SESGen
User Guide

Piotr Skulimowski and Pawel Strumillo

January 30, 2012

pskul@p.lodz.pl

Contents

1 Introduction

SESGen is the software tool designed for generating stereoscopic image sequences of synthetic 3D scenes containing surfaces and textured objects. Image sequences are recorded by a freely moving stereo rig along the motion path defined by the User. For each of the generated frames the left and right stereovision images, the corresponding ground truth disparity map and a segmentation map of scene image are producedTo facilitate scene generation task a simple script language and its interpreter are of help.

The reference camera is the left camera. Images for the right camera are generated by translating virtual camera by a defined in program base value in x direction.

2 Scene and motion description language

2.1 Objects

Each object in the scene is described by a single line described by a keyword and a set of parameters, for example:

```
CYLINDER ; id 50; texture: 1; colour: 1.0; 1.0; 1.0; position: -0.0; 0.3; -1.4; 0.0;0.0;0.0; dim: 0.45; 0.1;2
```

Parameter names and their values are separated by semicolon. Parameters' names can be ommited, so you can rewrite the command defining the CYLINDER object to the following script:

```
CYLINDER;50;1;1.0;1.0;1.0;-0.0;0.3;-1.4;0.0;0.0;0.0;0.45;0.1;2
```

- id is the id of an object in the segmented image. Hence, you can build complicated objects using primitives defined in this application. To choose proper id you can choose it from the color boxes above the editor window,
- texture: id of the texture available in this application, you can also use your own textures (just replace texture files with your own file in the same format and resolution), if you do not want to define texture for an object just put negative value, for example -1
- three values of color components, texture pixels can by multiplied by this values, if you do not want to change texture colour use unit values as in the given example
- six values, i.e. three translation parameters $[tr_x, tr_y, tr_z]$ and three rotation parameters $[rot_a, rot_b, rot_c]$, define 6DoF movement of each primitive object (except QUAD):

  ```
  glTranslated(tr_x,tr_y,tr_z);
  glRotated(rot_a, 1.0, 0.0, 0.0);
  glRotated(rot_b, 0.0, 1.0, 0.0);
  glRotated(rot_g, 0.0, 0.0, 1.0);
  ```

- primitive specific parameters.

2.1.1 Primitive specific parameters

SPHERE

Draws a sphere of the given radius centered around the origin, parameters:

- radius

CYLINDER

Draws a cylinder oriented along the z-axis. The base of the cylinder is placed at z = 0, and the top at z = height.

- baseRadius – radius of the cylinder at z = 0,
- topRadius – radius of the cylinder at z = height,
- height – height of the cylinder.

CUBOID

Draws a cuboid oriented along the z-axis, the base of the cuboid is placed at z = -sz/2, and the top at sz/2, opposite vertices of the cuboid are at (-sx/2,-sy/2,-sz/2) and (sx/2,sy/2,sz/2)

- sx - cuboid first dimension
- sy - cuboid second dimension
- sz - cuboid third dimension

Note: SPHERE, CYLINDER and CUBOID can be rotated and translated.

2.1.2 quadrangle - QUAD

Quadrangles cannot be rotated and translarted, they are defined by their vertices in the scene. Moreover, the user additionally should take care of defining coordinates of the vertices to be coplanar (but they don't have to be coplanar...). QAUD needs 12 parameters (3 parameters per vertex).

2.2 Camera movement definition

The keyword EGO is used for defining frame-to-frame egomotion parameters. 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. $S = 256$) according to the following OpenGL program code:

```
for (int j=0;j<S;j++)
{
        glTranslated(tr_x/S, tr_y/S, tr_z/S);
        glRotated(rot_a/S, 1.0, 0.0, 0.0);
        glRotated(rot_b/S, 0.0, 1.0, 0.0);
        glRotated(rot_g/S, 0.0, 0.0, 1.0);
}
```

where: $[tr_x, tr_y, tr_z]$ is the translation vector and $[rot_a, rot_b, rot_g]$ is the rotation vector defining camera motion between two consecutive frames.

To generate a sequence of EGO descriptions simple but useful movement generator is included in the program. Using checkboxes you can choose, which ego-motion components will be changed from defined start value with defined step, for example for the parameters set as shown in figure 1.

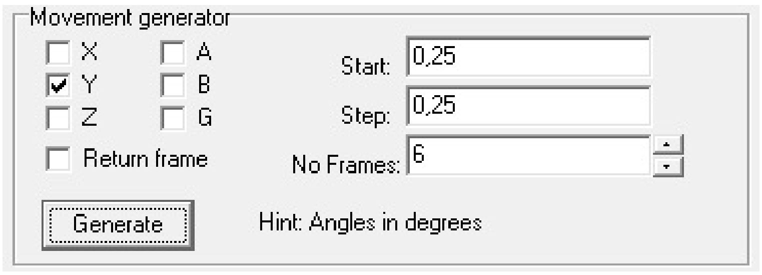

Movement generator
X [ ] A [ ]
Y [✓] B [ ]
Z [ ] G [ ]
Return frame [ ]
Start: 0,25
Step: 0,25
No Frames: 6
Generate
Hint: Angles in degrees

Figure 1: Movement Generator

the following listing will be automatically generated:

```
EGO; 0,000;0,250;0,000;0,000;0,000;0,000;
EGO; 0,000;0,500;0,000;0,000;0,000;0,000;
EGO; 0,000;0,750;0,000;0,000;0,000;0,000;
EGO; 0,000;1,000;0,000;0,000;0,000;0,000;
EGO; 0,000;1,250;0,000;0,000;0,000;0,000;
EGO; 0,000;1,500;0,000;0,000;0,000;0,000;
```

If 'Return frame' is checked, it allows to generate a set of forward/backward movement, for example if you want to test your algorithms for different values of parameters but from the same start frame:

```
EGO; 0,000; 0,250;0,000;0,000;0,000;0,000;
EGO; 0,000;-0,250;0,000;0,000;0,000;0,000;
EGO; 0,000; 0,500;0,000;0,000;0,000;0,000;
EGO; 0,000;-0,500;0,000;0,000;0,000;0,000;
EGO; 0,000; 0,750;0,000;0,000;0,000;0,000;
EGO; 0,000;-0,750;0,000;0,000;0,000;0,000;
```

3 Images' format

SESGen generates a set of output files containing scene images, disparity maps and segmentation maps. Each file name contains frame number stored in 4 digits.

3.0.1 bindisp_XXXX.dat and lbindisp_XXXX.dat

Ground truth disparity map for the right and left images stored in plain text format, first two lines contain image resolution, next lines contain disparity values, corresponding to a sequence of pixels starting from top left image pixel and scanned along consecutive image rows, for example:

```
1024
768
35.687587134066753
35.6178177919107384
35.5480484497547167
35.4782791075987021
35.408504282586712
```

3.0.2 disp_XXXX.bmp, ldisp_XXXX.bmp

Disparity map for the right and left image in the bmp format with subpixel accuracy. Images are in grayscale, $R = G = B = disp * 255/70$ where $disp$ is float number.

3.0.3 disp_p_XXXX.bmp, ldisp_p_XXXX.bmp

Disparity map for the right and left image in the bmp format with pixel accuracy. Images are in grayscale, $R = G = B = disp * 255/70$ where $disp$ is integer value.

3.0.4 right_XXXX.bmp, left_XXXX.bmp

The right and left colour image in the bmp format.

3.0.5 seg_r_XXXX.bmp, seg_l_XXXX.bmp

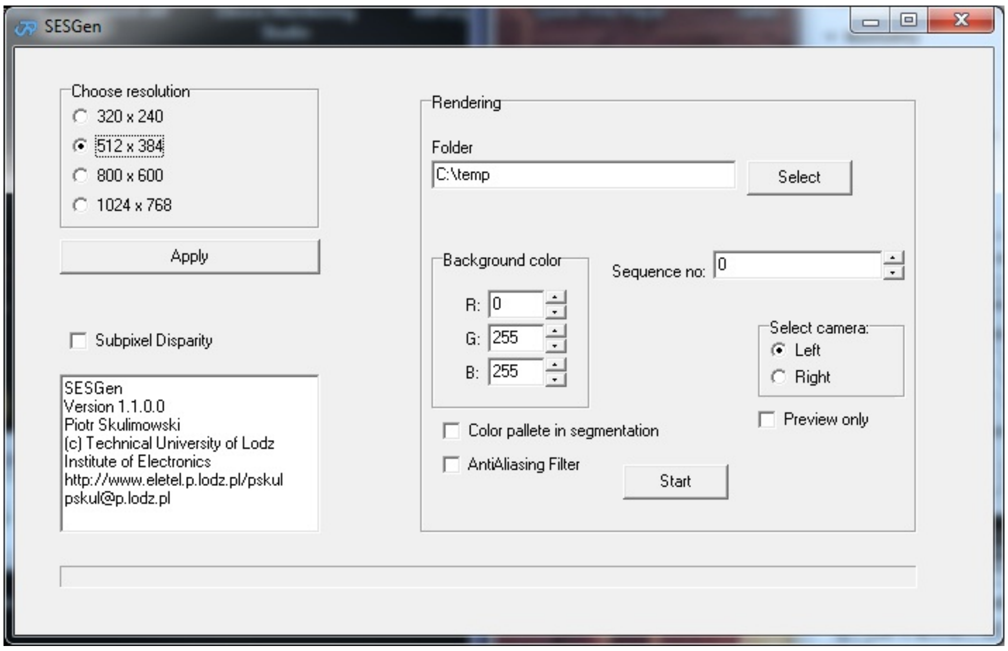
Segmentation maps for the right and left image in the bmp format. If grayscale mode is selected $R = G = B = id$, where id – identification number of the object. If pseudocolor mode is selected pixel value depends on pseudocolor map values for identification number of the object.

4 Other settings

4.1 Main window

SESGen's main window is shown in figure 2

Using the main window you can select resolution of the images and specify a folder where images and other data files will be stored along with the sequence number. The program automatically creates a subfolder named with the selected sequence number. Note, however, that if the earlier defined sequence number is displayed it will be overwritten by a new sequence! You can select also which camera images (left or right) will be generated and you can set scene background colour. If you choose *Preview only*, no



SESGen

Choose resolution
320 x 240
[✓ 512 x 384]
800 x 600
1024 x 768

Apply

Subpixel Disparity

SESGen
Version 1.1.0.0
Piotr Skulimowski
(c) Technical University of Lodz
Institute of Electronics
http://www.eletel.p.lodz.pl/pskul
pskul@p.lodz.pl

Rendering
Folder: C:\temp
Select

Background color
R: 0
G: 255
B: 255

Sequence no: 0

Select camera:
[✓ Left]
Right

Preview only

Color pallete in segmentation

AntiAliasing Filter

Start

Figure 2: SESGen Main Window

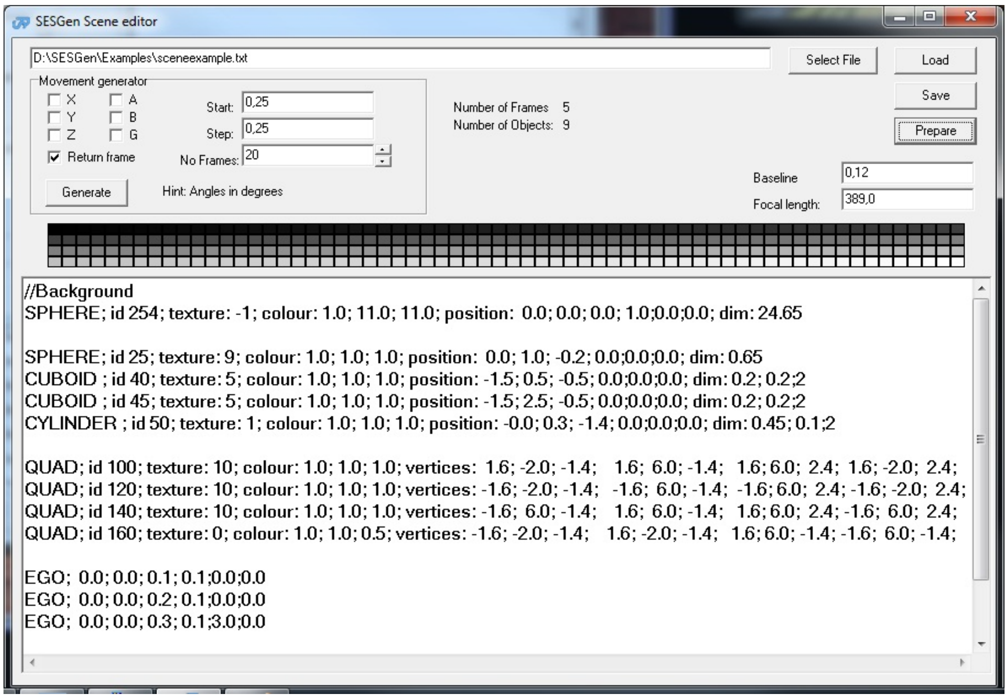
files will be saved to the folder. If *Color pallete in segmentation* is checked, a pseudo-color pallete will be used instead of grayscale in the segmentation map. You can also turn on the antialiasing filter. Press *Start* to start generating sequence (progress bar will be displayed).

4.2 Editor window

SESGen's editor window is shown in figure 3

Using the editor window you can load and save scripts that define scenes. If you press *Prepare* button a scene defined in the editor window (not in file) will be generated. Information about the number of frames and the number of scene objects will be displayed. In the editing windows *Baseline Focal length* you can define baseline of the stereovison system (i.e. the distance between cameras in opengl distance unit) and the focal length of the camera correspondingly. Note, hovever, that by changing image resolution the selection the focal length of the camera (given in pixels) will be also changed. This is because angle view of the camera changes accordingly.

The current colour pallete is displayed directly above the editing script field. By selecting a colour (or greyscale) rectangle colour (or grayscale) identification number will be displayed. This feature helps in defining scenes.



D:\SESGen\Examples\sceneexample.txt

Movement generator

[ ] X [ ] A
[ ] Y [ ] B
[ ] Z [ ] G
[✓] Return frame

Start: 0,25
Step: 0,25
No Frames: 20

Number of Frames 5
Number of Objects: 9

Baseline 0,12
Focal length: 389,0

```
//Background
SPHERE; id 254; texture: -1; colour: 1.0; 11.0; 11.0; position:  0.0; 0.0; 0.0; 1.0;0.0;0.0; dim: 24.65

SPHERE; id 25; texture: 9; colour: 1.0; 1.0; 1.0; position:  0.0; 1.0; -0.2; 0.0;0.0;0.0; dim: 0.65
CUBOID ; id 40; texture: 5; colour: 1.0; 1.0; 1.0; position: -1.5; 0.5; -0.5; 0.0;0.0;0.0; dim: 0.2; 0.2;2
CUBOID ; id 45; texture: 5; colour: 1.0; 1.0; 1.0; position: -1.5; 2.5; -0.5; 0.0;0.0;0.0; dim: 0.2; 0.2;2
CYLINDER ; id 50; texture: 1; colour: 1.0; 1.0; 1.0; position: -0.0; 0.3; -1.4; 0.0;0.0;0.0; dim: 0.45; 0.1;2

QUAD; id 100; texture: 10; colour: 1.0; 1.0; 1.0; vertices:  1.6; -2.0; -1.4;   1.6;  6.0; -1.4;   1.6; 6.0;  2.4;  1.6; -2.0;  2.4;
QUAD; id 120; texture: 10; colour: 1.0; 1.0; 1.0; vertices: -1.6; -2.0; -1.4;   -1.6;  6.0; -1.4;  -1.6; 6.0;  2.4; -1.6; -2.0;  2.4;
QUAD; id 140; texture: 10; colour: 1.0; 1.0; 1.0; vertices: -1.6;  6.0; -1.4;   1.6;  6.0; -1.4;   1.6; 6.0;  2.4; -1.6;  6.0;  2.4;
QUAD; id 160; texture: 0; colour: 1.0; 1.0; 0.5; vertices: -1.6; -2.0; -1.4;   1.6; -2.0; -1.4;   1.6; 6.0; -1.4; -1.6;  6.0; -1.4;

EGO;  0.0; 0.0; 0.1; 0.1;0.0;0.0
EGO;  0.0; 0.0; 0.2; 0.1;0.0;0.0
EGO;  0.0; 0.0; 0.3; 0.1;3.0;0.0
```

Figure 3: Editor Window