Control and Cybernetics

vol. **43** (2014) No. 1

Non-photorealistic rendering with the use of short segments of straight lines in a vector space of increments[*][†]

by

Mariusz Borawski

Faculty of Computer Science and Information Systems
West Pomeranian University of Technology
ul. Żołnierska 49, 70-210 Szczecin, Poland
mborawski@wi.zut.edu.pl

Abstract: In the article, a two-stage method of non-photorealistic rendering is presented. In the first stage, short segments of straight lines described by vectors and tangential to the object contours are generated. On this basis, a vector image is created with the use of vector space of increments. This image can be a vector representation of the object contours in the picture.

Keywords: non-photorealistic rendering, vector space of increments

1. Introduction

Non-photorealistic rendering involves a wide range of methods focusing mainly on providing artists with ready tools that may broaden their artistic potential. Nevertheless, there are also methods allowing for automatic non-photorealistic rendering directly from photographs. This process may be divided into the main stages of image analysis and image generation. In the case of more complex algorithms, image generation is based on the analysis of brush marks (Lee et al., 2007). As for simpler algorithms, this may consist of a change in colour and texture of image fragments. Image analysis is the most difficult part of non-photorealistic rendering. Normally, it consists of many stages and its effect depends not only on the selection of methods, but also on the parameters. An example of a complex algorithm of image analysis based on segmentation is presented in the article by Kasao and Miyata (2006).

Methods of non-photorealistic rendering are aimed at a gradual reduction in the number of details in the image and, at the same time, at exposing certain image elements. This requires detecting and classifying certain local image

[*]Submitted: May 2013; Accepted: February 2014

[†]This is an extended and modified version of the paper presented at the Congress of Young IT Scientists, SMI 2012.

properties. For instance, a gradient may be used for determining the direction in which the tool will be used (Haeberli, 1990) or for strengthening the strokes on the edges (Litwinowicz, 1997). The analysis of gradient changes allows for the detection of contours, which are major elements of many non-photorealistic rendering methods (DeCarlo and Santella, 2002; Kasao and Miyata, 2006). It is not plausible to develop a single universal method of non-photorealistic rendering. Depending on the painting technique, the elements to be exposed and techniques for image analysis and generation change. Different methods are adopted for images created with the use of pen and ink (Winkelbach and Salesin, 1994), watercolours (Curtis et al.,1997), coloured pencils (Takagi, Fujishiro and Nakajima, 1999), oil pictures (Litwinowicz, 1997), etc. Very often, various methods for image generation can be used as parts of a given painting technique. Each method requires a different approach to non-photorealistic rendering.

Methods of non-photorealistic rendering, which simulate pictures made with ink or pencil, have to produce contour images. This is the reason why they include contour detection. One of the most popular edge detectors is the Canny algorithm (Canny, 1986). This filter has four stages: noise reduction, horizontal, vertical and diagonal edge detection, non-maximum suppression, and thresholding to remove unimportant edges. The Canny edge detector is used in many more complex algorithms (Fisher, Bartz and Strasser, 2005). It was the basis for the Deriche edge detector (Deriche, 1987). The results obtained for these two algorithms are very similar.

Another algorithm for edge detection, very often used in non-photorealistic rendering, is the Difference of Gaussians (DoG). This algorithm involves the subtraction of one blurred version of an original image from another, less blurred version of the original. The difference between the levels of blur influences the result obtained. There are many different types of DoG algorithm used in non-photorealistic rendering, an example of which is the Flow-based Difference-of-Gaussian (FDoG) algorithm (Kang, Lee and Chui, 2007). This method uses a kernel-based nonlinear smoothing of the vector field and aims to preserve the most important directions of the edges. Less visible edges are eliminated, while stronger adjacent edges are reinforced. On the basis of FDoG, an edge detector examining the edges in the perpendicular direction to the tangent which determined the course of the contour was proposed in Wang et al. (2012af). In Wang et al. (2012b) several amendments to FDoG were proposed. These amendments considered information about colour for detecting edges and removing distortions introduced by FDoG at the contour branching.

The majority of non-photorealistic rendering methods are designed for creating graphics on a computer screen or for printing and do not create vector illustrations but only raster images. Methods that give very good visual effects are not always suited to every application. For example, the method described in Lu, Xu and Jia (2012) produces very attractive and effective images. However, it cannot be used for drawings created with the use of milling machines because, apart from the object contours, the images contain areas of varying degrees of brightness that are difficult to obtain on a milling machine. The major

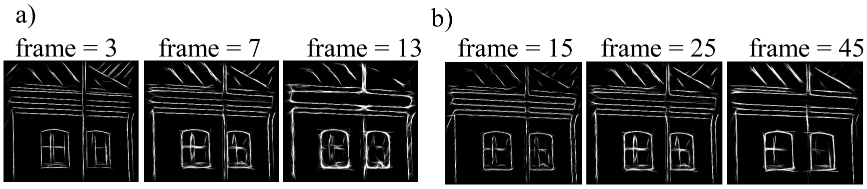

Figure 1. Influence of frame size on the result obtained: a) frames of neighbourhood standard deviation (frames of the block to determine the lines equal to 25), b) frames of the block to determine the lines (frames of neighbourhood standard deviation equal to 7)

problem in the case of using non-photorealistic rendering methods in other applications, apart from only printing or displaying, is generating the image only in raster form. Steering a CNC milling machine or a robot requires a vector image, because it can be easily converted into instructions for controlling these devices, for example in the format of G-code.

There are specialised plotters to perform the artworks. The examples are provided by Arman (2013) or Kelly and Marx (2013). The latter has the ability to self-mix the paints. Devices of this type require the appropriate algorithms of non-photorealistic rendering, which allow for creation of the information necessary for their control. In Lindemeier, Pirk and Deussen (2013), two methods for steering the painting robot are presented. The first one uses predefined sets of paths to the style, which are randomly chosen at random orientations (the device uses only those with the best fit). The second method generates an image dynamically on the basis of a photograph and information obtained from a camera, using the FDoG algorithm. The drawings obtained are composed of many lines running close to each other, making it impossible to apply this method to the control of the CNC milling machine. All lines would in this case integrate into one common cavity.

The paper presents an approach to the detection of contours and their vectorization using short line segments for controlling a CNC milling machine. This allows for the creation of souvenirs in wood or metal for the tourist market.

2. Line detection

Contourisation is the first stage in line detection; it is carried out with the use of neighbourhood standard deviation (Pratt, 2001). The standard deviation was calculated in a frame of approximate size 7x7. The size affects the rounding of the object contours: the larger the frame, the smoother the contours of the objects (Fig. 1a).

Once contourisation has been completed, segments of straight lines are determined and the contour of the object is approximated. The lines are determined on a pad of approximate size 25x25. The size of the block affects the length of

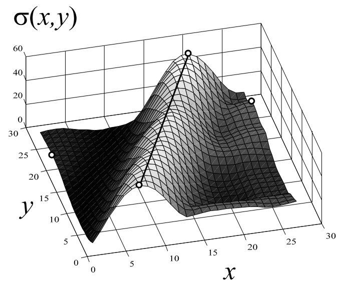

Figure 2. Pad approximately 25x25 and the line that crosses it

the lines detected. The larger the block, the longer the line has to be in order to be detected by the algorithm. This influences the level of detail which will be later reconstructed. The longer the lines are, the less detailed the picture will be. The pads should overlap. The best effect is produced when the start of every pad has been translated relative to the previous pad by one pixel (just as in the case of convolution or median filtering). On each side of every pad, the elements with the highest values are determined. In Fig. 2 these are marked with white dots. In this manner, one element is obtained for every side of the pad. These are elements joined by the line that crosses the pad. For four elements there are six combinations of plausible courses of the line. Combinations, for which the line is too short, are rejected. The minimum length of the line is 10. This value does not affect significantly the result obtained. However, it cannot be larger than the size of the frame, because none of the lines will be detected. Differences between values 1 and 25 for a frame are practically unnoticeable. This parameter is to reduce the amount of calculations needed. The larger it is, the fewer lines for further analysis remain.

A regression line in a three-dimensional space is determined for each combination left after the rejection. Another step consists of determining residual variance (the averaged square deviation from a theoretical line). Residual variance is a parameter that allows for assessing the correspondence between the line and the approximate data. Therefore, this enables the selection of a straight line, the course of which is most similar to the edge crossing the pad. Unfortunately, its value depends, to a great extent, on the level of the approximate values. In order to become independent of the above, residual variance should be divided by the average value of the pad elements that are found below the straight line. From all the lines, one selects the line with the lowest residual variance. Nevertheless, it may happen that the course of the determined line goes above the extremely heterogeneous space. Such a line should be rejected. Standardised residual variance that equals two has been adopted as the rejection threshold. If it exceeds this value, the line will be rejected, just as lines whose average is not at least 10 times higher than the average value of all the

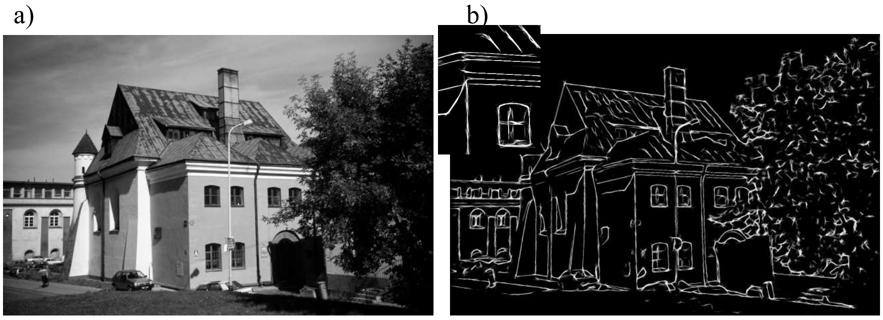

Figure 3. Segments of the line determined for an image of approximately 2048x1333: a) original image, b) image with line segments

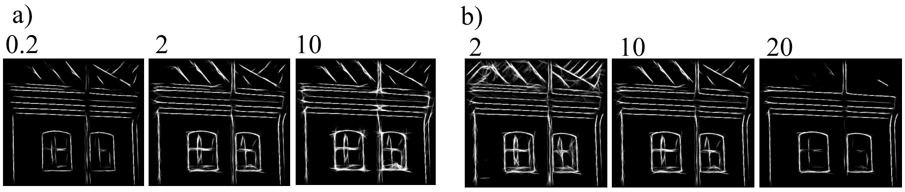

Figure 4. Influence of the rejection threshold on the result on the basis of: a) residual variance, b) average height of the line

elements on the pad. Fig. 3b shows the result of calculations for a photograph of approximate size 2048x1333. The brightness of the pixels depends on the number of overlapping segments of the line.

The rejection threshold on the basis of normalised residual variance or on the basis of average height of the line affects the number of considered lines and defines to which extent the line could be heterogeneous. The larger it is, the more 'creased' the contour under the line could be. The smaller it is, the more closely the line to the contour should adhere (Fig. 4a). The rejection threshold on the basis of average line height determines what the minimum height from the line level is relative to the average values of the pixels in the block. In fact, the contour is clearer if this difference is greater. With a high threshold value, only those lines that extend over very clearly visible contours will remain (Fig. 4b).

Fig. 5 shows the effects obtained for photos sized 1024x667 and 2048x1333. For the smaller photo, the frame size is reduced twice, for neighborhood standard deviation from 7 to 3, for the block in which the lines were determined from 25 to 13. It can be observed that results differ little from each other. Therefore, the size of frames should be adjusted to the size of the image, decreased or increased

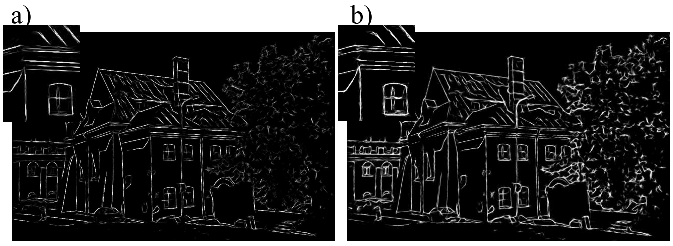

Figure 5. : Influence of the size of the image on results for photos sized: a) 1024x667, b) 2048x1333

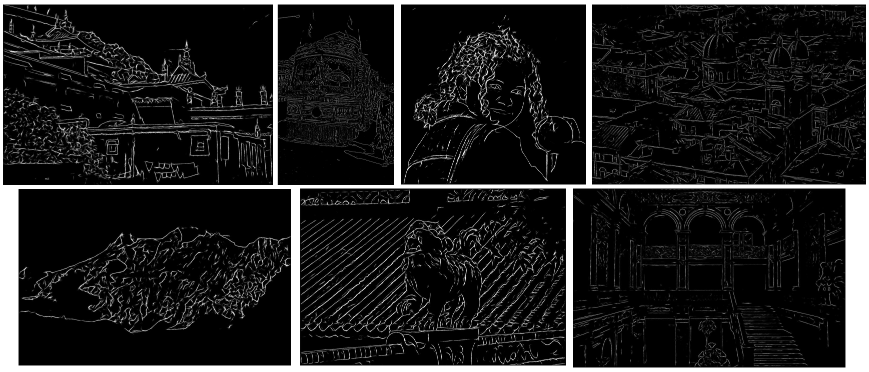

Figure 6. Images drawn with the use of lines

as much as the size of image decreases or increases. The change in the size of the image mainly affects in this case the thickness of the contours obtained. In practice, for the photo sized 2048x1333, the optimum frame size for the local standard deviation is 7x7, and for the block 25x25. If an image is smaller or bigger, the size of the frame should be proportionally decreased or increased. And so, for the photo sized 1024x667, frame size for the local standard deviation is approximately 4x4, and the block approximately 13x13.

The method under discussion is particularly useful for the non-photorealistic rendering of buildings. A satisfactory effect is also produced in the case of treetops and hair. A problem is posed by very dark spots where lines are usually not drawn. Such a situation is observed e.g. in the shade where there is sun glare (see Fig. 6).

3. Determining the contour of the object

Short segments of the line may be determined not only directly for generating the image, but also in much more complex methods. By reducing the number of

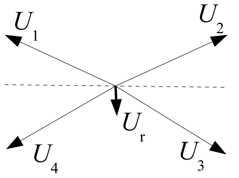

Figure 7. Summing the vectors representing lines

lines, the contour of the object can be described by vectors defining its direction. In order to do so, the determined segments of straight lines may be noted in the form of vectors P and U that represent the straight line in parametric notation (Bronsztejn et al., 2004):

$$R = P + Ut \; , \tag{1}$$

where P is a vector representing any point crossed by the line, U is a direction vector of the line, and t is a parameter that is a real number. After dividing the line into segments, many overlapping segments are obtained. As a result, the contour is determined by a great number of segments which are very often parallel. This number can be reduced.

The reduction is made by setting the vector representing the directions of the line described by vectors in blocks. The differences between effects obtained for different sizes of block are unnoticeable, whereas the calculation time for large blocks increases significantly. This is the reason why the size of block is decreased from 25x25 to 15x15. A major problem is that the vector determines not only the direction but also the orientation, whereas in the case of lines only the direction is considered. Such an operation cannot be performed by summing the vectors. This is demonstrated in Fig. 7. In the figure, four vectors representing the four directions of the lines are drawn. The sum of these vectors is vector U_r. A line which could represent the lines corresponding to vectors U_1, U_2, U_3 and U_4 has been drawn as a dotted one. It may be observed that it is almost perpendicular to the vector U_r. This is the result of the opposite orientation of vectors U_1 and U_3, and U_2 and U_4. To correctly determine the line representing the line defined by these vectors, the orientations of all the vectors must be changed so that the angle between any two vectors does not exceed 180 degrees.

The first stage for determining a vector representing the directions of other vectors is bringing all the vectors to unit vectors. This is done due to the fact that the length of the vector, which is not relevant because it does not represent any factor in determining the line, did not affect the result of the calculation. For instance, if vectors U_1, U_2 and U_3 are given, they will be turned into unit vectors, which can be marked as U_1', U_2' and U_3' (Fig. 8a). Next, as the vector representing the directions of these vectors U_r the first vector U_1' is accepted (Fig. 8b). In the following step it is checked whether the angle between vector U_r

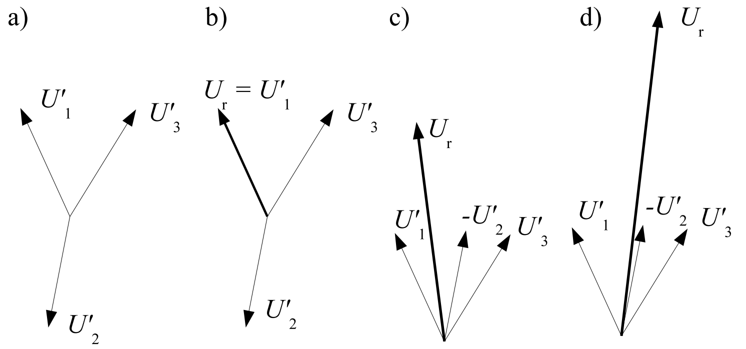

Figure 8. Determining the vector representing directions of vectors U'_1, U'_2 i U'_3: a) vectors U'_1, U'_2 i U'_3, b) first stage of determination of the vector U_r , c) second stage of determination of the vector U_r, d) third stage of determination of the vector U_r

and next vector U'_2 is not greater than 90 degrees. This can be performed by testing the sign of the scalar product of vectors: U_r and U'_2. In the proposed application, the scalar products were used:

$$(U'_k, U_\mathrm{r}) = \sum_{i=1}^{N} u'_{ki} u_{\mathrm{r}i} \ , \tag{2}$$

where $N = 2$ is a space dimension and u'_{ki} and $u_{\mathrm{r}i}$ are coordinates of vectors U'_k and U_r. The presented method of determination of vector representing the directions of other vectors works for any dimension of the space.

When the scalar product is smaller than zero, which is the case for the vectors U'_1 and U'_2, a change in orientation of vector U'_2 is made. This is achieved by multiplying it by -1 and adding to the vector U_r. If the value of the scalar product is greater than zero, vectors U'_2 and U_r without a change in sign are added. The step of checking the sign of the scalar product and summation with vector U_r is then also conducted for all the other vectors. The result is the first vector of the vectors U_r.

The algorithm can be extended with a measure of fit of the vector U_r to the set of vectors U'_k. To obtain such an effect, components of the vectors U'_k along the vector U_r reduced to a unit vector can be used:

$$c_k = \left(U'_k, \frac{U_\mathrm{r}}{\sqrt{(U_\mathrm{r}, U_\mathrm{r})}} \right) \ . \tag{3}$$

These components are a measurement of the length of the vectors U'_k along the vector U_r. A graphical interpretation of the components is presented in Fig. 9a. Because both U_r and U'_k are unit vectors with the same orientations c_k

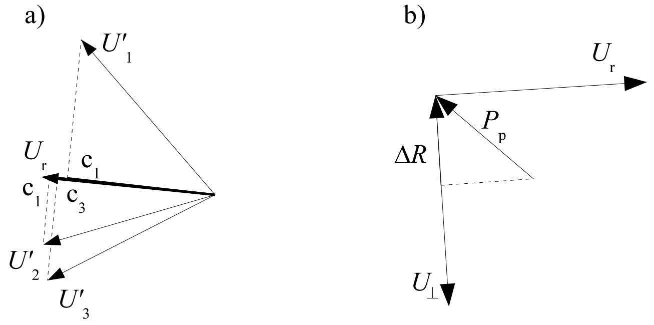

Figure 9. The interpretation in two-dimensional space of: a) calculation of the components of vectors U_1', U_2' and U_3' along the vector U_r, b) determining the vector ΔR

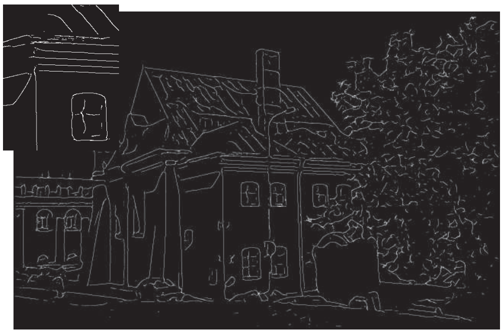

Figure 10. Result of segment reduction

they fit in the range from zero to one. Their mean value will be the measure of fitting the vector U_r to the vectors U_k'. The larger it is, the more similar the directions of the lines represented by the vectors U_k' will be.

After determining the vector U_r, the vector P_r that represents the origin of all the vectors on the pad is calculated. The coordinates of this vector are the average values of the coordinates of vectors P_k that represent the origin of vectors U_k. On the basis of vector P_r, the vector P_p, that is, the difference between the coordinates of the pad centre and vector P_r, are calculated. Subsequently, vector P_p is projected onto vector $U_\perp$ that is perpendicular to vector U_r. In this way, the vector ΔR that defines the translation of the pad is obtained (see Fig. 9b). After the translation, all the steps are repeated for all the vectors on the pad. These are repeated 20 times. Vector U_r, obtained during the last iteration, is stored as a vector representing the direction of contour on the spot determined by its origin. Vectors U_r and P_r represent segments. Fig. 10 shows the midpoints of the segments presented in the image.

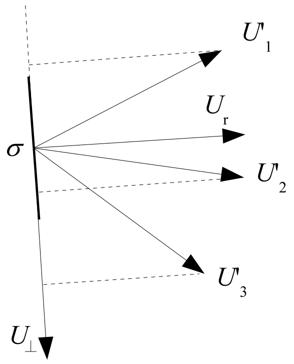

Figure 11. The determination of the standard deviation defining the dispersions of possible directions of the vector ΔU

The direction of the vector U_{r} was adjusted in the best possible way to present the directions of all the vectors that it represents. However, depending on the character of the set, this representation will be more or less precise. If the directions of all the vectors are quite close to each other, then the representation will be better. The "measurement" of representation can be the standard deviation, determining the possible dispersion of directions. Such a standard deviation can be determined by reducing to unitary form all the vectors taking part in calculating the vector U, and then calculating their components along the vector $U_{\perp}$ perpendicular to it (Fig. 11). If the vector $U_{\perp}$ is the unitary vector, then the deviation is the measurement of the dispersion of vector U. In practice, it can be more convenient to use the components of the deviation along the coordinate system axis. When exchanging the values of standard deviation for increments, one can define the vector of which the average increment values are the coordinates of the U vector.

All the parameters presented in this step are chosen experimentally. Frame size depends on the size of the image. And so, for the photo sized 2048x1333, optimal frame size is 15x15. For larger images it must be proportionally increased, for smaller images decreased. The number of repetitions does not depend on the image size, and for each image it can be left at the level of 20.

4. Linking contour points

Contour points can be recorded as tables of x and y coordinates and tables of these coordinate inaccuracies expressed via standard deviation increments (a description of the operations on increments of the standard deviation in the vector space can be found in Borawski, 2012). Coordinates x and y mark outline points and are the locations of the initial points of vectors $\underset{i}{U_{\mathrm{r}}}$, determining the directions of the line segments passing through these points. The initial points can be described with the use of vectors and marked as $\underset{i}{P}$. These vectors can be analysed in the increment space. In such a case, the coordinates of the outline points after thinning are treated as average value increments. Deviations are calculated perpendicularly to the segments that are related to these vectors, and treated as standard deviation increments. Each point $\underset{i}{P}$ corresponds to the vector $\underset{i}{U_{\mathrm{r}}}$. Average value increments of this vector determine the segment direction passing through the initial point, while standard deviation increments and standard deviations of possible vector directions are calculated in the direction perpendicular to it.

Thus, the vectors determined describe the contour of an object, but there is no information about the link between them. An outline description requires placing the vectors in a table and arranging them in a way that would allow for the vector description points to lay close to each other. The points of contour branching constitute a significant problem. Generally, solving this problem requires marking the branching points and dividing the contour into fragments between these points, and placing them in different tables. In the case of contour description with the use of straight segments, there is an increase in the standard deviation increment of the vectors $\underset{i}{P}$ and $\underset{i}{U_{\mathrm{r}}}$, which simplifies point detection and the division of the contour into sub-contours.

Non-sequenced vectors will be further marked $\underset{i}{P}$ and $\underset{i}{U_{\mathrm{r}}}$, while sequenced vectors in a table creating contour, as $\underset{j}{P'}$ and $\underset{j}{U'_{\mathrm{r}}}$. Having a determined set of sequenced vectors, every next vector is attached to this set on the basis of n_w vectors attached before. Here, n_w is a parameter which can be chosen freely. The bigger it is, the more the vectored curve should resemble a straight line. If n_w is bigger than the number of sequenced vectors, then all the sequenced vectors are taken into consideration. In the given example n_w was established as 20. Figure 12 presents two already sequenced contour points (marked with prime symbol). For n_w increments of average values of the lately sequenced vectors U'_j, a vector representing direction of their average values increments is established. This vector is normalised to unitary form. Its sense changes so, that the correlation with the lately determined vector representing the direction of all the vectors would be positive. Similarly, a vector representing the direction n_w of standard deviation increment of the lately sequenced vectors $\underset{j}{U'_{\mathrm{r}}}$ is established. This vector is normalised to the unitary form and then multiplied by the sum of the lengths of the considered vectors $\underset{j}{U'_{\mathrm{r}}}$, calculated as $\left\| \underset{j}{U'_{\mathrm{r}}}, \underset{j}{U'_{\mathrm{r}}} \right\|_0$. Average value increments and standard deviations create the vector U'_{rr}.

As in the case of vector U_{rr}, vector P_{rr} can be established on the basis of n_w last fixed vectors $\underset{j}{P'}$. Value increments of the vector averages are of no importance. Vector U_{rr} is initiated in a place pointed by the newly-determined vector $\underset{j}{P'}$ and multiplied by k which can take values $1, 2, \ldots, n_k$. The parameter n_k can take the value of 4. At the beginning, the value 1 is taken for k, and

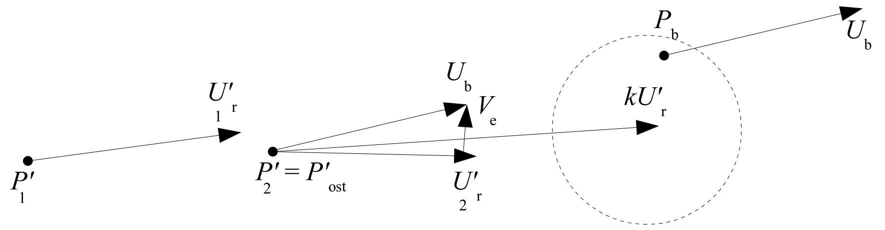

Figure 12. Establishing contour points

the vector $\underset{i}{P}$ positioned the closest to the end of vector U_{rr}, is sought. Further, the vector $\underset{i}{P}$ will be marked as examined vector P_{b} and the vector $\underset{i}{U}$ related to it will be marked as U_{b}. For this vector the condition is checked related to the location of vector P_{b} towards the last of the sequenced vectors P_{ost}:

$$\|P_{\text{b}} - P_{\text{ost}} - kU_{\text{rr}}\|_1 \leq w_{\text{pol}} \|R_{\text{odl}}\|_0 , \tag{4}$$

where w_{pol} is a parameter determining the quantity of multiplication of standard deviation increment vector length. Its value is assigned as six. R_{odl} is a vector representing the direction of standard deviation increment of three vectors: P_{b}, P'_{sr} and kU'_{sr}. Vector P'_{sr} is a vector representing standard deviation increments of the newly-sequenced n_w of vectors $\underset{j}{P'}$. Its length is an average value of projections n_w of vectors $\underset{j}{P'}$ along the direction established by the vector P'_{sr}. Similarly, the vector U'_{sr} is a vector representing increments of standard deviations of the newly-classified n_w vectors $\underset{j}{U'}$. Its length is a weighted average of the projections n_w of vectors $\underset{j}{U'}$ along the direction determined by the vector U'_{sr}.

Additionally, the condition related to the conformity of vector directions is verified. For this purpose, the component c of the vector U_{b}, is marked along vector U'_{r} , with the use of the scalar product and with the weight being one. On the basis of this component the vector deviation $V_e = U_{\text{b}} - cU'_{\text{r}}$ is determined. Then the following condition can be formulated:

$$\|V_e\|_1 \leq w_{\text{kier}} \|R_{\text{kier}}\|_0 , \tag{5}$$

where w_{kier} is, as in the case of parameter w_{pol}, the parameter determining the scope of multiplicity of standard deviation increments of vector length. Its value was assumed as four. R_{kier} is a vector representing the direction of standard deviation increments of two vectors: U_{b}, and U'_{sr}.

Fulfilling all of the above conditions results in adding the vectors P_{b} and U_{b} to the contour marking and repeating all the operations. If any of the conditions is not fulfilled, then the value of k is incremented by 1, and all the actions are repeated until the value of n_k is exceeded by k. After exceeding this value, a mirror-image of the tables with vectors $\underset{j}{P'}$ and $\underset{j}{U'}$ is made. Due to this operation, the first elements become the last. This causes a return to the starting point and enables the completion of contour vectorisation, which began from its inner part. Once more vector U_{rr} is determined, but in the first iteration its orientation changes to the opposite of that of the last vector $\underset{j}{U'}$. Then, all the operations are repeated until the moment of exceeding the value n_k by k. This causes the end of the contour vectorisation. From the vector set $\underset{i}{P}$ one should remove all the vectors which were found in the contour description, as well as all the vectors describing points located at a distance equal or lesser than $w_{\mathrm{pol}} \left\| \underset{j}{P'} \right\|_0$ from the points described by the vectors $\underset{j}{P'}$. In addition, all the vectors U'_j, corresponding to vectors P'_j, are to be removed. This enables the vectorisation of the next sub-contours with the omission of the already vectored sub-contours placed in different tables.

The method presented is a multi-stage method in which data from one stage are used as input to the second stage. As a result, this method is able to process the image from the raster to the vector form. The results from performing the method are presented in Fig. 13. The output of many methods of line and contour detection is a raster image. Such an image has to be vectorised. This may indicate the necessity of performing additional stages such as the thinning of lines. An example of a line detection method that is able to proceed directly from raster to vector form is the method described by Xiao and Tao (2007). This method uses the Hough transform to determine the straight line segments. It is mainly characterised by the fact that it seeks lines (not necessarily continuous) running through the whole picture. This makes it particularly well-suited for detecting such elements as the edge of the road or the centre of a traffic lane. In the case of images, however, it often leads to distortion, as shown in the example of a roof edge in Fig. 14a.

In order to compare the proposed method with other methods which create raster images, their output has to be vectorised. For comparison, the Canny and Deriche algorithms, and the LSD method detecting lines were selected (Grompone von Gioi et al., 2012). The raster image created by these methods was thresholded and subject to thinning with the use of morphological operations. For that purpose the function bwmorph from the Octave-Forge package with the parameter ’thin’ was used. Next, the image was vectorised with the nearest neighbour method in which the maximum distance between pixels was set to 5, and the minimum number of pixels belonging to a single contour was set to 5 as well. The result is shown in Figs. 14b, 14c and 14d.

The LSD method coped very well with the detection of the lines. A major problem, however, was caused by the components that do not contain lines. An example is the treetop, where there are very few lines detected. Therefore, this method is not suitable for use with human portraits, where the problem is the

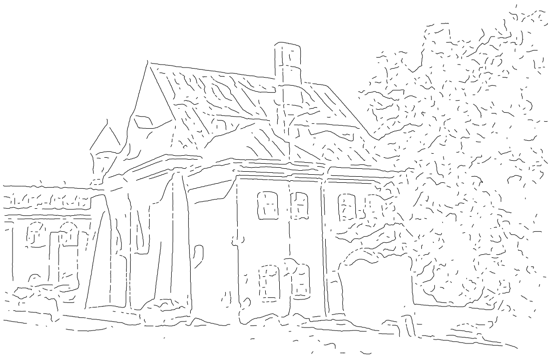

Figure 13. Contour detection with the proposed method

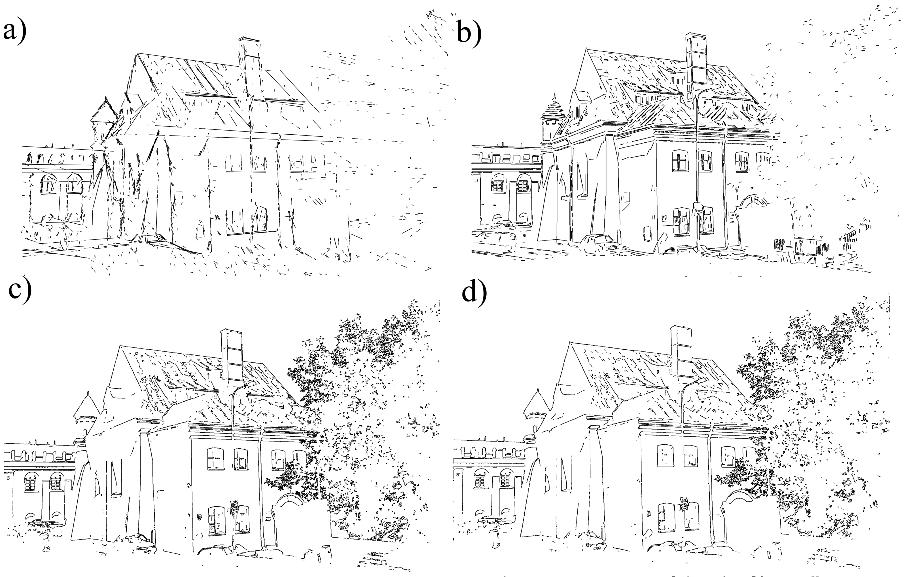

Figure 14. Contour detection with the use of: a) Hough transform, b) LSD method, c) Canny algorithm, d) Deriche algorithm

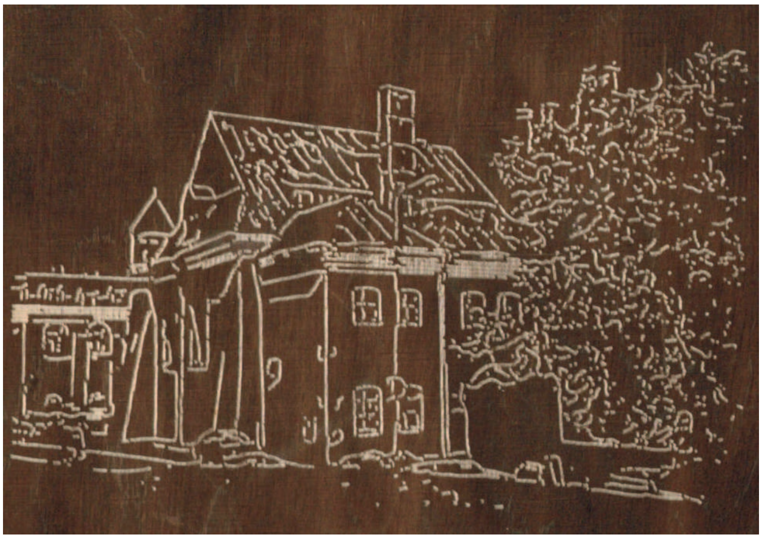

Figure 15. Drawing made in wood with CNC milling machines

reproduction of hair. In the case of the Canny and Deriche algorithms, such a problem is not observed. However, the thresholding which is needed with these methods results in some details of the drawing not being included. This is the case of the lines which were in the shade (e.g. the left wall of the building). The vectorised results from the Canny and Deriche algorithms tend to show a large number of details. Depending on the applications, this can be an advantage or a disadvantage to these methods. The vectorised image generated by the Canny algorithm consists of 2037 individual curves, the image generated by the Deriche algorithm contains 1784, the LSD 1309, and the image obtained by the proposed method has 1053 individual curves. In all cases, curves consisting of less than 5 points were rejected. In the case of the LSD method, the result is much better than that of the others, because of the poor 'vectorisation' of the tree. In typical applications, the number of individual curves is not important, but where a CNC milling machine is to be used it is crucial, since the image in vector form can be then engraved in metal or in wood. A photo of a sample image milled in wood is shown in Fig. 15.

Too many separate curves in a picture significantly extend the operating time of a CNC milling machine. Each separate curve requires the head of the milling machine to be raised, moved to another place and lowered. For short curves, this process will require more time than just milling. Another problem with a large number of separate curves is optimising the movement of the machine, so that the distance between the milled elements is as small as possible. The more curves there are on the drawing, the longer is the time required to calculate the best routing order. In the sample drawing it can be seen that the proposed

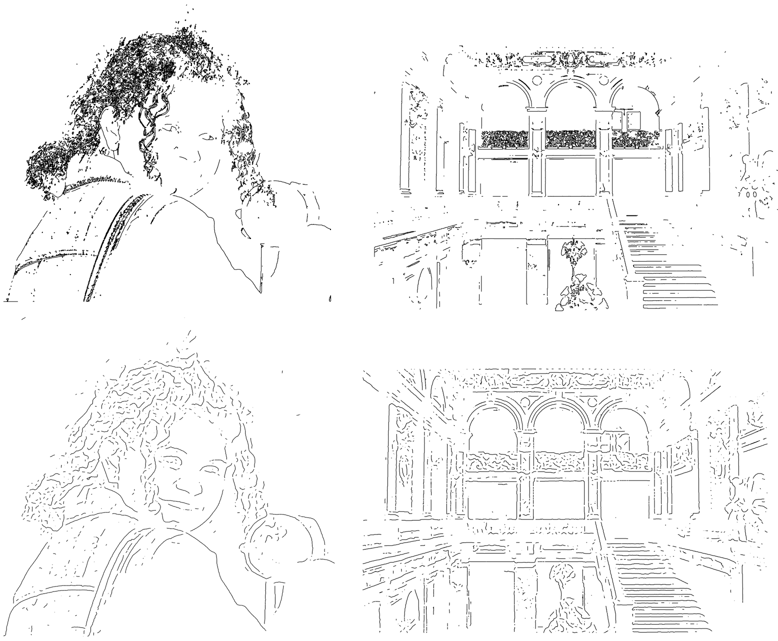

Figure 16. Sample result of the Deriche algorithm (top) and of the proposed method (bottom)

method has a tendency to combine lines running side by side into one line. In the case of images which are to be printed, this is a drawback, while in the case of milling, this is an advantage. Milling is done by a cutter of a certain thickness; however, it cannot be too thin, because then the drawing will be invisible. Because of this, lines running side by side are combined into a single thick line during the milling. Particularly problematic are the areas where a large number of lines run side by side. In Fig. 16, these will be the hair and railings. In the case of milling with the Deriche algorithm, the texture of these components will be lost, because the lines run too close to each other. The milling machine in that case will remove an entire layer of material. The surface here will be uneven, particularly in the case where wood is used; additional processing would then be required. In the case of the proposed method this will not be necessary, because the lines running close to each other are always connected.

5. Conclusions

In the article, a method of non-photorealistic rendering allowing for creation of an image from short segments of straight lines is presented. Images of this kind look like images carved on coloured ground with the use of a special tool. The method works especially well to reproduce buildings with uniform walls. An interesting effect can also be obtained for trees and hair. Data generated from this method can be used in the process of vectorisation. In comparison with other methods it gives a smaller number of separate curves and it joins curves which are very close to each other. In the case of applications for a CNC milling machine, these are very significant advantages because they affect the speed and quality of the drawings.

References

Arman, P.V. (2013) *Zanelle* /http://www.vanarman.com/S.

Borawski, M. (2012) Vector space of increments. *Control and Cybernetics*, **1**, 145–170.

Bronsztejn, I.. Siemiendiajew, K., Musiol, G. and Mühlig, H. (2004) *Nowoczesne kompendium matematyki (Modern Compendium of Mathematics).* PWN, Warszawa.

Canny, J. (1986) A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **86**, 679-698.

Curtis, C.J., Anderson, S.E., Seims, J.E., Fleischery, K.W. and Salesin, D.E. (1997) Computer–generated watercolor. *Proceedings of ACM SIGGRAPH.* ACM, 421–430.

DeCarlo, D. and Santella, A. (2002) Stylization and Abstraction of Photographs. *Proc. of ACM SIGGRAPH 02.* ACM, 769–776.

Deriche, R. (1987) Using Canny's criteria to derive a recursively implemented optimal edge detector. *Int. J. Computer Vision*, **1**, 167-187.

Fischer, J., Bartz, D. and Strasser, W. (2005) Stylized augmented reality for improved immersion. *VR'05. Proc. of the 2005 IEEE Conference on Virtual Reality.* IEEE Computer Society, 195–202.

Grompone von Gioi, R., Jereme, J., Morel, J. and Randall, G. (2012) LSD: a Line Segment Detector. *Image Processing On Line*, http://www.ipol.im/pub/art/2012/gjmr-lsd/.

Haeberli, P. (1990) Paint by numbers: Abstract image representations. *Proc. of ACM SIGGRAPH 90.* ACM, 207–214.

Kasao, A. and Miyata, K. (2006) Algorithmic Painter: a NPR method to generate various styles of painting. *Visual Comput*, **22**, 14-27.

Kang, H., Lee, S. and Chui, C.K. (2007) Coherent line drawing. *NPAR^{TM}07: Proceedings of the 5th international symposium on non-photorealistic animation and rendering.* ACM, New York, NY, USA, 43–50.

Lee, K. J., Kim, D. H., Yun, I. D. and Lee, S. U. (2007) Three-dimensional oil painting reconstruction with stroke based rendering. *Visual Comput*, **23**, 873-880.

Lindemeier T., Pirk S. and Deussen O. (2013) Image stylization with a painting machine using semantic hints. *Computers & Graphics*, **37**, 293-301.

Litwinowicz, P. (1997) Processing images and video for an impressionist effect. *Proc. of ACM SIGGRAPH 97.* ACM, 407–414.

Lu, C., Xu, L. and Jia, J. (2012) Combining Sketch and Tone for Pencil Drawing Production. *International Symposium on Non-Photorealistic Animation and Rendering (NPAR 2012).* Eurographics Associatin, Aire-la-Ville, 65–73

Kelly, L. and Marx, D. (2013) *The vangobot project.* http://vangobot.com/S.

Pratt, K.W. (2001) *Digital Image Processing.* A Wiley-Interscience Publication, New York.

Takagi, S., Fujishiro, I. and Nakajima, M. (1999) Volumetric modeling of colored pencil drawing. *Proceedings of Pacific Graphics.* IEEE Computer Society, 250–258.

Wang, S., Wu, E., Liu, Y., Liu, X. and Chen, Y. (2012a) Abstract line drawings from photographs using flow-based filters. *Computers & Graphics*, **36**, 224–231.

Wang, S., Ma, Z., Liu, X., Chen, Y. and Wu, E. (2012b) Coherence-enhancing line drawing for color images. *Science China Information Sciences*, **36**, 1-11.

Winkenbach, G. and Salesin, D.H. (1994) Rendering parametric surfaces in pen and ink. *Proceedings of ACM SIGGRAPH.* ACM, 91–100.

Xiao D. and Tao, Ch. (2007) A line detection algorithm based on error propagation. *Proceedings of the 26th Chinese Control Conference*, 493–497.