Landscape Generator

Michael Kainz
University of Applied Sciences Augsburg
WS 20/21

Interaction Engineering
by Prof. Dr. Michael Kipp

Abstract

The Landscape Generator is a tool to create levels or landscapes randomly by rolling dice or by placing them one by one. The dice get recorded by a webcam and a python program then reads the pips on the top side of the dice. Every number of pips is assigned to a model of a specified set. The position and the model get transferred to the a website, which shows the scenery in 3D. The user can move the camera, look at the creation and adjust the scene by moving the dice by hand.

1 Motivation

The idea originated from the use of the position and rotation of fridge magnets to create a smart grocery list. With the help of Prof. Kipp the surface transisioned from a vertical to a horizontical plane. Instead of magnets, which only can show one side, dice are used to represent objects.

Due to the random nature of a die roll many dice can be used to generate a scene with randomly placed models. The user still has control over the scenery as the dice can be rerolled, placed in another position or rolled over to another side. This way the scene can be adapted to the users needs.

There are also different sets of models implemented. At the current stage these are "Forest", "Farm" and "Monsters". Each set uses 6 different models to create a scene. Other sets can get added to display literally anything.

2 Concept

In Figure 1 the setup is shown. The camera sits on an arm over a marked zone where the dice roll takes place. The user can roll the dice and can interact with them and the corresponding models by rotating it or moving it.

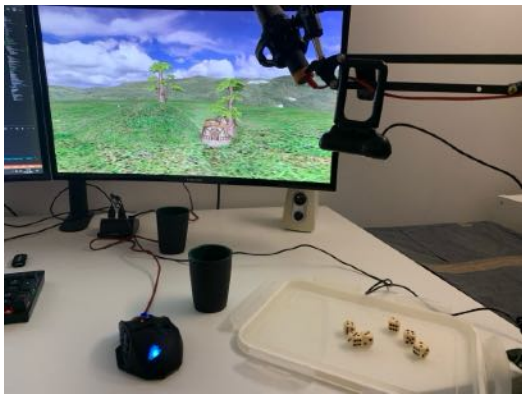
Figure 1: Setup.

3 Implementation

The camera gets read by a Python program powered by OpenCV. The captured frame gets converted to grayscale and thresholded so only the pips of the dice can be seen. The SimpleBlobDetector recognizes the pips when the area of a blob is between 140 and 260 pixels. Also circularity and inertia are used with values of 0.6 and 0.55 respectively. They then get clustered by range, so when one pip is in a range of 65 pixels to another it counts as "one" die. This also happens when two dice are placed near each other. Then the pips get added to represent hills in the scene. The higher the number of pips, the higher the hill. The detected dice then get converted to a JSON format with attributes like the number of pips, the coordinates and the rotation angle. This JSON gets send to the Python server via a WebSocket.
The server hosts a site running JavaScript with the Three.js library. The library can produce 3D graphics in a browser window. When starting the site loads the objects in the set and hides them to reduce loading time while operating. When a message from the Python program gets received the corresponding models only need to get copied and put on their respective place. Should this object miss in a message, it gets deleted. The pips only get detected,

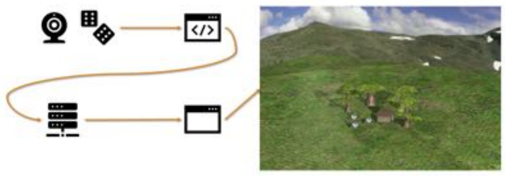

Figure 2: Toolchain.

not tracked, so every movement of a die is like a new one spawning.

4 Evaluation

4.1 Study

The project was evaluated with a group of 6 people, ranging from 21 to 45 years. Every participant had 3 tasks to solve.

The first task was to roll 12 dice in the three different sets. This showed how the generator works and made an first impression.

The second task asked the participants to get creative. Each one could choose a set and create a scene in 10 minutes how they liked it.

The last task was to recreate a pre-built scene. The participants knew by now how the system works and could apply it to rebuild that scenery.

The tasks were all solved successfully in time. Some of the created scenes can be viewed in the presentation. Some participants had trouble with the use of the camera. Others had a background in video games and tools and could use it right away. In conclusion the participants had fun testing the project and expressed a lot of helpful feedback.

4.2 Feedback

4.2.1 Pro

- It's fun to recreate
- Learn how hills get larger or smaller and in general which pips are which model
- Nice to be creative
- You can design your own game and influence it
- Forest theme nice, easy to understand
- Monster theme good
- Themes in general nice
- Idea and possibilities interesting
- Implementation well done, handling understandable
- Camera handling good
- Rotation works
- Intuitive

4.2.2 Contra

- Rotation of objects
- Area could be larger
- Recognition takes a long time
- Farm: barn too big, wind towers not as visible as animals
- Prevent "looking under the ground" or allow by button
- Additional 2D view helpful
- Clipping
- Button for standard view
- Sometimes hills "swallow" objects
- More terrain like water would be cool
- Detection is difficult with bad light

4.2.3 Ideas

- Use for children and the disabled to train coordination and recognize relationships (no spikes, larger dice not swallowable)
- Moving objects on VR goggles and through scenery
- Random scenery for first person shooters
- Random tracks for racing games
- Recreating accident scenes in court.

4.3 Lessons learned

The project taught some valuable lessons.

The work to recognize the dice and pips was very interesting. For the same problem there are a bunch of solutions. At first the whole die would get detected and in this subpicture the pips but that lead to much trouble. This way produced too many false results. The project from Kishaan (`https://github.com/Kishaan/Dice-Detection`) helped to detect the pips and cluster them to dice.

The frontend opened up a whole new world. Due to no experience with 3D libraries it was rather difficult to learn how the functions work. Fortunately Windows 10 comes with a 3D object library so there was no need to create the models by myself.

The case study went not exactly like planned. It is hard to teach people the controls and how the dice interact with the picture in the browser. When one works on a topic a long time it just feels natural. A precise documentation helps participants to get the gist of the project much faster.

There are still technical problems. Detecting and getting the right angle of rotation for example. 2s, 3s and 6s can only show up to 180°, 4s and 5s even only 90°, while 1s show no sign of rotation. This could get fixed by another pip on the die or by calculating the delta of the angle between two states.

5 Conclusion

The prototype could get extended by many features. Fixing the rotation issue, adding new sets and making the modifiable area larger would be on top of the to-do list.

Participants proposed to add sets like courses for racing games or settings for ego-shooters. An idea would be to roll a course and the pieces get connected automatically. This could replace conventional level editors.

Also the idea to use the generator for educational purposes was proposed. Kids and handicapped person could learn to make connections between real life and virtual objects.