real scene. The bottom line shows selected frames from a sequence of input images.

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:

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

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

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

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:

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

and

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

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: $\nu_1$, $\nu_2$, $F$, $D$, $\gamma$. Determine the value of regularization parameters $\alpha$ and $\kappa$. Determine threshold value $\varepsilon$ 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 $\frac{\partial s}{\partial \tau} \triangleq -M(s)(E'_1(s)+E'_2(s)) - E'_3(s)$, 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

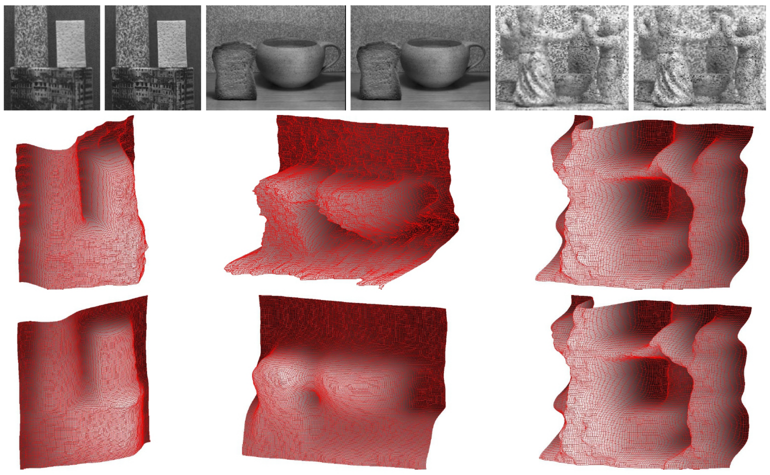

Fig. 23. The results of the Shape from Defocus via relative diffusion method [67] for three pairs of input images (top row). The scene's model is obtained without pre-conditioning conditioning (middle row) and with pre-conditioning (bottom row)

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.