HW/SW Codesign [LU]

Main Task - Ray Tracing

Florian Huemer, Dylan Baumann, Andreas Steininger
{fhuemer, dbaumann, steininger}@ecs.tuwien.ac.at
Department of Computer Engineering
TU Wien

Vienna, November 8, 2023

1 Assignment

The goal of the main task is to implement a ray tracer to render a simple 3D scene (in almost realtime, depending on the actual settings) and display the resulting image using the VGA port of the DE2-115 FPGA board. All arithmetic operations should be implemented with the Q16.16 fixed point format. You don't have to take care of overflows, the test data we provide won't produce values that are out of the range of this data type. Figure 1.1 shows an example scene produced by the used ray tracing algorithm[1].

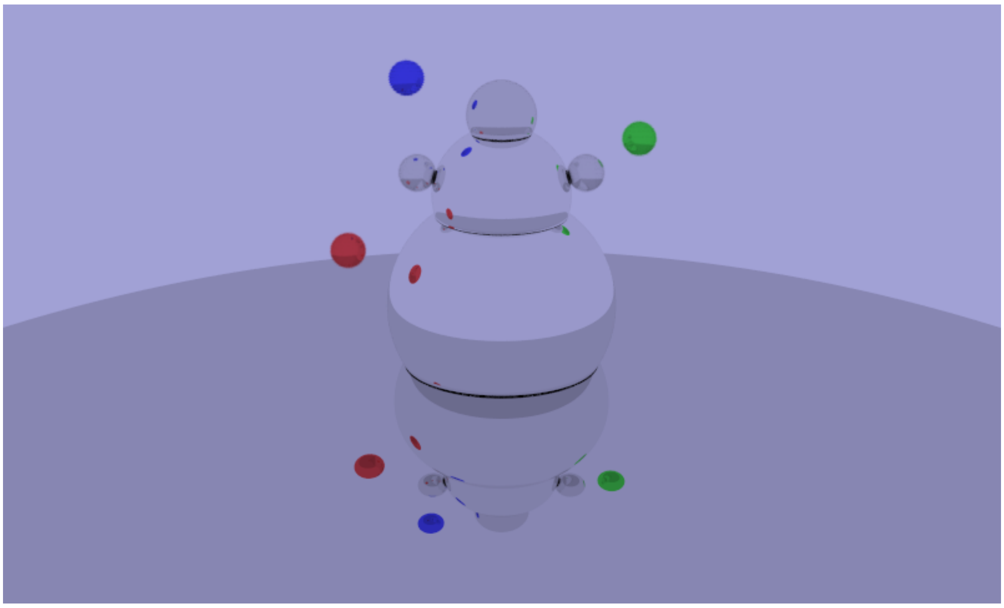

Figure 1.1: Example scene (snowman with decorative balls, appropriate for the time of year)

The software implementation is based on the ray tracer developed by Peter Shirley. The original book and the corresponding source code can be found on github [1]. For more details about the ray tracing algorithm and the provided template please refer to the recording of Main Task discussion provided in TUWEL.

2 Submission and Grading

The goal is to render the two test scenes (`basic` and `snowman`) included in the template design with a certain number of frames per second. You may use all resources on the board and the FPGA without restrictions.

The task is divided into two parts, i.e., the midterm presentation and the final submission.

2.1 Midterm presentations:

In the first part we ask you to profile the provided code (e.g., gprof, gcov) to gain a good understanding of the computational effort required for each part of the calculation. Based on these insights, you are then supposed to develop an abstract implementation strategy for your solution. You have to prepare a presentation containing your profiling insights together with your overall concept and present that to the remaining groups and the teaching staff. See Section 3 for further details on what we expect from this presentation.

As a group (one group member is sufficient) upload your presentation slides before the day of the presentation. Note that it is not required nor expected that you show a working HW/SW implementation at this point in time!

[1]However, the values of the render parameters used here are far beyond the range that you have to support with your solution. Nevertheless it shows what images can be created with relatively simple means.

You can get 35 points for this presentation. You will be evaluated based on the results you gathered, the conclusions you drew from them (15 points), the quality and feasibility of your concept (15 points), your expertise during the discussion and last but not least the quality of your presentation (5 points).

Final submission: Submit your solution of the main task using TUWEL (again the day before the presentations) and demonstrate in a final presentation. Prepare a few slides summarizing your solution, especially in comparison to the concept presented at the midterm presentation (What changed? What did not work as expected?). It would also be nice, if you could include a "lessons learned" section. For the submission create an archive using the `submission` target of the provided makefile. Remove all temporary folders and files. Make sure that the submission compiles and run in the TILab or the VM. Also go through your (Quartus) warnings and make sure that your submission is in accordance with the coding guidelines!

The grading will be based on the achieved performance, measured in frames per second (FPS) the quality of your design and the concrete solution as well as your expertise in the final presentation. Keep in mind that individual group members may be graded differently.

For the performance evaluation, your solution will be used to render the two provided test scenes (`basic` and `snowman`). Let f_b and f_s denote the FPS your solution achieves for the `basic` and the `snowman` scene, respectively. You will then be awarded

$$P = P_{\text{basic}}(f_b) + P_{\text{snowman}}(f_s)$$

$$P_{\text{basic}}(x) = \begin{cases} x & x < 5 \\ 25 - \frac{100}{x} & \text{otherwise} \end{cases}$$

$$P_{\text{snowman}}(x) = \begin{cases} 10 * x & x < 0.5 \\ 25 - \frac{25}{2*x} & \text{otherwise} \end{cases}$$

points. The target value for the `basic` scene is 10 FPS, while the `snowman` scene should yield 1.25 FPS (i.e., $P_{\text{basic}}(10) = P_{\text{snowman}}(1.25) = 15$). Hence, if you achieve these values the equation yields 30 points. For higher performance additional bonus are awarded.

These points will then be added to the points achieved considering the remaining evaluation criteria, where at most 20 points will be reached. Hence, overall 50 points are achievable for the final submission (excluding the bonus points).

3 Midterm Guidelines

We expect the following things of you and your presentation:

- A good understanding of the used algorithm (i.e., what is calculated where, what is the meaning of intermediate results, etc.)
- A good understanding of the complexity of the involved (mathematical) operations as well as their rates of execution (i.e., profiling results and their interpretation).
- A detailed and well-justified concept and draft for a system architecture.

You should be able to defend your design decisions and explain why you think that your proposed architecture will be able to reach the required benchmark goal.

3.1 Concept

Use the following questions and keywords as a guideline to explain your architecture/concept:

- What is the "main" processing unit of your design? Is it

  - Programmable: small specialized processor, large high-performance processor, SIMD, multi-threading, superscalar, VLIW, caches, (application specific) ISA ...
  - Fixed function: pipeline depth and processing stages, forwarding (data dependencies), data interleaving, resource sharing ...
- How many processing units are there?
- Which component processes which data at what point in time?
- What data is stored where and when (Flip-flops (Pipeline-Stages), block-RAM (On-Chip), SRAM, DRAM)?
- How do the individual function units communicate with each other?
  - (Avalon) Memory-Mapped Interface
  - (Avalon) Streaming Interface
  - Custom Instruction Interface
  - Full custom interfaces
- What is your desired clock speed?
- What is the role/purpose of the Nios processor?
- Where do you expect bottlenecks in your design?
- What will be the limiting FPGA resource?
- How long does the computation of a single ray/pixel take?
- How many rays/pixels can be processed at the same time?

We are aware that the numbers you will be presenting are based on estimations and are of course subject to change as the design evolves and is finalized. However, they should be realistic and based on reasonable assumptions!

3.2 Presentation

This section provides guidelines for the actual presentation to make it successful for you and your group members as well as entertaining and interesting for all other colleagues. Please recall that your performance in the presentations will be evaluated and is part of the final grade.

- The talk length is limited to 20 min and will be aborted by the teaching personal at that point.
- Every member of the group should talk in (roughly equal) parts. Naturally, this may entice you to split the task and make three separate mini-presentations. However, you can only benefit from your colleagues if you work together on this task. Therefore, we highly encourage collaboration instead of cooperation.
- Assume the talk to take place at a conference, thus your audience is familiar with the overall topic and the task but not your specific implementation. This means that there is now need to repeat what the assignment is about or how the algorithm works.
- Choose your content wisely. Too much will overwhelm the audience. It is not expected that you present every little detail. Some parts are better covered in the follow-up discussion (for that you can e.g., prepare backup some slides).
- Keep a clear structure in your talk. You may want to follow the order of the subtasks in the assignments (i.e., first the analysis and then the concept) but you do not have to. For the audience grasping new stuff is easier when it is connected to things that are already known. If possible use such attachment points.

- Design your slides properly. In this regard less is more. How much text is optimal often depends on who you ask. Some demand more to also capture people that are not listening while others propose a maximum word count of 20 per slide. Overall you should avoid full sentences, make pictures well readable and focus on the message you want to convey.
- The arguably hardest part is the talking itself. Try to speak slowly, maybe even slower than you anticipate. Making pauses is not always a bad thing but gives the audience some time to process the information just heard.
- Try to convince the audience that your investigations are sound and your solution is appropriate. Therefore, properly present your gathered data and explain it. Tell the listeners what conclusion you drew from your observations and how the it influenced your concept.
- The talks must be given in English. However, during discussion you may switch to German if it allows you to better address a question.
- Follow the presentations of the other groups closely. This is in your best interest, since this might reveal weak spots in your own concept and/or additional potential for optimizations! Meaningful participation in the discussions will be valued positively in your assessment!
- Keep in mind that individual group members can be assessed differently.

References

[1] Peter Shirley. Ray Tracing in One Weekend. `https://github.com/RayTracing/raytracing.github.io`. visited November 2023.

Revision History

Revision	**Date**	**Author(s)**	**Description**
1.0	08.11.2023	FH, DM	Initial version

Author Abbreviations:

FH	Florian Huemer
DM	Dylan Baumann