

Tutorial 3 - Rendering animations with PBRT

Computer Graphics

Kartic Subr and Martin Asenov
October 4, 2019

You are working for Volkswagen and have been given the task of creating a video of the iconic VM bus, representing the company's history. You have been provided a 3D shape and its associated texture (<https://www.artec3d.com/3d-models/vw-bus>). Write a C++ program that programatically reads parameters from a file, renders an image using those parameters using a PBRT template file and finally concatenates all images in a video as seen in fig.1.

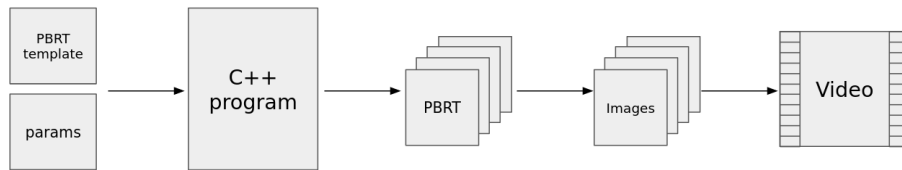


Figure 1: Generating video frames from PBRT template

Part 1: Implement the C++ program

You have been provided `animation.cpp` which contains most of the code to get you started. Implement the `populate_pbrt_file` function. The function should read the `param.pbrt` file which will include one `{{param1}}` which needs to be replaced with an appropriate value, provided by `replace`. You can compile `animation.cpp` on a DICE machine using the command:

```
g++ animation.cpp `pkg-config --cflags --libs opencv` -std=c++11 -o animation
```

In addition to implementing the `populate_pbrt_file` function in the supplied code, you will also need the texture and shape files for the VM bus from the link above.

When ready, execute the program and you should get a video similar to the one in fig. 2



Figure 2: Example video frames rendered with PBRT

Part 2: *Explore different video rendering options*

Start by displaying the car in the middle of the scene and rotate in around the center as seen in fig.2. Of course, this wouldn't make the most exciting car promotional video.. What additional changes you can implement in order to make a more appealing advertisement?

1. Camera position - rotate the camera around the car to show interesting details
<https://pbrt.org/fileformat-v3.html#cameras>
2. Lights - dynamically change the lights in different parts of the video - start with low light, and then alternate stronger and weaker light?
<https://pbrt.org/fileformat-v3.html#lights>
3. Car position - move the car while the camera is following it

Explore different equations to define the movement of different objects (e.g. camera). You can start simple, by using a circle equation for example - <https://www.mathopenref.com/coordparamcircle.html>. Then you can explore other parameteric equations of different curves, such as an eight curve (<http://mathworld.wolfram.com/EightCurve.html>) to use for more dramatic camera positions/change the source of the light.