Inhaltsverzeichnis

1 Project Description

1.1 General Problem Definition

UHF-RFID systems demands a reliable transponder detection. A reliable RFID detection system can activate all the RFIDs in a specific range from the gate's center. To activate an RFID, an electromagnetic field strength is required in transponder position. It is the UHF-RFID antennas installation that determines the electromagnetic field strength distribution in the area around the gate. The main goal of this project is to detect the areas in the gate which the magnetic field strength is not in a specific acceptable domain. Therefore, the user is able to change the antenna installation to make the transponder detection more reliable.

1.2 IDP Goals

First of all, we have to find a way to make the real-world environment consistent with the environment which we have in the application because finally user should be able to map what he/she sees in the application to the real-world. Then we should start to collect the pose and magnetic field strength data; and finally, we should find a way to visualize the magnetic field in a specific range. So, the main task is first divided into three main sub-tasks:

- **Setting up Scene** In this part we are going to set up the size of the gate, the origin of the part of the environment which are going to measure the magnetic field in it.

- **Collecting Data** First of all we have to define the resolution of the data which we are going to achieve. We will make a grid in the area of the interest and we will try to store one value for each element of the grid. The resolution of the grid, determines how precise do we need to collect data. This task will be done by moving a dipole in the area of interest and collect the data about the dipole pose and magnetic field strength in each point. The dipole is able to measure the magnetic field which is received from the antennas and there is tracking system which can measure where the dipole is.

- **Visualizing Data** Then, the user wants to visualize the data in the grid in a way which the magnetic field in different areas of the gate can be well-presented in classified areas. Areas which their magnetic field strength are in some specific range is displayed and colored based on the user preferences.

The main contributions of this IDP project was to first use specific interaction method using tracking system to define the virtual world corresponding to the real world. As the later contribution, I have implemented a method to inform the user about the quality of measurements in parts of the area of interest. Finally, appropriate visualization methods (sphere/torus glyph visualization, and direct volume rendering) are utilized to show the measured values in appropriate way to the user.

1.3 Problem Solutions

1.3.1 Setting up the scene

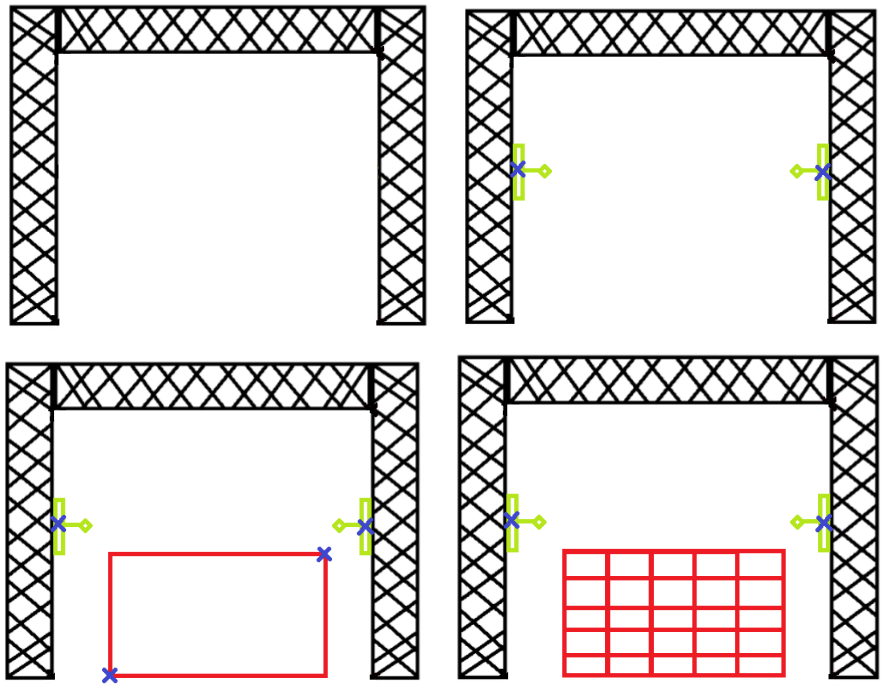
To set up the scene, we have to define the gate dimension and origin, and the area of interest. The area of interest is the area which magnetic field measurements are done there. Additional, to these two main needed specifications, we want to define the antenna position and orien-tation on the gate to make the visualization en-vironment closer to the real world.

a. Finding the 4 inner corners of the gate to determine the gate dimensions and cen-ter. Tip: Why 4 corners? Because calculating the center of the gate is hard to do in real world but putting the dipole's head in the corners of the gate is pretty simple. Then, finding the center of the gate is easy to calculate.

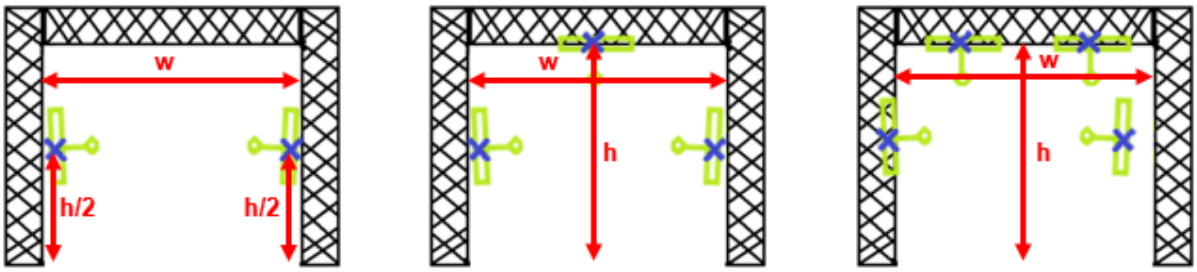
b. Determining the antenna position and orientation. This step is not necessary but to make the visualiza-tion closer to the real world we are trying to include that. To determine the antenna position and orien-tation, we will put the dipole close to the antenna with almost the same orientation of the antenna and then ask the application to store this position as an antenna position and orientation.

c. Then we will use the dipole again to determine the area of interest for measurements. First we put the dipole in one corner of the area of interest and then we will put it in another corner of the area of the interest. As it is obvious, here we only determine the bounding box of the area of interest. If we want, we can use some other meshes using the application. But this is not our goal. We just want to fill a 3d cube of data.

Using these steps, the scene will be set up. During the scene setup the user is able to see the3 results until now. Additionally, during all these set up scene there should be good tutorials to tell the user how to define his/her goals.



Depending on number of defined antennas, the grid dimension can be defined. Therefore, if two antennas are defined, it is assumed that the antennas are in the sides of the gate. If three antennas are defined, it is assumed that the third antenna is on top of the gate. When four antennas are defined, it is assumed that the forth one is also on top of the gate.



1.3.2 Collecting the data and user assistance

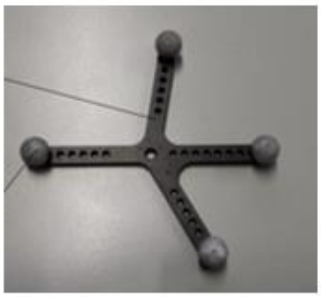
As the first step, we should determine the resolution of the box. Possibly, it would be a widget which is opened and user is able to enter the number of segments along length, width and height of the bounding box of the area of inter-est. Then, everything is setup and we are able to scan the area of interest.

While we are measuring the magnetic field, we should determine the position of the dipole, store the measured values, dipole position and orientation to a data structure. During collecting the data via scanning, the application should show which part of the area is scanned and which part is not scanned. Also, there should be a mechanism to help the user to find the parts which are not scanned and needed to be scanned to finish the measurements.

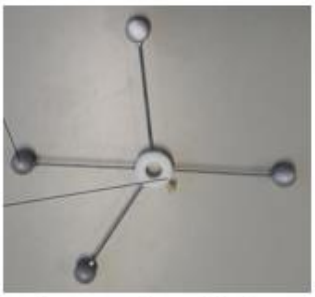
As we are measuring the values in some arbitrary points and there is not a specific structure in the measured points, the dataset which is going to be achieved is an unstructured dataset. Some of the visualization methods like torus glyph visualization are going to use these points directly and show them as discrete representations but some other visualization method like direct volume rendering are going to visualize the magnetic field as a continuous function, therefore we need some interpolation methods to approximate the magnetic field values in the points which their corresponding values are not available. In this part, we are going to present two different solutions to this problem:

1.4 Markers

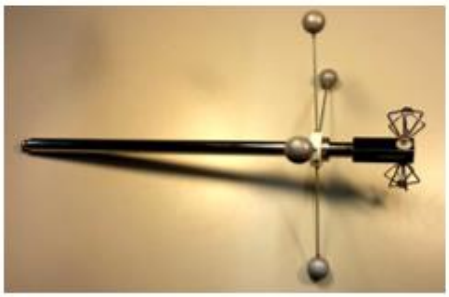
In this project, a tracking system is used to set up the virtual world, track the measurement device and etc. The following markers are used to reach these goals. The left marker is used to define the coordinate system's origin. The middle one is used to define the antenna pose (position and orientation), and the last is used to define track the measurement device.



Coordinate System Origin Marker

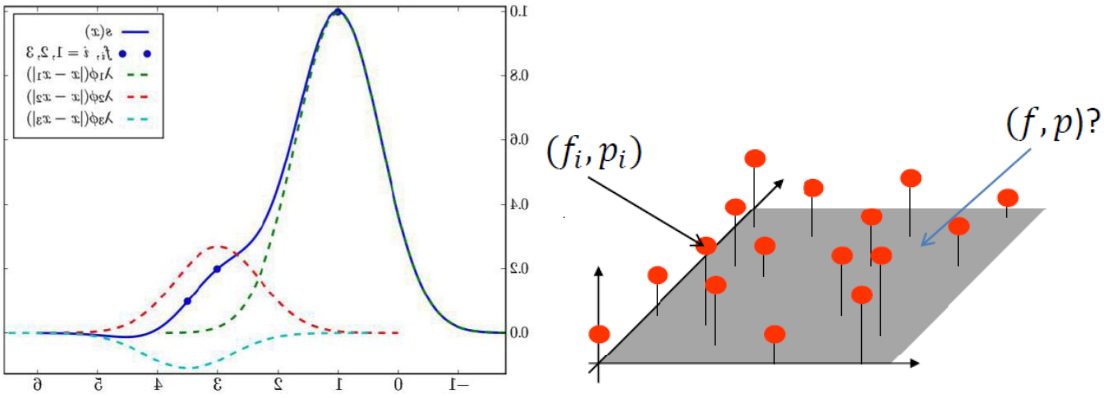


Antenna Pose Marker



Dipole Tracking Marker

Fitting a function to measured values



One of the ways to approximate the magnetic field in an arbitrary point is to build a continuous function based on the known discreet measurements and use the function to approximate smoothly the unknown values. As above figure shows, it is possible to approximate a continuous function if we have the function value in some discrete points. Therefore, as we have measured the magnetic field for some points in the space, we can also approximate the function. It will help us to calculate the magnetic field in a point which its value is not measured but it is needed. Given is a set of n points $p_i = (x_i, y_i, z_i)$ and function values $F_i(p_i)$, a function value $F(p)$ is created to interpolate between values of the points p_i.

Shepard method is one of the ways to calculate this function. It is an inverse distance weighted interpolation technique and the function can be written as follows:

$$F(p) = \frac{\sum_{i=1}^{n} \frac{F_i}{\|p-p_i\|^2}}{\sum \frac{1}{\|p-p_i\|^2}}$$

where $F(p_i)=F_i$. This method is easy to implement but has undesirable property that limit its usefulness for most of the practical applications. The interpolation function is going to become a "flat point" at points p_i as the derivatives are going to become zero. So, we are going to use another method to solve this problem.

In this method, also we are going to use inverse distance weighted interpolation strategy method but by using some functions $r(x)$ which is not getting infinity in the points p_is. To form this function, we have known values on the measured points and their corresponding function values (p_i, f_i). So, simply for an arbitrary point x, we can form the $f(x)$ function by using the following equation:

$$f(x) = \sum_i w_i r(\| p_i - x \|)^2$$

where w_i are the weights to be calculated for the measured points (which will be used in calculating the function value in arbitrary point), the p_is are the measured points position in the space and x is the point which we are going to calculate the function value in it. The w_i can calculate just by putting the p_i values as the x and their corresponding f_is in f(x), then we have n equations with n unknowns (w_i) which the unknown w_is can be calculated using some linear solvers like Gaussian elimination. We are going to use $r(x)=\exp(-x^2)$, because this function value is going to quickly decrease by increasing the distance from p_i so the influence of the point on the function value is going to decrease.

Also, this method is not easy to use as calculating the weights needs too much computation power. It can be used for more accurate visualization but it is hard to use during the real time visualizations. Also, the computation will increase by measuring more data in more points; because we will sample at least 10 times in each direction, so for one minutes measurement, we will have 600 equations with 600 unknown values which should be calculated. So, it makes using this method even hard in off-line visualization.

Making a Grid, Interpolate in it.

Basically, the idea is to create a grid with sufficient granularity and store each measured value in the grid, so that we can simply average the measured values in each element and store it in the grid. The good thing about this method is that it makes the interpolation easier, as we can assume that these values are in the center of each element, therefore for each arbitrary point in one of the elements of the grid, interpolate between the current element values and neighboring elements. While this method is fast, the accuracy of it is highly dependent on the granularity of the grid. Therefore, it should be decided very carefully.

In our application, we have used this method for implementing the continuous magnetic field visualization. The user is able to specify the granularity of the grid, then the grid will be filled during measurement. The grid's element magnetic field value is going to be calculated with the following equation:

$$F = \frac{\sum_i F(p_i) * w(p_i)}{\sum_i w(p_i)}$$

where F is the grid's element value, $F(p_i)$ is the measured value for the i-th point inside grid's element, and $w(p_i)$ is the calculated weight for the measured value in p_i. The points are weighted based on the distance to the center of the grid's element. The weight has maximum value in the center of the grid's element and min value in the corners of the element. It is calculated as follows:

$$w(p) = 1 - \frac{\| p - C_e \|}{\sqrt{\frac{D_x}{2}^2 + \frac{D_y}{2}^2 + \frac{D_z}{2}^2}}$$

where $w(p)$ is the calculated weight for point p, C_e is the position of the center of element in the space, and D_x, D_y, and D_z are the dimensions of the element in the space.

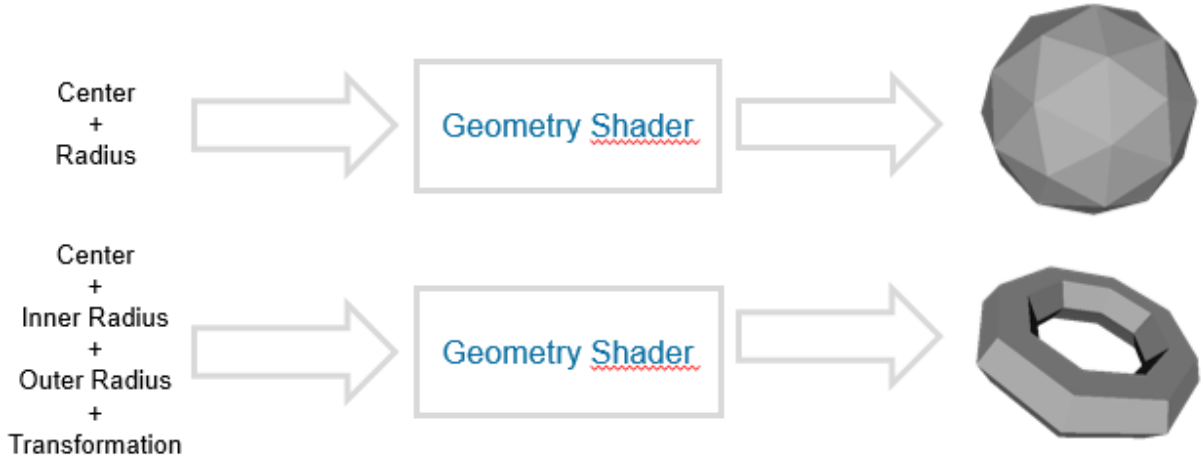
Hybrid method

As we discussed before, fitting the function on the data needed so much computation resource even if we use it for the non-real time visualization because the number of measured values will increase by time, therefore calculating the weights will need more computation, also interpolating will takes more computations. Furthermore, we discussed the creating grid method and then interpolate in the grid. The interpolation in this method is fast but its accuracy is highly dependent on granularity of the grid.

In this method, we are going to combine two previously discussed methods, and make a fast and more accurate method. The idea is to create a grid with sufficient granularity, then store the values for each element of the grid and fit a function for values stored for the grid's element. This method is well-fitted to visualization after measurements, because fitting the function, needs the weight calculations and it need too much computation resource if we do it during the data addition or modification.

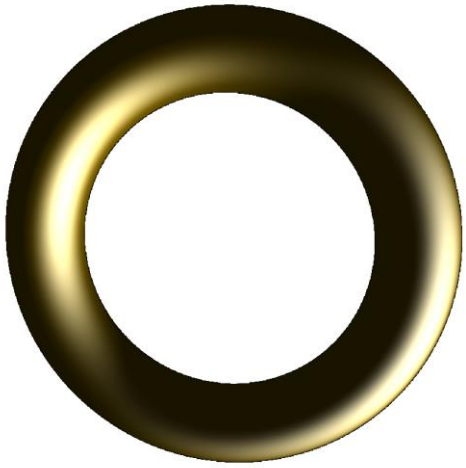
1.5 Sphere/Torus Glyph

To reduce the data transfer between CPU and GPU, minimum data is sent to the GPU via buffers. Then, the primitive is going to be generated on the GPU using geometry shader.



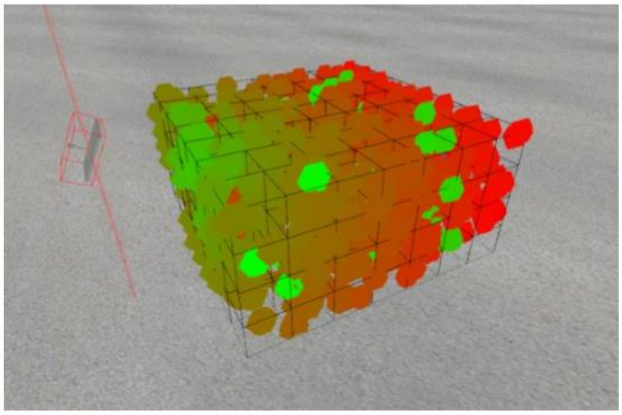
1.6 Sphere/Torus Higher Order Primitive

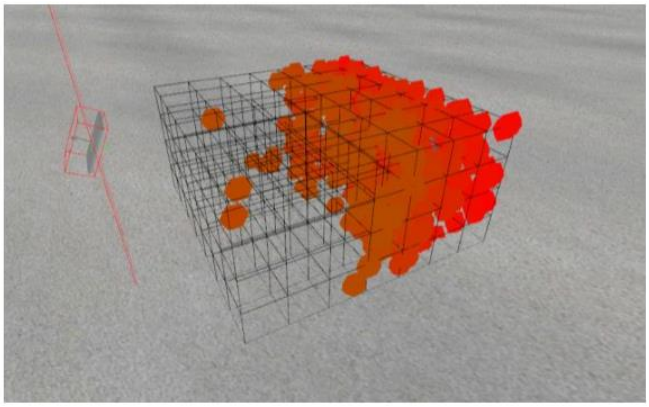
To not only reduce the data transfer, but also the computation power for high number of primitives, the minimum data is sent to GPU for each primitive. Then, a quad covering the area which primitive will cover will be generated in the geometry shader. Afterwards, the primitive is rendered on the screen by intersecting the ray-primitive equation. This will ends up with a smooth, pixel-accurate primitive.



1.7 Transparency

Additionally, the project supports transparency to focus on part of dataset more. A threshold value is entered. The transparency for values bigger than threshold and smaller than are defined. One sample is shown as follows:





1.8 Direct Volume Rendering (DVR)

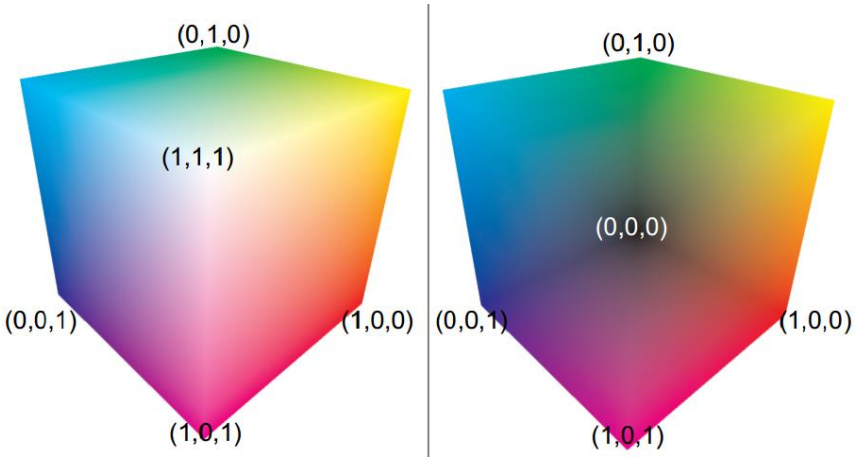
Direct volume rendering performs ray cast in the grid of collected data from the

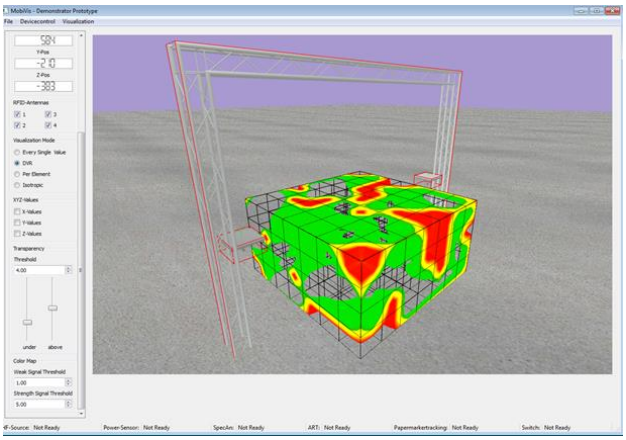
bounding box's entry point towards the exit point with steps of a fixed size. At each step, sample the

3D texture and compare the result with the high/dweak electromagnetic field strength. Stop when the ray reaches the exit point. During the ray casting, the ray will step iteratively through the volume with a fixed step-size, starting at the entry point, and in every step do the following:

1. Sample the volume scalar value (s).
2. Apply the electromagnetic field strength mapping to get an RGBA value.
3. Adjust the alpha value to the step-size by multiplying it with h (h is the step size of ray casting). This makes the overall opacity in the final image independent of h.
4. Update the current accumulated RGBA value using the rule for front-to-back alpha blending.
5. Terminate if you reach the exit point or if accumulated gets close to 1 (= early ray termination: everything behind won't have an effect on the final color), and return the accumulated value.

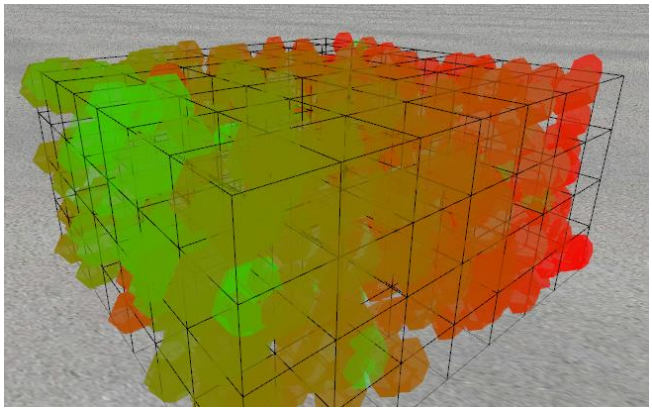
The direct volume rendering is implemented as follows in the fragment shader. Additionally the entry and exit points are shown in the following figure.

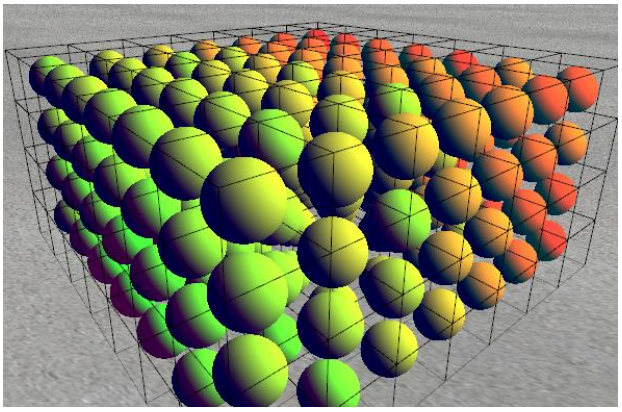




```
void main() {
  vec3 pos = localPos.xyz ;
  vec3 org = camera_pos;
  vec3 dir = normalize(pos - org);
  vec3 t_min = (min_corner - org) / dir;
  vec3 t_max = (max_corner - org) / dir;
  vec3 tmp0 = min(t_min, t_max);
  vec3 tmp1 = max(t_min, t_max);
  t_min = tmp0;
  t_max = tmp1;
  float t_entryPoint = max(t_min.x, max(t_min.y, t_min.z));
  float t_exitPoint  = min(t_max.x, min(t_max.y, t_max.z));

  vec3  texCoords;
  float t = t_entryPoint;
  float t_prev = t;
  float alpha = 0;
  vec3 color = vec3(0.0f);
  while(t < t_exitPoint) {
    texCoords = ((org + t * dir) - min_corner) / (max_corner - min_cor-
ner);
    float s = texture(tex, texCoords).r;

    if(inRange(s, WeakThreshold, StrengthThreshold)){
      float alphaF = getAlpha(s);
      vec3  colorF = getColormap(s);
      alpha = alpha + (1 - alpha) * alphaF;
      color = color + (1 - alpha) * colorF;
      if(abs(alpha - 1) < 0.001f)
        break;
    }

    t += h;  // stepsize
  }
  outColor = vec4(color, alpha);
}
```

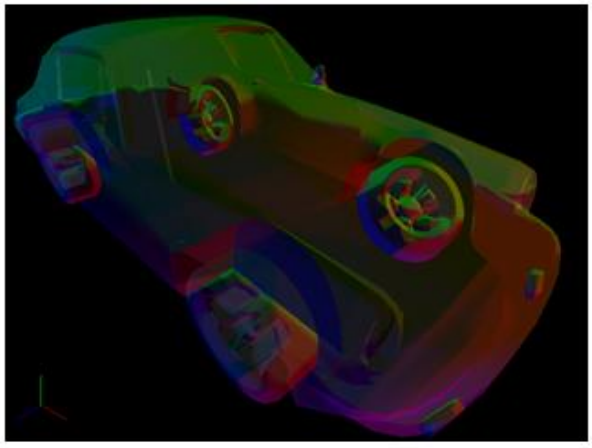
1.9 Results



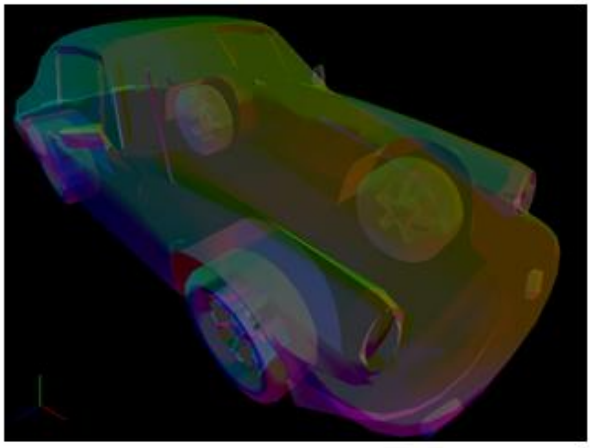


1.10 Future works

One of the main techniques that can be solved in this IDP project is to fix the transparencies. Currently, the transparency is not correct as the objects are not rendered in the right order. Following figure shows this in action:



Wrong rendering order



Right rendering order

Eidesstattliche Erklärung

Ich erkläre hiermit eidesstattlich, dass ich die vorliegende Arbeit selbstständig angefertigt habe.

Alle aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Diese Arbeit wurde bisher keiner anderen Prüfungsbehörde vorgelegt.

Garching, 13.4.2016

Seyedmorteza Mostajabodaveh