SCALE FOR PROJECT RT (/PROJECTS/RT)

Introduction

Please respect the following rules:

- Remain polite, courteous, respectful and constructive
throughout the correction process. The well-being of the community
depends on it.

- Identify with the person (or the group) graded the eventual
dysfunctions of the work. Take the time to discuss
and debate the problems you have identified.

- You must consider that there might be some difference in how your
peers might have understood the project's instructions and the
scope of its functionalities. Always keep an open mind and grade
him/her as honestly as possible. The pedagogy is valid only and
only if peer-evaluation is conducted seriously.

Guidelines

- Only grade the work that is in the student or group's
GiT repository.

- Double-check that the GiT repository belongs to the student
or the group. Ensure that the work is for the relevant project
and also check that "git clone" is used in an empty folder.

- Check carefully that no malicious aliases was used to fool you
and make you evaluate something other than the content of the
official repository.

- To avoid any surprises, carefully check that both the correcting
and the corrected students have reviewed the possible scripts used
to facilitate the grading.

- If the correcting student has not completed that particular

project yet, it is mandatory for this student to read the
entire subject prior to starting the defence.

- Use the flags available on this scale to signal an empty repository, non-functioning program, a norm error, cheating etc. In these cases, the grading is over and the final grade is 0 (or -42 in case of cheating). However, with the exception of cheating, you are encouraged to continue to discuss your work (even if you have not finished it) in order to identify any issues that may have caused this failure and avoid repeating the same mistake in the future.

Attachments

General presentation of RTv1 and RT

General presentation of RTv1 and RT (https://cdn.intra.42.fr/video/video/921/rtv1-rt_en.mp4)

Présentation générale RTv1 et RT

Présentation générale RTv1 et RT (https://cdn.intra.42.fr/video/video/101/rtv1-rt.mp4)

Subject (https://cdn.intra.42.fr/pdf/pdf/1871/rt.en.pdf) Demo (/uploads/document/document/30/Demo.zip)

Preliminaries

Reminder : Remember that for the duration of the defence, no segfault, nor other unexpected, premature, uncontrolled or unexpected termination of the program, else the final grade is 0. Use the appropriate flag. This rule is active thoughout the whole defence.

Basic stuff

Check the following:
- Something was submitted
- The author file is at the root of the repository and formatted
as explained in the subject.
- Norm is OK (using the norminette)
- The whole groupe is present

If at least one isn't ok defence the is over and final grade is 0.

✓ Yes ✕ No

Group organisation

Evalute here how the was the group organised to work through the corewar project. Here again a lot of scenarios are acceptable stay open. Don't validate this if you feel like like the group is messy and didn't really show any bit of organisation, or time management. This question is purely objective and is recognized as such. Know that subjective judgment of a hierachy is part of your professionnal future and even if you don't understand it yet, you can act on it.

✓ Yes ✗ No

Mandatory part

This part match up the RTv1. It is mandatory and eliminatory. If parts are missing, the defense ends, the final grade is 0. As stipulated in the subject: "The mandatory part is worth 0 and options will reward points only if the mandatory part is 100% complete". The subjects requires 3 scenes (see illustration in the subject) to validate quickly and easily the mandatory part. That's when the groupe is supposed to raytrace them.

Display's technical component

In this section we'll evaluate the display's technical component. Run the program and execute the following 3 tests. If at least one fails, no points will be awarded for this section. Move to the next one.

- A window must open when launching the program and stay open during the whole program's execution.

- Hide either part of or the whole window with another one or the screen's borders, minimize the window and maximize it back. In every cases, the window's content must remain consistant.

- Ask the student to show you the code prooving that expose was managed properly and doesn't involve a re-calculation of the whole image or even only a part of it.

✓ Yes ✗ No

The 4 Basic Shapes

In this section we'll evaluate the 4 basic shapes. Run the program and execute the following 4 tests. If at least one fails, no points will be awarded for this section. Move to the next one.

- Place a sphere at the following coordinates {0, 0, 0}, the camera facing the sphere, and display the rendered image. The sphere should be visible and displayed without glitch.

- Place a plane with a null z value, the camera facing the plan, and display the rendered image. The plane should be visible and displayed without glitch.

- Place a cone's center at the following coordinates {0, 0, 0} extending along the y axis, the camera facing the cone, and display the image rendered. The cone should be visible and displayed without glitch.

- Place a cylinder extending along the y axis, the camera facing the cylinder, and display the image rendered. The cylinder should be visible and displayed without glitch.

✓ Yes ✗ No

Translations and rotations

In this section we'll evaluate that rotation and translation transformations can be applied on scene's objects. Run the program and execute the following 2 tests. If at least one fails, no points will be awarded for this section. Move to the next one.

- Place two spheres at the following coordinates {0, 0, 0}, the camera facing those spheres. Then put a translation on one of the two spheres oriented in a direction parallel to the camera's, of a greater distance than the sphere's diameter and display the rendered image. Both spheres should be visible and displayed without glitch.

- Place a cylinder extending along the y axis, the camera facing the cylinder. Then put a 90° rotation (PI/2 radian) along the z axis and display the rendered image. The cylinder should be visible and displayed without glitch.

✓ Yes ✗ No

Multi-objects

In this section we'll evaluate that it's possible to put several object in one scene. Run the program and execute the following 2 tests. If at least one fails, no points will be awarded for this section. Move to the next one.

- Place several intersecting objects on the scene, such as for example a sphere and a cone, and display the rendered image. Both objects should be visible and displayed without glitch. (especially where both object intersect).

- Execute the same test, but ensure it's possible to place several times the same object, for example two cylinders and a plane.

✓ Yes ✗ No

Camera's position and direction

In this section we'll evaluate that rotation and translation transformations can be applied on scene's objects. Run the program and execute the following 4 tests. If at least one fails, no points will be awarded for this section. Move to the next one.

- Generate a random scene and place the camera extending along the x axis pointed towards the following coordinates {0, 0, 0} and display the rendered image. The scene must be visible and displayed without glitch.

- Generate a random scene and place the camera extending along the y axis pointed towards the following coordinates {0, 0, 0} and display the rendered image. The scene must be visible and displayed without glitch.

- Generate a random scene and place the camera extending along the z axis pointed towards the following coordinates {0, 0, 0} and display the rendered image. The scene must be visible and displayed without glitch.

- Generate a random scene and place the camera at a random location which isn't on any axis or a diagonal, pointed towards the following coordinates {0, 0, 0} and display the rendered image. The scene must be visible and displayed without glitch.

✓ Yes ✕ No

Brightness 1/2

In this section we'll evaluate brightness on scene's objects.
Run the program and execute the following 2 tests. If at least one fails, no points will be awarded for this section. Move to the next one.

- Place a sphere at the following coordinates {0, 0, 0}, the camera facing the sphere, and put a spot left or right of the camera but positioned in such a way that the sphere will be lit sideways. Display the rendered image. The sphere should be visible, enlightened and displayed without glitch.

- Place a sphere at some coordinates resulting from a translation, the camera facing the sphere, and place a spot left or right of the camera but positioned in such a way that the sphere will be lit sideways. Display the rendered image. The sphere should be visible, properly enlightened and displayed without glitch. Properly means that the halo of light should be computed after translation not before.

✓ Yes ✕ No

Brightness 2/2

In this section we'll evaluate shadow management generated by scene's objects. Run the program and execute the following 2 tests. If at least one fails, no points will be awarded for this section. Move to the next one.

- Place a vertical spot, a sphere and a plane. The spot liting the sphere's position to create a sphere shadow on the plane. Put the camera aside so we can see the sphere, the plane and the sphere's shadow on the plane. The shadow must be properly displayed and without glitch.

- Put a complex scene together with several objects like on illustration V.6 page 10 of the subject. Shadows must be properly displayed and without glitch.

✓ Yes ✕ No

Shine effect

In this section we'll evaluate shine effect on the scene's objects. Run the program and execute the following test. If it fails, no points will be awarded for this section. Move to the next one.

- The shine effect is characterized by the saturation of brightness towards the color of the spot, where the object's surface is perpendicular to the light. Put togethere a scere with a spot enlightening a sphere without shine effect and then with shine effect. Is the shine effect clearly visible?

✓ Yes ✕ No

Multi-spots

In this section we'll evaluate that it's possible to have several spots in the same scene. Run the program and execute the following test. If it fails, no points will be awarded for this section.

- Put together a scene with several objects including at least a plane on which shadows will be projected as well as 2 spots at the minimum. Check that brightness, shadows and shine effect (if implemented) work properly.

✓ Yes ✕ No

Options

Reminder : Remember that for the duration of the defence, no segfault, nor other unexpected, premature, uncontrolled or unexpected termination of the program, else the final grade is 0. Use the appropriate flag. This rule is active thoughout the whole defence. There is a lot of options. Since the large posibility and range of possible options for the RT, the success grade isn't very high. With a fair number of option it should be a pass but less XP than with MORE options.

Direct light

We're blinded by light spot facing us.

✓ Yes ✕ No

Parallel light

A parallel light can light up the scene, following a precise direction. (Unlike a spot that will emit light to a precise point).

✓ Yes ✕ No

Ambiance light

No objects is never really in the dark.

✓ Yes ✕ No

Ambiance ++

More points if the ambiance light can be managed from a configuration file.

✓ Yes ✕ No

File ++

Scene files are in XML, or following a proper stucture or hierachy. To make it simple it's not just a file with one information per line or basic blocs separated by just an empty line..

✓ Yes ✕ No

Limited objects

For this section give one point for each of the following:

- It's possible to slice objects on the x,y,z axes.
- It's possible to choose the slice from simple or really position (a cylinder can be sliced following its own axis or following a real axis)
- Rotations and translations still works after the slice.
- The slice effect is unique to every object and not applied on all of them.
- It's possible to slice the plane differently than on the axes (if you limit it on x and y, you'll get a square) like for example a triangle or a disc.

Rate it from 0 (failed) through 5 (excellent)

○

Textures

For this section give one point for each of the following:

- It's possible to apply a texture on at least one of the 4 basic objects.
- It's possible to apply a texture on the 4 basic objects.
- It's possible to stretch (or the opposite) a texture on an object.
- It's possible to shift a texture on an object.
- Another library than minilibX and its xpm was used to load texture (jpeg, png, ...).

Rate it from 0 (failed) through 5 (excellent)

○

Negative objects

It's possible to substract an object from another or deform an object with another one. For example:

- A negative sphere that will make a hole in a plane
- A cylinder will deform another perpendicular cylinder to its core.

✓ Yes ✕ No

Usual visual effects

One point per implemented effect:

- Antialiasing
- Cartoon effect
- Motion blur
- Sepia or any other color filter
- Simple stereoscopy (like red/green glasses)

Rate it from 0 (failed) through 5 (excellent)

○

Technical effects

Here again one point per implemented effect:

- Clustering rendering (computed on several computers) (2 points)
- Multi thread computing
- The rendering is really fast
- It's possible inside the RT to save, screenshot the rendered image.

Rate it from 0 (failed) through 5 (excellent)

○

Shadows and transparency

Is the shadow more or less darken if the object is transparent?

✓ Yes ✗ No

Disruptions

One point per implemented disturbance:

- Normale disruption: using sine for example which gives a wave effect.
- Color disruption: checkerboard for example.
- Color disruption: a more complicated algo to disrupt the color.
- Color disruption: a very complicated algo for example Perlin noise (this one is worth 2 point expect if it's the only one implemented then don't count the last point).

Rate it from 0 (failed) through 5 (excellent)

○

Simple native objects

Everything that can be solved with a smaller or equal complexity
than sphere/cylinder/cone (which are of second degree of complexity)
mostly paraboloid et hyperboloid. If at least one object of this
nature this section is validated.

✓ Yes ✕ No

More texture applications

One point per implemented option:

- A texture can be used to disrupt an object's normale (bump
mapping roughly)
- A texture can be used to modify at some places of an object
its transparancy.
- A texture can be used to limit or slice an object.
- A texture on a semi-transparant object serve as a slide and
is projected on another object. (2 points)

Rate it from 0 (failed) through 5 (excellent)

○

Composed elements

It is possible to define a composed element using simple objects.
For example a cube can be made with 6 limited planes, a "glass"
can be made with limited cone + cylinder + sphere. It's possible
to put several time the same composed elements but at different
positions or orientations (if it's not the case, the composed
element is useless...)

✓ Yes ✕ No

Reflection and transparancy

One point per implemented option:

- Reflection works, there is a mirror effect.
- It's possible to change the % of reflection (it's not all or nothing)
- Transparancy work we can see through

- The refraction index works (if required check the Descartes formula in the code)
- It's possible to change the transparancy %.

Rate it from 0 (failed) through 5 (excellent)

○

Environment

5 possibility for 5 points:

- There is a summury interface: a graphic loading message, a progress bar, something more than just terminal messages.
- There is a cool interface (made with gtk or QT) with configuration elements like file loading, render control, etc. (if true also count for the first point)
- It's possible to interact with the scene live (camera or object position, colors or textures...) without rerunning the program.
- It's possible to automatically render with modification between the rendering (no interface needed a serie of scripts can be used here).
- It's possible to automatically render objects for a scene for example a torus made of a serie of sphere, and helix made of spheres and cylinders...

Rate it from 0 (failed) through 5 (excellent)

○

More options

It's possible to make a lot of cool stuff with a RT.

The last.... and the least

Is it beautiful? It's 100% subjective, but everyone has different tastes.

✓ Yes ✗ No

In bulk

Here again one point per implemented option:

- A video made from your RT (share the love on the forum)
- Modelers files: it's possible to import pov or 3ds files (for example), and you can render them with your RT.
- Using 3D TV technology or OculusRift !
- Infrequent spot: like a light bulb filament, the light source is infrequent and shadows don't have sharpness.
- Any other crazy stuff.

Rate it from 0 (failed) through 5 (excellent)

○

Caustics and/or Global illumination

That's super cool don't forget to share images on the forum and slack.

✓ Yes ✕ No

Exotic objects

One point per implemented exotic object:

- Perforated cube
- Table cloth
- Torus
- Random equation resolution from a configuration file (the GNU lib does it well)
- More (fractal objects, etc.) as much as implemented up to 5.

Rate it from 0 (failed) through 5 (excellent)

○

The Moebius ribbon

A cool and well implemented Moebius ribbon !!

✓ Yes ✕ No

Ratings

Don’t forget to check the flag corresponding to the defense

✓ Ok

★ Outstanding project

Empty work

Incomplete work

No author file

Invalid compilation

Norme

Cheat

Crash

Incomplete group

⊘ Forbidden function

Conclusion

Leave a comment on this evaluation

Preview!!!

General term of use of the site
(https://signin.intra.42.fr/legal/terms/6)

Privacy policy
(https://signin.intra.42.fr/legal/terms/5)

Legal notices
(https://signin.intra.42.fr/legal/terms/3)

Declaration on the use of cookies
(https://signin.intra.42.fr/legal/terms/2)

Terms of use for video surveillance
(https://signin.intra.42.fr/legal/terms/1)

I
(https://sig