J Real-Time Image Proc (2007) 2:3–9
DOI 10.1007/s11554-007-0038-9

ORIGINAL RESEARCH PAPER

Real-time 2D to 3D video conversion

Ianir Ideses · Leonid P. Yaroslavsky · Barak Fishbain

Received: 2 April 2007 / Accepted: 1 August 2007 / Published online: 28 August 2007

Abstract We present a real-time implementation of 2D to 3D video conversion using compressed video. In our method, compressed 2D video is analyzed by extracting motion vectors. Using the motion vector maps, depth maps are built for each frame and the frames are segmented to provide object-wise depth ordering. These data are then used to synthesize stereo pairs. 3D video synthesized in this fashion can be viewed using any stereoscopic display. In our implementation, anaglyph projection was selected as the 3D visualization method, because it is mostly suited to standard displays.

Keywords Real-time · 3D · Anaglyph · Depth-maps · MPEG 4

1 Introduction

In recent years we are seeing great advances in the development of stereoscopic display methods. These advances include the use of autostereoscopic displays—displays that enable unaided 3D viewing and multi-view autostereoscopic displays—displays that show more than two view points for a given scene [1, 2].

While display technology has greatly advanced, the problem of content generation still lingers. Acquisition of stereoscopic content is still problematic, mainly due to the issues of temporal synchronization, as well as zoom and focal properties of the stereo setup. In addition, stereo setups do not enable the use of multi-view displays.

I. Ideses (✉) · L. P. Yaroslavsky · B. Fishbain
Department of Interdisciplinary Studies,
Tel Aviv University, 69978 Tel Aviv, Israel
e-mail: ianir@eng.tau.ac.il

One possible solution to this problem is conversion of 2D to 3D video. One method of conversion relies on simple time delay between frames and adjustment of left–right images [3]. In this method, computations are only necessary to align the images [4, 5] (in the case of anaglyph projection) and to assess which image corresponds to the left eye and which to the right. However, this method is mostly suited for videos that contain lateral or rotational motion, and even in these cases it does not allow to adjust image parallax with the speed of the movement. The ultimate approach is to use adjacent frames in order to synthesize depth maps. These depth maps can then be used to generate synthetic views, either a stereo pair for stereoscopic vision, or multi views for multi-view autostereoscopic displays. In addition, these depth maps can also be used for other applications, as they contain information on the 3D shape of the scene.

There are many methods to compute depth maps from a stereo pair (or adjacent video frames). Among them are the works of Lucas and Kanade [6], Horn and Schunck [7], Periaswamy and Farid [8], Wu et al. [9], Alvarez et al. [10], Schmidt [11] and Ran and Sochen [12]. This is a greatly advancing field and a lot of effort is guided in this direction.

The major drawback of all these methods is that they require extremely computationally intensive operations. While these may be feasible to implement in real-time on modern high-end computers or dedicated hardware [13–20], they are not suited for conversion of 2D to 3D video on low-end hardware or thin clients. Moreover, while some software solutions for depth map estimation (not including stereo synthesis) for low resolution images (QVGA) and hardware solutions for VGA resolution exist (http://www.videredesign.com), with the introduction and gaining popularity of high resolution HDTV, the amount of

computations has grown dramatically. It is obvious that any additional information that may help to determine optical flow without increasing the computational complexity or introducing more computational operations is desirable.

In this paper, we present a method to generate depth maps for 2D to 3D conversion in real-time utilizing the properties of modern compression standards. In this method, depth maps are constructed within the video decoding stage, requiring very little extra computations for their synthesis; in fact, depth maps can be created without decoding the data stream at all. This method assumes three-stage processing: (1) extraction of inter-frame disparity maps; (2) converting the inter-frame disparity maps into depth maps, and (3) generating, using the synthesized depth maps, artificial stereo (two or multiple view) images. This method was successfully implemented on standard hardware with real-time performance [21]. In addition, these depth maps can be used to complement other shape estimations and serve as a first estimation (these depth maps are only approximations of the true 3D geometry, sufficient for 3D visualization) for accurate iterative depth map estimation techniques.

2 Depth map calculation

In order to synthesize stereoscopic or multi-view videos, one must first acquire a depth map. Calculating a dense depth map is basically the process of finding the correspondence between a pixel location in one frame and its position in the other frame. This mapping of one image to the other one can be obtained by registering a spatial neighborhood, surrounding each pixel in one image, to the other. In this way a field of disparity vectors is recovered. Each pixel is then assigned its disparity. This can be performed in several methods, among them are correlational methods, optical-flow methods and hybrid methods as described in the previous section. These methods are usually very computationally expensive and not suited for computing high-resolution depth maps for video in real time. An alternative to direct computation of these depth maps is the extraction of motion vectors that exist in compressed video files. For this work, MPEG 4 (H.264) was selected because of its ability to compute motion vectors with ¼ pixel accuracy for blocks as small as 4×4 pixels [22, 23].

2.1 Motion vector extraction

MPEG 4 (H.264) is a modern compression standard that uses both temporal and spatial compression. While spatial compression is basically a form of JPEG compression, it is temporal compression that enables the high compression rates of MPEG 4.

In temporal compression, each frame is divided into blocks and block search is performed between adjacent frames for the location of the blocks. In this fashion, it is necessary to store the movement of the block from one frame to the other, thus reducing the amount of information to store.

MPEG 4 enables computation of motion vectors in blocks as small as 4×4 pixels with quarter pixel accuracy [22, 23]. These data are very useful for depth estimation. In its simplest form, the horizontal, X-axis, motion vectors can be used as depth data. This holds for cases where there is only lateral motion on the X-axis and no scene motion is present (a canonical stereo setup). For other motion types it is necessary to make a transformation from motion vector maps to depth maps. In our implementation, motion vector maps were extracted as a part of the MPEG 4 encoder schema. The encoder was instructed to extract motion vectors for every frame (regardless of the ultimate frame type) with the minimal block size.

2.2 Transformation of motion vector maps to depth maps

In some cases, motion vector maps can be directly treated as depth maps. This approximation holds when the two images/frames are taken in parallel viewing or when they are acquired in small ranges of disparity in the case of epipolar acquisition. This is usually the case in computation of depth maps of 2D video. However, there are many cases where this approximation does not hold. This happens when the motion is either too rapid in terms of camera rotation or in the case of camera zoom. Such cases can be detected by analysis of the motion vector maps and dealt with.

In the case of zoom, it is necessary to change the dynamic range of the depth values (while cropping out border pixels), the amount of dynamic range scaling has to be congruent with the zoom factor, for the purpose of visualization this has to be visually comfortable rather than true-to-life accurate. In the case of rotation around a specific object, one needs to invert the disparity values, so that the close object receives high disparity values although it appears to be static. Other depth values should remain the same. This is a non-trivial case and indeed is error prone.

In our implementation, we treated motion as a sole depth cue, namely, we calculated the depth solely on the values of the X and Y motion vector values. The depth was estimated by

$$D(i,j) = c\sqrt{\mathrm{MV}(i,j)_x^2 + \mathrm{MV}(i,j)_y^2} \quad (1)$$

where $D(i,j)$ is the depth value for pixel (i,j) and $\mathrm{MV}(i,j)_x$, $\mathrm{MV}(i,j)_y$ are the X and Y motion vectors values for that pixel, respectively, and $\boldsymbol{c}$ is a custom defined scale parameter.

The scaling parameter $\boldsymbol{c}$ can be utilized in two ways, either automatically, or set as a user-selected factor. In our implementation, both methods are supported. One may opt to scale the parameter to fit the maximal disparity over all frames—simply adding a constant gain to the depth map values, or perform automatic scaling unto some predefined parallax, keeping maximal parallax constant in all frames. In essence this operation stretches the dynamic range of all depth maps to this level (a nominal parallax value for comfortable viewing is of the order of 20 pixels). In order to have motion vectors for every pixel (the MPEG standard assigns motion vectors to blocks) nearest neighbor interpolation was used. Using this simple interpolation does not significantly reduce the quality of the resulting 3D video as was shown by Yaroslavsky et al. [24, 25] and does not require any extra computation. Although more accurate and still computationally simple interpolation methods such as linear interpolation are also feasible, we opted not to use them in order to keep extra computations to minimum.

3 Video synthesis

It is known that using a depth map and one of the images of the stereo pair, it is possible to reconstruct the stereo pair for autostereoscopic display or an anaglyph. In order to generate these synthetic 3D images, methods that rely on image resampling are used. In our implementation, the image is oversampled four times in the X-axis to enable the use of the quarter pixel accuracy and then resampled by a grid that is controlled by the depth map.

Interpolation is based on the same scheme that MPEG 4 employs; namely, the image is interpolated to double size using a six-tap filter and then bilinear interpolation to achieve four times interpolation. This interpolation was chosen because this type of interpolation is efficiently implemented within the MPEG CODEC.

In this manner we are able to simulate the disparity that can be observed in the stereo pair. It should be noted that this method does not guarantee the resulting stereo pair to be identical to the original stereo pair, due to the depth map inaccuracies and the cases of occlusion. Furthermore, because of the different views, the right image contains pixels that cannot be seen in the left image and vice versa—this due to the limited field of view. These details cannot be recreated using a resampling process, be it as accurate as possible. However, for 3D visualization, synthetic views are sufficient for 3D perception.

A flow chart of this process is shown in Fig. 1.

4 Visualization

3D video can be visualized in many ways, among them are autostereoscopic displays, shutter glasses, polarizing glasses, and anaglyphs. Anaglyphs are the most economical and easily attainable method for 3D visualization and most suited for viewing using standard hardware. This method, as opposed to other techniques such as polarized glasses, or shutter glasses, requires no special display hardware and the glasses can be made from simple materials found in the hobby shops.

Anaglyphs produce a visual effect of 3D images when viewed using color-filtering spectacles. Synthesis of anaglyphs is a simple process in which the red channel in one image is replaced by the red channel of the second image of the stereo pair. Because conventional anaglyphs usually suffer from ghosting effects, in our implementation we used several techniques to improve the visual quality of these images [4]. Specifically, defocusing of the red channel and depth map compression were used.

5 Data

In our experiments, video sequences were acquired using standard digital cameras and saved as motion JPEG (MJPEG) sequences. These were then separated to JPEG image frames, so that adjacent frames could be dealt with as stereo pairs. For testing different motion types, the camera was moved along the X- and Y-axes. Rotation around an object was also tested. In addition, we tested our method on public broadcast video.

Our implementation accepts as input a series of frames to be encoded, however, it can also be used to calculate depth maps for stereo pairs by simple interleaving of the still images. Our tests were performed, both on video frames and on stereo pairs. Examples of video frames and a stereo pair can be seen in Figs. 2 and 3, respectively.

6 Results

The images were fed to the video encoder and disparity maps were computed for every frame/stereo pair. The encoder's output included a standard MPEG 4 stream, as well as X and Y disparity maps. These disparity maps were then used to synthesize depth maps and 3D video from the compressed video stream in real-time (25 fps) for QVGA sized videos on a standard P4 2.8 GHz PC. This performance relates both to video decoding and conversion. Analysis of the computational complexity shows that most of the computations are spent in the standard video decoding stage, the extra computation

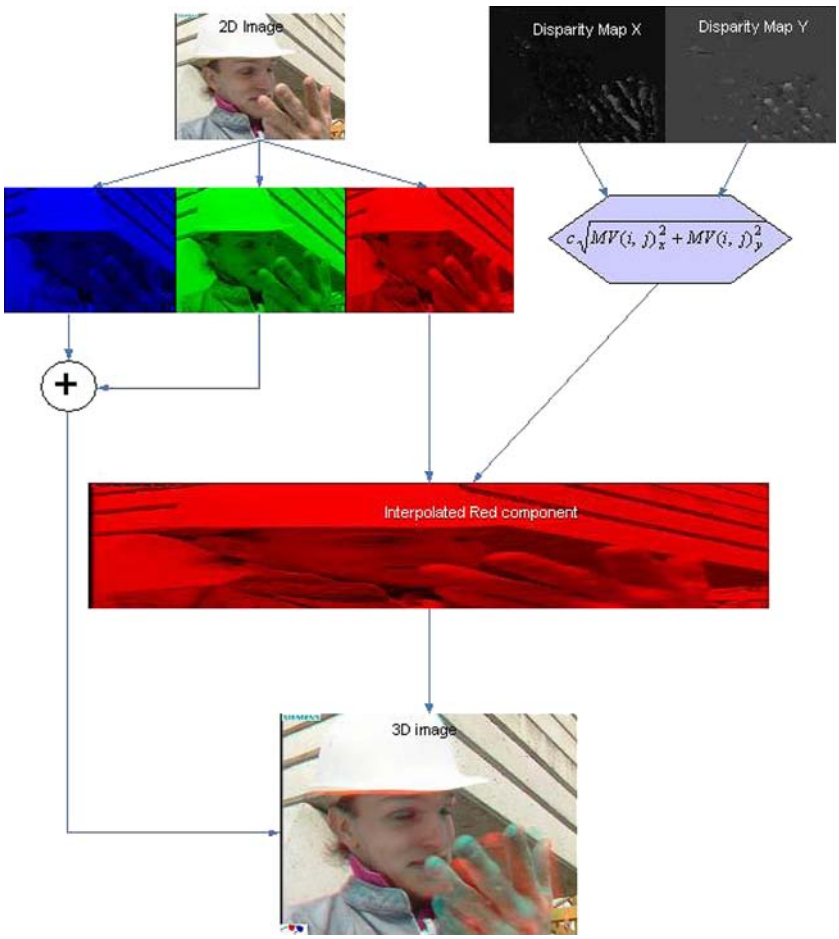

Fig. 1 Flow diagram of 2D to 3D conversion for anaglyph display. The incoming image is split to its RGB components. For anaglyphs display it is sufficient to synthesize only the left channel of the artificial stereo pair. This is performed by resampling the red channel according to the depth map. Finally the channels are merged together to form the anaglyph

Fig. 2 Frames taken from a video sequence

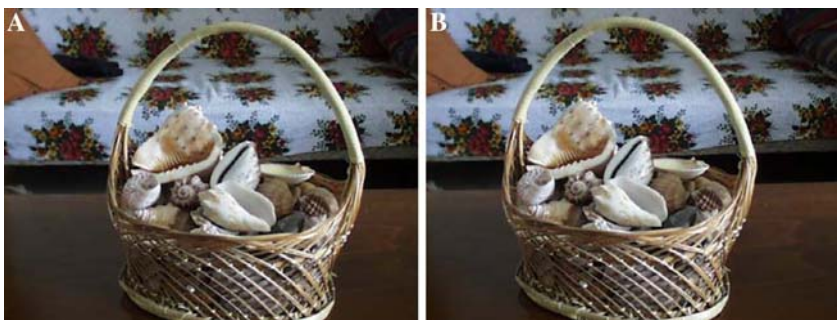

Fig. 3 A stereo pair

required to sample the image prior to resampling (efficiently coded within the MPEG decoder) and the operation performed on the X and Y motion fields (in Eq. 1) are far lower than that of the MPEG decompressor. In principle, between these two operations, the most computationally expensive is the motion field

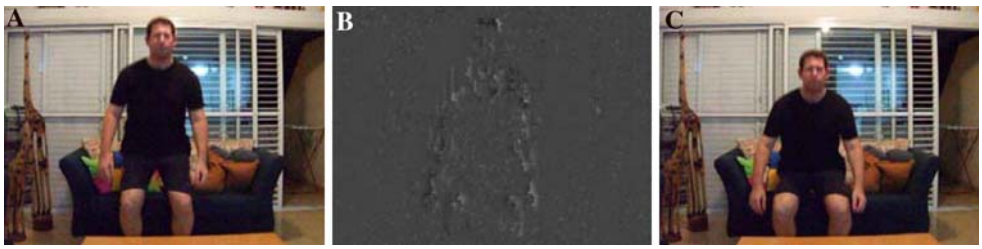

Fig. 4 Video frames (*left* and *right* images) and the corresponding depth map (*center* image). Although the resulting depth map does not show the exact metrics of the stereo pair, it is sufficient for the purpose of visualization

Fig. 5 Comparison of the MPEG based depth map to other commonly used optical flow method. It can be seen that the MPEG depth map contains the important motion elements. The Lucas & Kanade implementation and elastic registration in this case were able to produce a similar depth map (although with some missing details). It should be noted that the MPEG based depth map does include noise artifacts and should be smoothed prior to being used for visualization

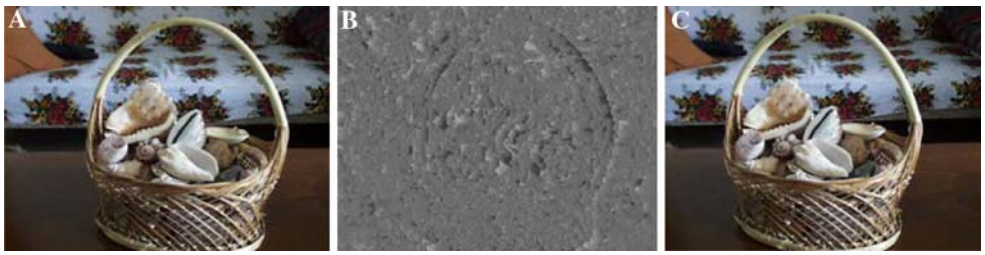

Fig. 6 Stereo images (*left* and *right* images) and the corresponding depth map (*center* image)

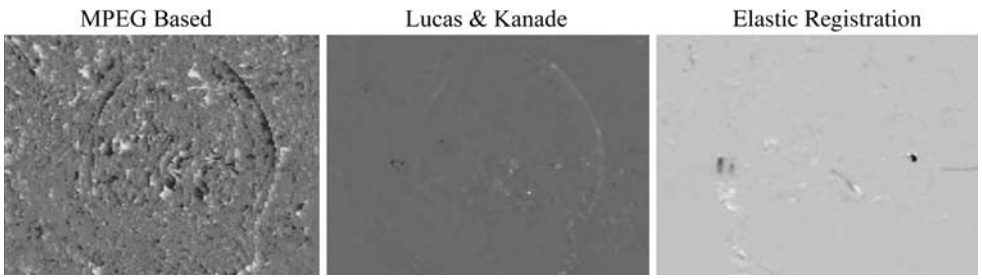

Fig. 7 Comparison of the MPEG based depth map to other commonly used optical flow method. It can be seen that the MPEG depth map contains the important motion elements that are missing in the Elastic Registration implementation. The Lucas & Kanade algorithm in this case was able to produce a similar depth map. It should be noted that the MPEG based depth map does include noise artifacts and should be smoothed prior to being used for visualization

computation—one square root operation, one addition operation and two operation of raising to the power of two per image pixel.

An example of a depth map generated from video frames is shown in Figs. 4 and 5 which show a comparison of the depth maps to those attained by other optical flow methods. An example of a stereo pair and a resulting depth map is shown in Fig. 6, resulting depth maps can be compared to other optical flow methods in Fig. 7.

By performing the processing described in previous sections, we were able to reconstruct the stereo pair and generate anaglyphs, shown in Figs. 8 and 9.

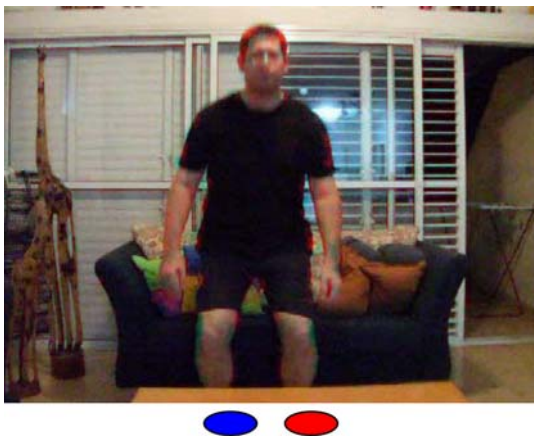

Fig. 8 Color anaglyph with P-law ($P = 0.5$) compressed depth map

7 Future work

The depicted algorithm has been shown to produce quite good depth maps and 3D visualization for several motion types—horizontal and vertical translation and camera rotation and zoom.

Like all shape from motion algorithms, this algorithm fails to extract shape when no motion is apparent. Given a static scene (static camera and no object motion), no depth can be recovered and 3D perception would be impossible. One method to retain 3D information in videos in this case is by identifying this scenario and repeating the depth maps that were computed until then. If no depth maps were computed, no depth can be visualized.

Another inherent drawback to this method is that a remote object that moves with great velocity can be interpreted as closer than a closer slow moving object.

In these cases, other shape extraction algorithm such as shape from focus, shape from occlusion and shape from perspective are in order.

Fig. 9 Color anaglyph with P-law ($P = 0.5$) compressed depth map

8 Conclusions

In this paper we have described a method to generate high quality anaglyphs from compressed video sequences. Our method relies on computation of depth maps from adjacent video frames. For this purpose we extracted the motion vectors found in the MPEG 4 standard and transformed them into depth maps. We demonstrate this ability for stereo pairs, either as a set of images acquired using a still camera and interleaved for MPEG compression, or adjacent frames extracted from a video stream.

Video motion, as opposed to stereo pairs usually presents a greater challenge than still images, due to the nature of motion that may vary from frame to frame. In order to tackle this we used a temporal distance that was small enough to allow stereo pair approximation.

Visualization of the 3D videos can be achieved using any 3D display device available. In our implementation we used anaglyphs because of their suitability to standard display hardware.

Examples of video streams converted to 3D are available on the web (http://www.eng.tau.ac.il/~ianir/3DVideo.html). Limitations of this algorithm such as cases of no motion are acknowledged and left for future research.

References

1. Blundell, B., Schwarz, A.: Volumetric Three Dimensional Display Systems. Wiley, New York (2000)
2. Halle, M.: Autoestereoscopic displays and computer graphics. Comput. Graph. (ACM) **31**, 58–62 (1997)
3. Ideses, I., Yaroslavsky, L.: A method for generating 3D video from a single video stream. VMV 2002 435–438 (2002)
4. Ideses I., Yaroslavsky L.: 3 methods to improve quality of colour anaglyphs. J. Optics. A: Pure, Applied Optics 7(12), 755–762 (8) (2005)
5. Ideses, I., Yaroslavsky, L.: New methods to produce high quality color anaglyphs for 3-D visualization. In: Image Analysis and Recognition: International Conference ICIAR 2004, Lecture Notes in Computer Science. pp. 273–280. Springer, Heidelberg (2004)
6. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of 7th International Joint Conference on Artificial Intelligence (IJCAI), pp. 674–679 (1981)
7. Horn, B., Schunck, B.: Determining optical flow. Artif. Intell. **17**, 185–203 (1981)
8. Periaswamy, S. Farid, H: Elastic registration in the presence of intensity variations. IEEE. Trans. Med. Imaging. **22**(7) (2003)
9. Wu, Y.T., Kanade, T., Li, C.C., Cohn, J.: Image registration using wavelet-based motion model. Int. J. Comput. Vis. (2000)
10. Alvarez, L., Deriche, R., Sanchez, J., Weickert, J.: Dense disparity map estimation respecting image discontinuities: a PDE and scalespace based approach. Technical Report RR-3874, INRIA (2000)
11. Schmidt, J., Niemann, H., Vogt, S.: Dense disparity maps in real-time with an application to augmented reality. In: IEEE

Workshop on Applications of Computer Vision (WACV 2002), 3–4 December 2002. IEEE Computer Society, Orlando
12. Ran, A., Sochen, N.A.: Differential Geometry Techniques in Stereo Vision Proceedings of EWCG, pp. 98–103 (2000)
13. Corke, P., Dunn, P.: Real-Time Stereopsis Using FPGAs, IEEE TENCON—Speech and Image Technologies for Computing and Telecommunications, pp. 235–238 (1997)
14. Faugeras, O. et al.: Real time correlation based stereo: algorithm, implementations and applications. INRIA Technical Report 2013 (1993)
15. Kimura, S., Kanade, T., Kano, H., Yoshida, A., Kawamura, E., Oda, K.: CMU video-rate stereo machine. Proceedings of Mobile Mapping Symposium (1995)
16. Konolige, K.: Small vision systems: hardware and implementation. In: Eighth International Symposium on Robotics Research, Hayama, Japan (1997)
17. Kimura, S., Shinbo, T., Yamaguchi, H., Kawamura, E., Naka, K.: A convolver-based real-time stereo machine (SAZAN). CVPR, pp. 457–463 (1999)
18. Matthies, L.: Stereo vision for planetary rovers: stochastic modeling to near realtime implementation. Int. J. Comput. Vis. **8**, 71–91 (1992)
19. Mulligan, J., Daniilidis, K.: Real-time trinocular stereo for tele-immersion. ICIP (2001)
20. Woodfill, J., Von Herzen, B.: Real-time stereo vision on the PARTS reconfigurable computer. In: Proceedings of IEEE Workshop FPGAs for Custom Computing Machines, pp. 242–250 (1997)
21. Ideses, I.P., Yaroslavsky, L.P., Vistuch, R., Fishbain, B.: 3D video from compressed 2D video. In: Proceedings of Stereoscopic Displays and Applications XVIII. SPIE and IS&T, San Jose, CA (2007)
22. Ohm, J.R.: Stereo/multiview video encoding using the MPEG family of standards. In: Merritt, O.J., Bolas, M.T., Fisher,S.S., (eds.) The Engineering Reality of Virtual Reality, vol. 3639, pp. 242–253. SPIE, San Jose (1999)
23. Wiegand, T., Sullivan, G.J., Bjøntegaard, G., Luthra, A.: Overview of the H.264/AVC video coding standard. IEEE. Trans. Circ. Syst. Video Technol. **13**(7), 560–576 (2003)
24. Yaroslavsky, L.P., Campos, J., Espínola, M., Ideses, I.: Redundancy of stereoscopic images: experimental evaluation. Opt. Express. **13**, 10895–10907 (2005)
25. Yaroslavsky, L.P.: On redundancy of stereoscopic pictures. In: Proceedings of Image Science '85, Helsinki, Finland, 11–14 June 1985, vol. 1, pp. 82–85. Acta Polytechnica Scandinavica, no. 149 (1985)

Author Biographies

Ianir A. Ideses is a Ph.D. student in the School of Electrical Engineering in Tel Aviv University, researching 3D visualization, synthesis and compression. Ianir holds an M.Sc. degree in Electrical Engineering from Tel Aviv University (Magna cum laude, 2004) and a B.Sc. degree in Electrical Engineering from the Technion, Israel's Institute of Technology (1998).

Leonid P. Yaroslavsky MS (Summa cum laude, 1961), Ph.D. (1968), Dr. Sc.-Phys. Math. (1982). Till 1995, he had headed a Laboratory of Digital Optics at the Institute for Information Transmission Problems, Russian Academy of Sciences. From beginning of 1995, he is a Professor at Department of Interdisciplinary Studies, Faculty of Engineering, Tel Aviv University. He has authored several books and more than 100 papers on digital image processing and digital holography. He is also a Fellow of Optical Society of America.

Barak Fishbain is a Ph.D. student in the school of Electrical Engineering in Tel Aviv University, researching video enhancement through super resolution and motion estimation algorithms for traffic monitoring and remote sensing applications videos. Barak holds an M.Sc. degree in Electrical Engineering from Tel Aviv University (2004) and a B.Sc. degree in Electrical Engineering from the Technion, Israel's institute of Technology (1998).