Journal of Electronic Imaging

JElectronicImaging.org

Verification of visual odometry algorithms with an OpenGL-based software tool

Piotr Skulimowski
Pawel Strumillo

SPIE. IS&T imaging.org

Verification of visual odometry algorithms with an OpenGL-based software tool

Piotr Skulimowski* **and Pawel Strumillo**
Lodz University of Technology, Institute of Electronics, 211/215 Wolczanska Street, Lodz 90-924, Poland

Abstract. We present a software tool called a stereovision egomotion sequence generator that was developed for testing visual odometry (VO) algorithms. Various approaches to single and multicamera VO algorithms are reviewed first, and then a reference VO algorithm that has served to demonstrate the program's features is described. The program offers simple tools for defining virtual static three-dimensional scenes and arbitrary six degrees of freedom motion paths within such scenes and output sequences of stereovision images, disparity ground-truth maps, and segmented scene images. A simple script language is proposed that simplifies tests of VO algorithms for user-defined scenarios. The program's capabilities are demonstrated by testing a reference VO technique that employs stereoscopy and feature tracking. [DOI: 10.1117/1.JEI.24.3.033003]

Keywords: stereovision; OpenGL; egomotion estimation; visual odometry; passive navigation.

Paper 14751 received Nov. 26, 2014; accepted for publication Apr. 13, 2015; published online May 7, 2015.

1 Introduction

Motion parameters of an object (further termed egomotion parameters) are given by a set of kinematic quantities defining the object's movement in relation to its environment. Robust estimation of egomotion parameters is an important problem in robotics, automatic navigation, and computer vision systems.[1] Egomotion parameters are essential in planning movement paths in obstacle avoidance systems,[2] structure from motion methods,[3,4] and in simultaneous localization and mapping.[5] Although there are various dead-reckoning techniques available for estimating egomotion parameters, e.g., odometry, inertial, and laser sensing, the vision-based techniques termed visual odometry (VO)[6] are continuously gaining in importance.[7] This is because imaging sensors (including stereo imaging) offer passive data acquisition techniques that work in natural lighting conditions, and are inexpensive and are miniatured. Furthermore, efficient algorithms have been worked out for calibrating camera systems,[8] tracking image objects,[9,10] three-dimensional (3-D) scene reconstruction,[11] and recognition.[12] Video imaging also becomes an important modality in multisensor navigation systems incorporating inertial sensors[13] and the global positioning system.[14]

VO systems can be divided into two major groups: single camera systems (also termed monocular VO) and multicamera systems.

Monocular VO algorithms primarily use feature tracking methods. In Ref. 15, a 1-point-random sample consensus (RANSAC) algorithm was employed for estimating vehicle motion from single camera image sequences. In Ref. 16, real-time tracking of camera position is estimated by applying the particle filter combined with the unscented Kalman filter. In Refs. 17 and 18, approaches to estimating rotation, scale, and focal length parameters were presented for motions devoid of translational movements. On the other hand, in Ref. 19, a method based on optical flow for estimating only the translational motion vector was proposed. In Ref. 20, a special, omnidirectional camera for maximizing the field of view (FOV) was used, which allowed a more flexible choice of the tracked keypoints. An algorithm for estimating six degrees of freedom (6 DoF) motion vector was proposed in Ref. 13; however, the image scale information was derived from an altimeter.

Most of the VO-based egomotion estimation algorithms are dedicated to multi camera systems.[6,7,21] These methods incorporate depth maps that allow for significant simplification of computations involved in egomotion estimation. In Ref. 21, to obtain an accurate global position and to achieve robustness to motion blur, multilevel quadtrees and true scale scale-invariant feature transform descriptors were used. In Ref. 7, an iterative closest point algorithm was implemented in real time for 6DoF egomotion estimation by tracking keypoints detected by the Shi–Thomasi feature detector. In Ref. 6, the RANSAC algorithm was used to estimate the motion of a stereovision unit (720×240 pixels image resolution and baseline 28 cm) mounted on an autonomous ground vehicle. Also, in Ref. 22, a stereovision system with a wider baseline (approximately 40 cm) was mounted on a vehicle and the employed algorithm allowed for estimating the vehicle's motion path in rough terrain.

Although many efficient vision-based egomotion estimation algorithms were developed (for single or multicamera systems), objective verification and comparison of their performance is a difficult task. Building a robot-like device that would generate arbitrary 6DoF motion trajectories is expensive and would offer limited precision, repeatability, and range of movements. One possible solution for verification of egomotion estimation algorithms, as proposed in Refs. 15, 21, and 23, is to use predefined image sequences. In Ref. 21, a sequence of 50 stereo frames was captured by a stereovision unit moving along a straight path. Location of the

*Address all correspondence to: Piotr Skulimowski, E-mail: pskul@p.lodz.pl

system was advanced by 20 cm with each image recording. The accuracy of the system was verified by comparing consecutive locations that were estimated by VO techniques to ground-truth values. The obvious shortcoming of such a verification method is the path shape that is constrained to line segments. In Ref. 15, vehicle movement in virtual urban terrain is simulated to test the RANSAC-based structure from motion algorithms. Facades of buildings were modeled by surfaces spanned on 1600 uniformly scattered keypoints. Image sequences of the simulated scene were captured by rotating a stereovision system with one camera located at the center of rotation. Again, the movement path of the system is limited to a simple rotational motion. In Ref. 23, a method for detecting objects in stereoscopic images of scenes was proposed. For testing the accuracy of the algorithm, a 600-frame simulated image sequence of an urban scene was rendered and the corresponding ground-truth disparity maps were generated.

There have also been special image databases built (see Ref. 24) that contain depth maps and the corresponding stereovison images; however, the number and type of imaged scenes are limited.[25] Moreover, the offered databases contain a small number of short, few seconds long sequences of image pairs, and they do not contain the ground-truth depth maps and the segmented images.

Although it is possible to generate synthetic data by using 3-D modeling software, the resulting models are not suitable for direct and straightforward generation of the depth maps and the segmented image. Such a procedure requires advanced programming skills from the user.

In this paper, a software tool for verification of VO algorithms is proposed. The software enables generation of user-defined 3-D scenes with sequences of egomotion parameters of a freely moving stereovision camera in custom defined scenes and recording of the corresponding stereoscopic and ground-truth disparity maps. An additional feature of the program is the availability of image sequences in which the scene objects are segmented out and labeled, which allows testing the segmentation algorithms. The program is written in C++ with the use of open graphics library (OpenGL). The tool, that is made freely available on a webpage,[26] features a simple interpreter for defining scene objects and the motion parameters of the user selected perspective camera model. Our software tool enables the generation of much longer image sequences than the existing databases and contains the ground-truth maps and the corresponding segmentation images. Test sequences and scripts used for their generation are available from the webpage of the project.

The paper is organized as follows. In Sec. 2, a very brief introduction to stereovision is given. In Sec. 3, a basic algorithm for egomotion estimation from stereoscopic VO is presented. This algorithm will serve as a reference method tested by the proposed software tool presented in Sec. 4, in which the main features of the developed program are described. In Sec. 5, the test results are outlined and Sec. 6 summarizes the advantages of the proposed software tool and its possible applications.

2 Stereovision Basics

Viewing the environment from more than a single spatial location allows for recovering 3-D relationships of imaged scenes that facilitate computation of egomotion parameters of a moving camera system. In computational stereo, two or more planar projections are recorded and the obtained images are compared to reconstruct 3-D scene geometry. Due to the difficulties in precise alignment of relative orientations of the cameras and the limited precision of their optics, the scene reconstruction task needs to be preceded by appropriate camera calibration procedures, see Ref. 27. Once internal camera geometries are corrected and external geometries rectified, the stereovision system is termed to be calibrated.[27]

For a calibrated, nonverged two-camera system, a scene point $P(X, Y, Z)$ placed within the FoV of the cameras is projected onto image points $p_L(x_L, y)$ and $p_r(x, y)$ in the left and right cameras correspondingly. Note that y coordinates of the scan lines of these two points are the same for a calibrated stereovision system. Hence, the depth of a scene point P can be estimated from the shift $d = x_L - x$ termed disparity. If frontal pinhole camera models are adopted, origins of the image coordinate systems oxy are located in image centers and the origin of the space coordinate system $OXYZ$ is positioned in the focal point of the right camera (see Fig. 1), the relation between the coordinates of a space point $P(X, Y, Z)$ and the coordinates of its projection onto the right camera image plane is given by[27]

$$x = f\frac{X}{Z}, \quad y = f\frac{Y}{Z}, \quad Z = \frac{Bf}{d}, \tag{1}$$

where f is the focal length of the camera, B is the baseline of the stereovision system (i.e., distance between the cameras' optical axis), and Z is the depth of point P.

Determining disparity for a given scene point is termed the correspondence problem and it can be solved by applying various methods as reviewed in Ref. 11. From local methods, the block matching techniques are frequently used because they lend themselves well to parallel computations, e.g., on GPU platforms.[28] By computing disparities for all imaged scene points within the FOV, a so-called disparity map can be obtained, see Fig. 2. Then the depth of scene points can be directly computed from the right hand side of Eq. (1). However, due to quantized values of disparities (for digital cameras), the corresponding depths are also quantized with a resolution that decreases with increasing depths. Access to the ground-truth depth map allows for verification of the algorithms for computing the depth maps. It should be

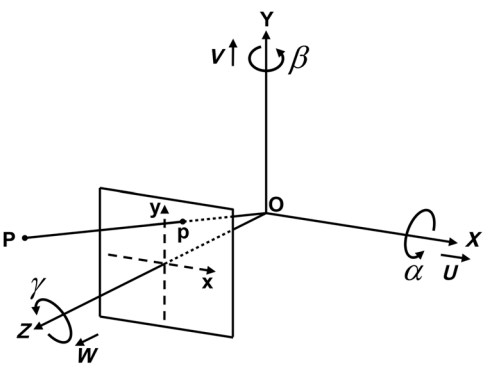
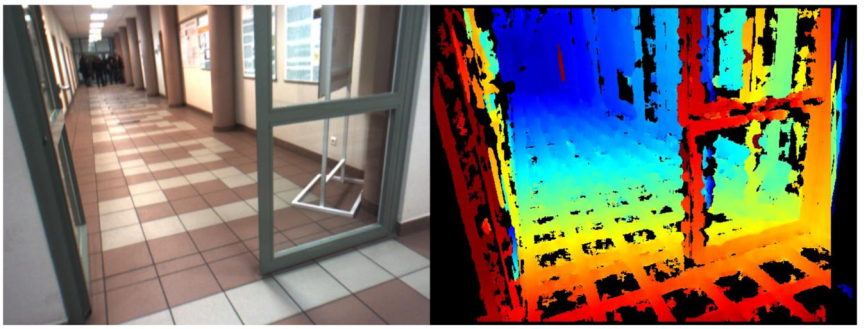


Fig. 1 The $OXYZ$ frame of reference and the image coordinate system oxy with the indicated translation motion parameters (U, V, W) and rotation motion parameters (α, β, γ).



(a) (b)

Fig. 2 (a) Image taken by the right camera of a stereovision system after compensation of geometric distortions and (b) the corresponding disparity map displayed in pseudocolors. Black regions denote areas of the map for which disparity could not be calculated with sufficient confidence.

noted, however, that estimation of the precision of the depth map calculated for real scenes (e.g., as shown in Fig. 2) is, in practice, impossible without the knowledge of the scene model and precise coordinates of the cameras in that scene.

3 Estimation of Egomotion Parameters from Stereovision Sequences

Consider a coordinate frame of reference $OXYZ$ whose origin is located at a focal point of the right camera of a stereovision system (Fig. 1). Assume the system moves freely in a space in which there are no moving objects. Translational movement of the system in the defined coordinate frame is described by a vector of translation velocities $[U, V, W]$ and a vector of angle velocities $[\alpha, \beta, \gamma]$ as shown in Fig. 1. Hence, the system can be defined as a rigid object with 6DoF.

The objective is to estimate the motion parameters of the stereovision system from the optical flow of the scene points projected onto the pair of images. Assume $[u, v]$ denotes the velocity vector of point $p(x, y)$ in the right camera image. The relationship between the velocity vector $[u, v]$ and the system motion parameters is given by (see derivation in Refs. 4, 29, and 30)

$$u = \frac{-Uf + xW}{Z} + \alpha\frac{xy}{f} - \beta\left(\frac{x^2}{f} + f\right) + \gamma y,$$
$$v = \frac{-Vf + yW}{Z} + \alpha\left(\frac{y^2}{f} + f\right) - \beta\frac{xy}{f} - \gamma x. \quad (2)$$

Assuming a sufficiently high video frame rate, the optical flow of image points can be assumed to be linear and approximated by the following difference equation:

$$x' = x + u, \quad y' = y + v, \quad (3)$$

where $p(x, y)$ is a coordinate of point p at frame t and (x', y') is the new coordinate of this point in frame $t + 1$.

By determining the optical flow for P ($P \geq 3$) image points, the six-element motion vector of the stereovision system can be computed by finding the minimum of the following cost equation:

$$E = \sum_{i=1}^{P}[(u_i + x_i - x_i')^2 + (v_i + y_i - y_i')^2]. \quad (4)$$

By substituting appropriate expressions from Eqs. (1) and (2)

$$a_{1_i} = \frac{-f}{Z_i} \qquad a_{2_i} = 0 \qquad a_{3_i} = \frac{x_i}{Z_i}$$
$$a_{4_i} = \frac{x_i y_i}{f} \qquad a_{5_i} = -\left(\frac{x_i^2}{f} + f\right) \qquad a_{6_i} = y_i$$
$$b_{1_i} = 0 \qquad b_{2_i} = \frac{-f}{Z_i} \qquad b_{3_i} = \frac{y_i}{Z_i}$$
$$b_{4_i} = \left(\frac{y_i^2}{f} + f\right) \qquad b_{5_i} = -\frac{x_i y_i}{f} \qquad b_{6_i} = -x_i$$

one gets

$$E = \sum_{i=1}^{P}[(a_{1_i}U + a_{3_i}W + a_{4_i}\alpha + a_{5_i}\beta + a_{6_i}\gamma + x_i - x_i')^2 + (b_{2_i}V + b_{3_i}W + b_{4_i}\alpha + b_{5_i}\beta + b_{6_i}\gamma + y_i - y_i')^2]. \quad (5)$$

The following matrix equation is obtained after partial derivative equations with respect to the searched parameters of the motion vector $\mathbf{X}^T = [U, V, W, \alpha, \beta, \gamma]$ are evaluated and equated to zero:

$$\mathbf{HX} = \mathbf{K}, \quad (6)$$

where elements h_{mn} and k_m of matrices $\mathbf{H}_{6\times6}$ and $\mathbf{K}_{6\times1}$ are defined as follows:

$$h_{mn} = \sum_{i=1}^{P}(a_{m_i}a_{n_i} + b_{m_i}b_{n_i})$$

$$k_m = \sum_{i=1}^{P}[a_{m_i}(x_i' - x_i) + b_{m_q}(y_i' - y_i)].$$

Matrix Eq. (6) needs to be solved for $\mathbf{X}$ for each new incoming stereoscopic image frame.

3.1 *Selection of Image Keypoints for Egomotion Estimation*

Estimation of six egomotion parameters requires the determination of the optical flows of at least $P = 3$ image points (further termed keypoints). However, for a robust estimation of the egomotion parameters, the following conditions need to be fulfilled:

- the keypoints should be selected according to the adopted stereo-matching algorithms, e.g., for the block methods the keypoints should correspond to corners, edges, characteristic local color regions, or textures;[31]
- the number of the tracked keypoints should be sufficiently large to minimize the quantization effect of their coordinates;
- the depth of the keypoints should be computed with sufficient precision (gross mistakes should be eliminated), hence, due to a hyperbolic relation between the disparity and depth [see Eq. (1)], the keypoints featuring a small depth are preferred.

The Shi–Tomashi algorithm, derived from the Harris detector[32] and implemented in the OpenCV library, was applied for detecting the keypoints.[33] For tracking the keypoints in consecutive video frames, a full block search matching method, similar to the one used in MPEG compression standard, was used. The optical flow $[u, v]$ of a given keypoint is thus determined by searching for a coordinate of a block of size $M \times M$ in current frame $t + 1$ that best matches the block corresponding to a keypoint $p(x, y)$ in frame t. The minimum of the sum of absolute difference (SAD) of the blocks' pixels is used as the block matching criterion. In Fig. 3, a single video frame is shown with the highlighted optical flow vectors of the keypoints.

4 Generating Stereovision Sequences Using OpenGL

Here, we propose a software tool for generating user-defined arbitrary motion paths that can be used for testing user VO algorithms. Because the tool employs OpenGL depth buffering, a short introduction to its role in rendering 3-D scenes is given.

Fig. 3 Scene image and optical flow vectors of the keypoints indicated by white line segments.

4.1 *Z-Buffer*

OpenGL is a universal programming library used for generating and rendering 2-D and 3-D scenes.[34] The basic tools used for generating 3-D graphics are the two buffers: the color buffer which stores an image array for display and the depth buffer (termed the Z-buffer) which stores each pixel's depth.[35] The Z-buffer stores the depth of a scene point from a pool of candidate scene points that has the smallest depth and picks it up for rendering. This process, termed Z-culling, eliminates the need for rendering hidden scene elements. If the depth values in the Z-buffer are stored with N-bit precision, the depth is quantized to $[0, 2^N - 1]$ levels. The values z_b stored in the Z-buffer are related to metric depth values Z of the scene points by the following equation:[36,37]

$$z_b = (2^N - 1) \cdot \left(a + \frac{b}{Z} \right), \tag{7}$$

where

$$a = \frac{z\,\mathrm{Far}}{z\,\mathrm{Far} - z\,\mathrm{Near}}, \tag{8}$$

$$b = \frac{z\,\mathrm{Far} \cdot z\,\mathrm{Near}}{z\,\mathrm{Near} - z\,\mathrm{Far}}, \tag{9}$$

and z Near, z Far are the depths of the near and far clipping planes correspondingly. These clipping planes define the scene depth range selected for rendering (see Fig. 4).[34]

Taking into account Eqs. (1) and (7), one can derive the following equation for disparity:

$$d = \frac{B \cdot f}{Z} = \frac{B \cdot f \cdot \left(\frac{z_b}{2^N - 1} - a \right)}{b}. \tag{10}$$

This value is defined as the subpixel disparity.

4.2 *Program for Generating Stereovision Sequences*

The motivation for writing the program was the need to develop a tool for verifying visual odometry egomotion estimation algorithms. The proposed program allows the user to define a static 3-D scene and the movement path of the camera. For the defined movement trajectory in the defined scene, the program generates: sequences of 6DoF egomotion parameters of the camera, corresponding stereovision images, and ground-truth disparity maps of the explored scene.

The scenes can be built from quadrangles and three types of 3-D solid objects: cuboids, spheres, and cylinders. An

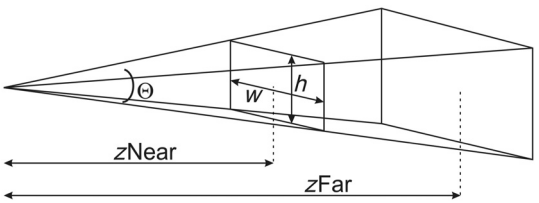


Fig. 4 A viewing frustum defining the field and depth of view for a perspective camera model.

object's parameters, i.e., size, texture, color as well as its location and orientation can be defined by the user.

The program outputs the following data:

- a sequence of left and right camera images in Windows bitmap format (bmp) with an option to corrupt the images with an additive Gaussian noise of user-specified standard deviation;
- a sequence of disparity maps for the left and right cameras with pixel or subpixel accuracy;
- a sequence of matrices containing disparity maps with 16-bit resolution;
- a sequence of segmented scene images.

In Fig. 5, a flow diagram explaining the generation of data sequences comprising camera motion vectors, stereovision images, depth, and segmentation maps is shown. First, the scene model is built as defined by the user in the stereovision egomotion sequence generator (SESGen) script. Then the stereovision camera is moved in that scene along the path specified by a sequence of camera motion vectors. For each new camera position, left and right camera images are captured and the corresponding ground-truth depth and segmentation maps are calculated and stored. The segmented scene images are derived as follows. For each of the defined scene objects, a disparity map is computed with the object of interest being removed. Simple comparison of the so obtained disparity map to a disparity map containing the scene object allows one to identify image regions corresponding to the scene object under question. Then the region corresponding to the rendered object is labeled with a unique ID number which can be further used for indexing a color from a predefined color palette [see an example of a segmented scene image in Fig. 6(b)]. It is worth noting that the disparity map is a reference map obtained from the depth buffer, so that the disparity for each point of the image is determined with subpixel accuracy. Note also that Eq. (2) includes translational and angular velocities, whereas OpenGL methods define frame-by-frame translational motions and rotations. In order to increase the precision of the camera movement, the frame-to-frame motion is subdivided into an arbitrarily selected number of in-between frames (e.g., $S = 256$). Code implementation of this procedure is explained in the SESGen user guide.[26] The program was written in C++ using OpenGL and is made available free of charge on the website.[26] Sample sequences with disparity maps can also be downloaded from the website.

4.3 *Scene and Motion Description Language*

In order to facilitate building arbitrary 3-D scenes and defining 6DoF camera motions, a simple script language is proposed. The scripts are stored as plain text, which allows for convenient editing and reusing of previously defined scripts in new projects. The custom built script language features keywords that define the following simple scene objects, i.e., QUAD—quadrangle, CUBOID—cuboid, SPHERE—sphere, and CYLINDER—cylinder. Each object keyword should be followed by a sequence of object properties. For the quadrangles, the user should take care of defining coordinates of the vertices to be coplanar. The keyword EGO is used for defining frame-to-frame egomotion parameters. Inserting comments into the script is possible by starting the script line with "//" (i.e., double slash). A graphical user interface of a program designed for scene and egomotion editing is shown in Fig. 7. The text window lists the script defining a 3-D scene shown in Fig. 6 and a sequence of translation and rotation egomotion parameters individually defined for consecutive frames.

5 Application of SESGen Software Tool for Verifying Egomotion Estimation Algorithms from Visual Odometry

In order to show the capabilities of the proposed SESGen tool, an example 3-D scene and stereovision system egomotion path were defined by the script listed in Fig. 7. These data served as ground-truth reference for verifying scene reconstruction and egomotion estimation algorithms as described in Sec. 3. The disparity maps were computed with pixel accuracy by applying a block matching method in which the SAD criterion was used.[11] Depth values were calculated from Eq. (1) for a predefined focal length of the cameras and the baseline of the stereovision system.

Estimation results of the egomotion parameters computed by means of the visual odometry technique defined in Sec. 3 are summarized in Table 1. The generated sequence consists of 40 stereovision images. The maximum absolute values for each motion vector component are: $U_{\max} = 0.030$, $V_{\max} = 0.039$, $Z_{\max} = 0.044$, $\alpha_{\max} = 0.8$, $\beta_{\max} = 0.8$, and $\gamma_{\max} = 0.5$. The root-mean-square errors (RMSEs) for each of the estimated motion components are the following: $\mathrm{RMSE}_X = 0.0049$, $\mathrm{RMSE}_Y = 0.0101$, $\mathrm{RMSE}_Z = 0.0062$,

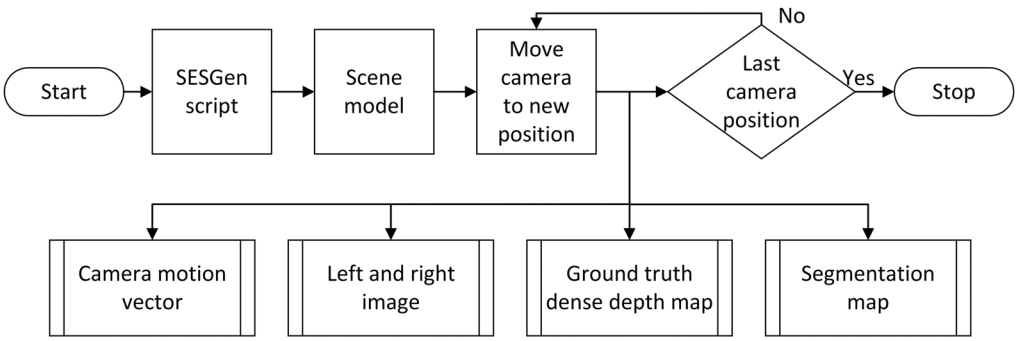


Fig. 5 Block diagram illustrating generation of motion vectors, images, depth, and segmentation maps in the stereovision egomotion sequence generator (SESGen) software tool.

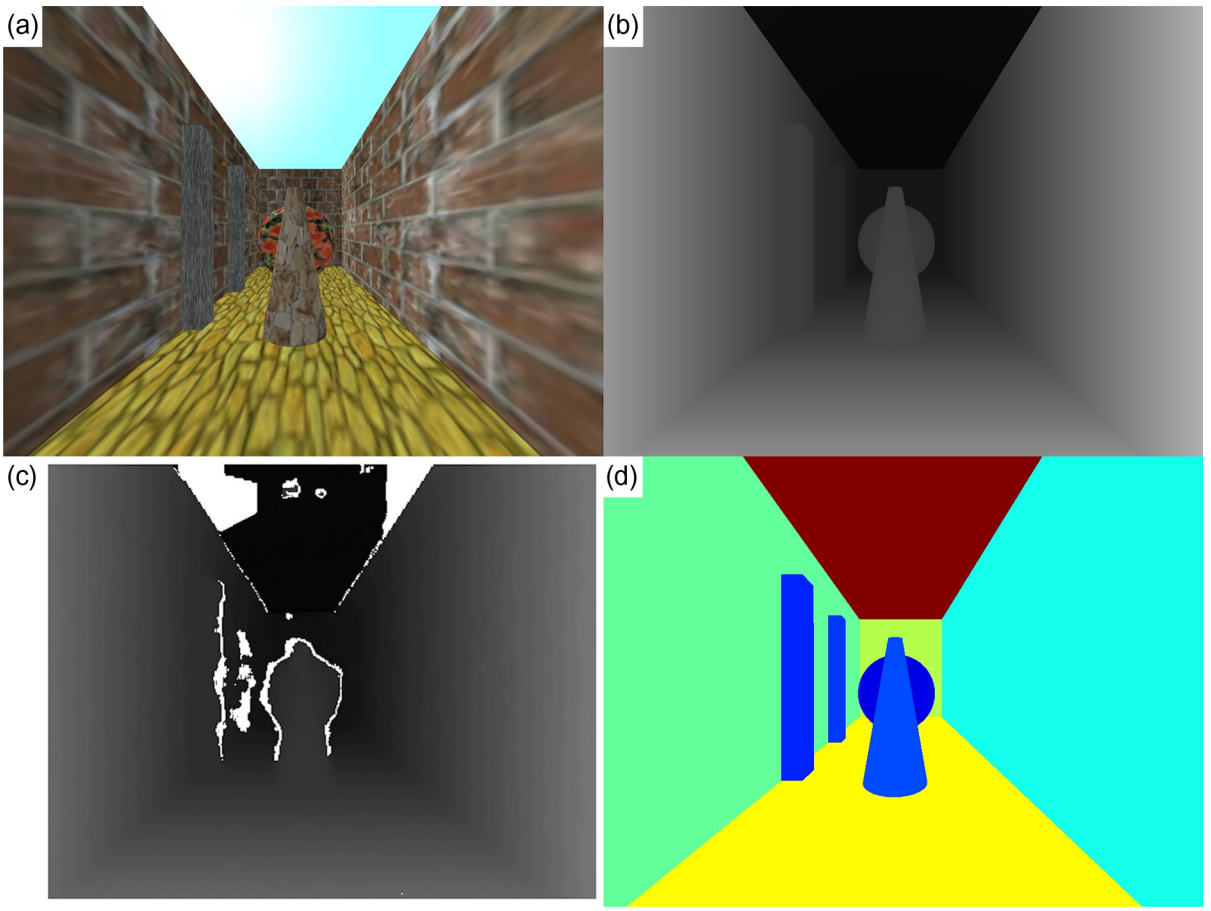


Fig. 6 A 3-D test scene defined by the script listed in Fig. 5: (a) 3-D test scene, (b) ground-truth disparity map computed with subpixel accuracy, (c) disparity map obtained using sum of absolute difference (SAD) (block matching) criterion, (d) scene segmented into 3-D objects and flat quadrangles positioned in 3-D space.

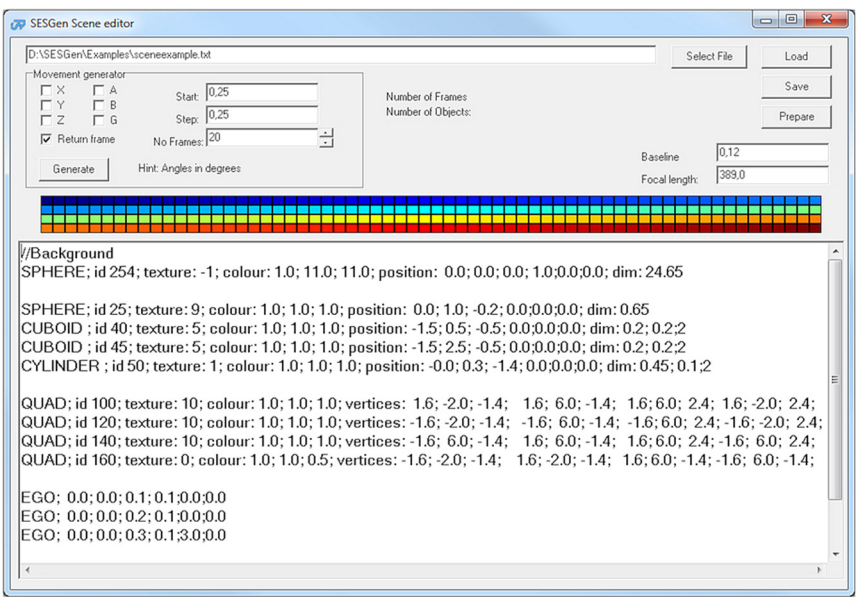


Fig. 7 View of the program window for defining and editing three-dimensional (3-D) scenes and ego-motion paths (see the SESGen User Guide[26] for a complete explanation of the script keywords and parameters).

Table 1 The ground-truth egomotion six degrees of freedom (6DoF) parameters and the estimated parameters obtained from the tested visual odometry technique (distances are given in arbitrary OpenGL units and angles are given in degrees).

	Ground-truth values						Estimated values					
	Translational movement			Rotational movement			Translational movement			Rotational movement		
Frame number	U	V	W	α	β	γ	U	V	W	α	β	γ
1	0.020	0.020	0.025	−0.400	0.325	0.200	0.013	0.009	0.019	−0.197	0.327	0.157
2	0.023	0.011	0.026	−0.340	0.326	0.210	0.013	0.004	0.021	−0.184	0.277	0.144
3	0.025	0.017	0.027	−0.310	0.327	0.100	0.012	0.007	0.028	−0.137	0.268	0.014
4	0.020	0.021	0.028	−0.720	0.328	0.000	0.012	0.005	0.033	−0.482	0.325	0.114
5	0.023	0.000	0.029	−0.600	0.329	0.000	0.015	0.004	0.025	−0.486	0.291	0,054
6	0.020	0.000	0.030	−0.700	0.530	0.500	0.009	0.011	0.037	−0.502	0.489	0.444

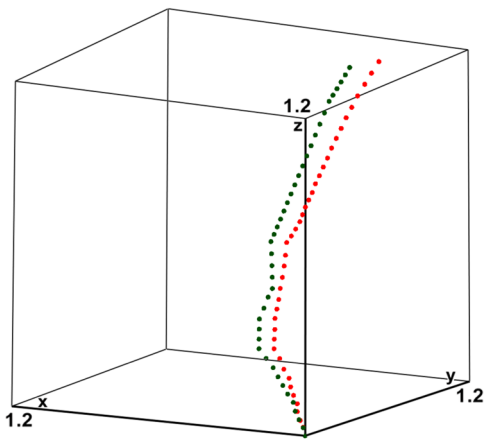
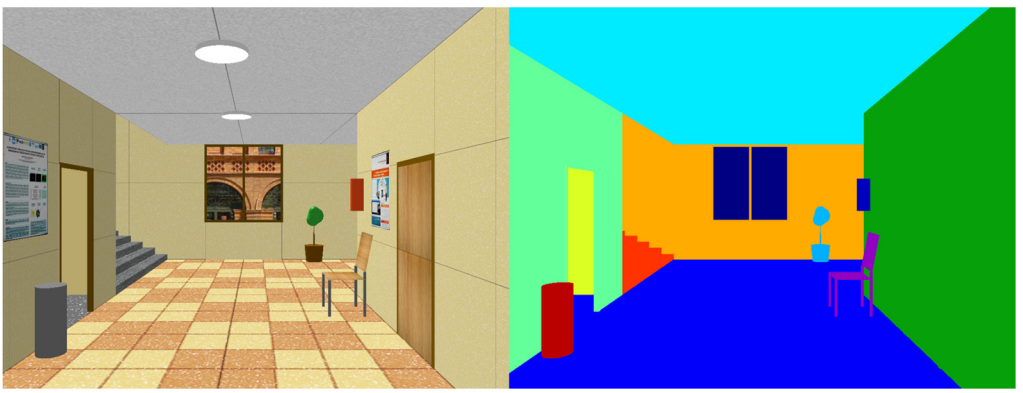


Fig. 8 Visualization of 3-D egomotion paths of the stereovision system: green dots (on the left) indicate a sequence of camera positions defined from the SESGen tool and the red dots (on the right) denote a sequence of camera positions computed from the tested visual odometry (VO) algorithm.

$\mathrm{RMSE}_\alpha = 0.101$, $\mathrm{RMSE}_\beta = 0.091$, and $\mathrm{RMSE}_\gamma = 0.057$, where the distance RMSEs are given in arbitrary OpenGL units and the angle RMSEs are given in degrees. The ground-truth values are obtained from the SESGen software tool. Note, however, that the table lists the egomotion parameters computed for six consecutive frames that were selected from a longer sequence of camera movements as shown in Fig. 8.

It was identified that the major contribution to egomotion estimation inaccuracies comes from two sources. First, there are errors in estimating motion vectors of the keypoints. Second, errors occur due to calculations of the disparity values with pixel accuracy resulting in inaccurate estimation of depth values for the keypoints. Plots of the defined motion path and the estimated paths computed from the VO algorithm are shown in Fig. 8. Note a successive diversion of path trajectories due to the incremental nature of the applied VO algorithm described in Sec. 3. Improvements on the results presented in Table 1 can be achieved by applying subpixel methods for computing the motion vectors of the keypoints in consecutive images.



(a) (b)

Fig. 9 Example of a realistic scene with its segmentation map: (a) the "Hall" test sequence, and (b) segmentation map for the "Hall" test sequence.

6 Summary

A software tool, named SESGen, for testing the performance of visual odometry algorithms was proposed. The program allows the user to define virtual 3-D static scenes and specify 6DoF motion paths of a camera (monocular or binocular) within a defined static scene. The SESGen outputs are: scene projection images, disparity maps (with pixel and subpixel accuracy), the sequence of camera motion vectors, and images with scene objects segmented out. SESGen uses the OpenGL depth-buffer (Z-buffer) to manage depth coordinates for the rendered 3-D scenes. A simple script language simplifies the task of defining 3-D scenes and motion paths that the user can apply for testing various VO techniques for unconstrained motion trajectories.

We hope that SESGen can serve as a useful tool for benchmarking different VO and image segmentation algorithms and can help in better identification of error sources in the tested algorithms. The tool can also be useful in verifying segmentation algorithms of user-defined 3-D scene images. Another foreseen application of the SESGen tool is to use it for validation of algorithms integrating stereovision sequences and signals from inertial sensors in egomotion estimation tasks.[38] This line of research has been initiated within an international project aimed at developing assistive devices aiding visually impaired people in independent mobility and travel. Compared to the existing databases of images, e.g., as reported in Ref. 1, our software tool enables the generation of much longer image sequences, with the corresponding ground-truth maps and the segmented images. It is possible to add user-defined textures of scene objects. Also, altering the parameters of the stereovision rig along with adding special effects such as distance fog has been made possible. Additionally, it is possible to corrupt the generated scene images with an additive Gaussian noise of user-defined standard deviation and verify noise robustness of potential visual odometry algorithms. An example of a realistic scene with its segmentation map generated with the use of our software is shown in Fig. 9. Test sequences and the scripts used for their generation are available from the webpage of the project. The authors would like to invite other users to contribute to a richer collection of 3-D scenes and motion paths for benchmarking.

Acknowledgments

This project has received funding from the European Unions Horizon 2020 research and innovation program under Grant No. 643636 "Sound of Vision."

References

1. D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE Robot. Autom. Mag.* **18**(4), 80–92 (2011).
2. D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry for ground vehicle applications," *J. Field Robot.* **23**, 2006 (2006).
3. D. Crandall et al., "SFM with MRFS: discrete-continuous optimization for large-scale structure from motion," *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(12), 2841–2853 (2013).
4. P. Skulimowski and P. Strumillo, "Refinement of depth from stereo camera ego-motion parameters," *Electron. Lett.* **44**, 729–730 (2008).
5. A. J. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *Proc. Ninth IEEE Int. Conf. on Computer Vision (ICCV 2003)*, Vol. 2, pp. 1403–1410, IEEE Computer Society, Washington, DC (2003).
6. D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proc. IEEE Comput. Soc. Conf. on Computer Vision and Pattern Recognition (CVPR 2004)*, Vol. 1, pp. 652–659 (2004).
7. A. Milella and R. Siegwart, "Stereo-based ego-motion estimation using pixel tracking and iterative closest point," in *Proc. IEEE Int. Conf. on Computer Vision Systems (ICVS 2006)*, pp. 21–21 (2006).
8. A. Fusiello, E. Trucco, and A. Verri, "A compact algorithm for rectification of stereo pairs," *Mach. Vis. Appl.* **12**, 16–22 (2000).
9. R. Kapoor and A. Dhamija, "Fast tracking algorithm using modified potential function," *IET Comput. Vis.* **6**(2), 111–120 (2012).
10. A. Ström and R. Forchheimer, "Low-complexity, high-speed, and high-dynamic range time-to-impact algorithm," *J. Electron. Imaging* **21**(4), 043025 (2012).
11. M. Brown, D. Burschka, and G. Hager, "Advances in computational stereo," *IEEE Trans. Pattern Anal. Mach. Intell.* **25**, 993–1008 (2003).
12. D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.* **60**, 91–110 (2004).
13. S. Roumeliotis, A. Johnson, and J. Montgomery, "Augmenting inertial navigation with image-based motion estimation," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA 2002)*, 4, pp. 4326–4333 (2002).
14. R. G. Garcia-Garcia et al., "3D visual odometry for GPS navigation assistance," in *Proc. 2007 IEEE Intelligent Vehicles Symp.*, pp. 444–449 (2007).
15. D. Scaramuzza, "1-Point-RANSAC structure from motion for vehicle-mounted cameras by exploiting nonholonomic constraints," *Int. J. Comput. Vis.* **95**(1), 74–85 (2011).
16. S.-H. Lee, "Real-time camera tracking using a particle filter combined with unscented Kalman filters," *J. Electron. Imaging* **23**(1), 013029 (2014).
17. Y. Tan, S. Kulkarni, and P. Ramadge, "A new method for camera motion parameter estimation," in *Proc. Int. Conf. on Image Processing*, Vol. 1, pp. 406–409 (1995).
18. E. T. Kim and H. Kim, "Recursive total least squares algorithm for 3-D camera motion estimation from image sequences," in *Proc. Int. Conf. on Image Processing (ICIP 1998)*, Vol. 1, pp. 913–917 (1998).
19. C. Garcia and G. Tziritas, "3D translational motion estimation from 2D displacements," in *Proc. 2001 Int. Conf. on Image Processing*, Vol. 2, pp. 945–948 (2001).
20. P. Corke, D. Strelow, and S. Singh, "Omnidirectional visual odometry for a planetary rover," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS 2004)*, Vol. 4, pp. 4007–4012 (2004).
21. C. Mei et al., "RSLAM: a system for large-scale mapping in constant-time using stereo," *Int. J. Comput. Vis.* **94**(2), 198–214 (2011).
22. W. van der Mark et al., "Vehicle ego-motion estimation with geometric algebra," in *Proc. IEEE Intelligent Vehicle Symp.*, Vol. 1, pp. 58–63 (2002).
23. A. Bak, S. Bouchafa, and D. Aubert, "Dynamic objects detection through visual odometry and stereo-vision: a study of inaccuracy and improvement sources," *Mach. Vis. Appl.* **25**(3), 681–697 (2014).
24. Middlebury Stereo Vision, http://vision.middlebury.edu/stereo/ (April 2015).
25. H. Hirschmuller and D. Scharstein, "Evaluation of cost functions for stereo matching," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2007)*, pp. 1–8 (2007).
26. P. Skulimowski and P. Strumillo, *SESGen User Guide*, Lodz University of Technology, http://stereo.naviton.pl (April 2015).
27. B. Cyganek and P. Siebert, *An Introduction to 3D Computer Vision Techniques and Algorithms*, John Wiley & Sons, Ltd., Chichester, West Sussex (2009).
28. P. Strumillo et al., "Implementation of stereo matching algorithms on graphics processing units," in *Image Processing & Communications Challenges, Academy Publishing House EXIT*, A. Z. R.S. Choras, Ed., pp. 286–293, Academy Publishing House EXIT, Warsaw, Poland (2009).
29. R. Guissin and S. Ullman, "Direct computation of the focus of expansion from velocity field measurements," in *Proc. IEEE Workshop on Visual Motion*, pp. 146–155 (1991).
30. A. Bruss and B. Horn, "Passive navigation," *Comput. Vis. Graph. Image Process.* **21**, 3–20 (1983).
31. K. Matusiak, P. Skulimowski, and P. Strumillo, "A mobile phone application for recognizing objects as a personal aid for the visually impaired users," in *Proc. Human-Computer Systems Interaction: Backgrounds and Applications 3, Advances in Intelligent Systems and Computing*, Vol. 300, pp. 201–212 (2013).
32. J. Shi and C. Tomasi, "Good features to track," in *Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR 1994)*, pp. 593–600 (1994).
33. G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*, O'Reilly, Cambridge, Massachusetts (2008).
34. D. Shreiner et al., *The OpenGL Programming Guide: The Redbook*, Addison-Wesley Professional, Reading, Massachusetts (2005).
35. R. J. Wright et al., *OpenGL SuperBible: Comprehensive Tutorial and Reference*, 5th ed., Pearson Education, Inc., Upper Saddle River, New Jersey (2011).
36. D. Shreiner, *OpenGL Reference Manual: The Official Reference Document to OpenGL, Version 1.2*, 3rd ed., Addison-Wesley Professional, Upper Saddle River, New Jersey (1999).

37. S. Beaker, "Learning to love your Z-buffer," 2012, http://www.sjbaker.org/steve/omniv/love_your_z_buffer.html (April 2015).
38. P. Pelczynski, B. Ostrowski, and D. Rzeszotarski, "Motion vector estimation of a stereovision camera with inertial sensors," *Metrol. Meas. Syst.* **19**(1), 141–150 (2012).

Piotr Skulimowski is an assistant professor at the Lodz University of Technology. He received his MS degree in electronics and telecommunications and his PhD in computer science from the Lodz University of Technology in 2003 and 2009, respectively. He is the author of more than 40 papers. His current research interests include stereo vision, image processing on mobile devices, and passive navigation. He is a member of the Polish Information Processing Society.

Pawel Strumillo received his MSc degree in electrical engineering from the Lodz University of Technology (TUL), Poland, in 1983 and his PhD in technical sciences from the University of Strathclyde in 1993. Currently, he is the head of the Medical Electronics Division. His recent activities concentrate on running projects dealing with development of systems aiding the visually impaired in independent mobility and human–computer interfaces. He is a senior member of IEEE.