**Object Oriented C++ Geometry API**

**Personal Information**

I am Kalpit C. Thakkar, a Junior, studying at International Institute Of Information Technology, Hyderabad (AKA IIIT-H). I love Graphics and visualization, and I love playing soccer.

I have some basic experience in designing a Geometry API, but it didn’t incude primitives like BRL-CAD. I am doing honors at Centre For Visual Information and Technology (CVIT, IIIT-H), in the area of Raytracing. I have some familiarity with raytracing libraries like, PBRT, Appleseed and Intel’s Embree. I have been using C++ in all my coding endeavors.

Here are my details for contact :

E-mail addresses : [ceasy983@gmail.com](mailto:ceasy983@gmail.com)

IRC username : dracarys983

Phone : +917842964055

Timezone : UTC +0530

Primary Language : English

**Project Information**

* *Project Title*

Object Oriented C++ Geometry API

* *Brief description*

This project is a part of building the Geometry Engine of BRL-CAD, an engine built on top of BRL-CAD’s core geometry libraries. My aim is to build an Object Oriented C++ Geometry API for primitives implemented in core geometry libraries (written mostly in C).

My project consists of two main sections :

1. Adding classes for BRL-CAD core primitives
2. Adding following features for the core primitives and combinations of primitives :
   1. Centroid
   2. Volume
   3. Surface Area

My main focus would be on section 2. Section 1 is the next important thing.

* *Detailed Project description*

BRL-CAD has a core library for primitives written in C, called LIBRT. This implementation of primitives in C, do not provide a good interface for external applications, as change in implementation in LIBRT would need a change in applications as well.

This project comes under the Geometry Service Project, which aims at building a Geometry Engine that will provide an Object Oriented C++ interface to external applications that use BRL-CAD core libraries. This would make the code re-usable and would hide the implementation details. This would assure that change in implementation won’t affect the external application using the interface.

There are several areas of Geometry Engine which can be enhanced. I have chosen contributing in the following way :

I have divided my project into two main sections (according to priority) :

1. Adding features (Centroid, Volume and Surface area) :
   1. For all primitives,
   2. For combination of primitives.
2. Adding new primitive classes to coreInterface (Extrude, ARBN and ARS - in chronological order)

**Part 1)**

**This would be done for the primitives whose classes are designed in the coreinterface.**

**FOR PRIMITIVES (Subject to change after discussion)**

* Case 1 : The primitives in LIBRT, who have their centroid, volume and surface area callbacks implemented => Only 3 methods have to be added to the class which would provide an interface to the three features.
* Case 2 : The primitives in LIBRT, who don’t have either one or two or all three of the callbacks implemented =>

1. Implement the callbacks that are not present in LIBRT
2. Define the 3 methods in it’s coreInterface class to provide the interface

**FOR COMBINATION OF PRIMITIVES (Subject to change after discussion)**

* Research is going on about how to find the centroid, volume and surface of any arbitrary combination of primitives.
* A combination is a non-leaf node in the tree formed (CSG) from a scene. It is derived by applying operations such as union, intersection and / or subtraction of it’s child nodes.
* **For volume\* : (Routines are already there : gqa)** We can make a grid in the object space, with the grid having 1mm x 1mm cells. Now, we shoot rays on this grid. For each cell, the distance travelled by the ray inside the object in that cell (outdist - indist) will help us in finding the volume covered by that cell. We sum up the volumes of all such cells. (in cubic mm’s).
* **For Surface Area : (An approximation)** For this part, we can tessellate the combination (convert to mesh of triangles using NMG) and hence the surface area of the combination would be equal to the sum of area of all triangles.
* **For Centroid : (To be discussed)** For finding the centroid, we follow the same procedure as finding the volume, with the following additions :

1. We find the d = (outdist - indist) / 2 for each cell and store it.
2. We find the average “d” for each row by summing the d’s along the row and dividing by the number of columns.
3. We find the centroid by finding the average “d” from the d’s obtained in step (b).

\* As discussed with Sean, this won’t be a good method for extremely small and large objects.

**Part 2)**

**Adding new primitives to the coreInterface**

In this part, I will use OOP principles to design an efficient and clean interface for the primitives : Extrude, ARBN and ARS.

The main aim of this part is to make an interface such that it works smoothly with the internal structure of the primitive.

Extrude has been kept first in the chronological order as it depends on Sketch primitive and it has been implemented already last year.

The class design will be discussed with the mentor first and then will be worked upon.

All the code written will be thoroughly tested, cleaned and licensed. It will follow the conventions mentioned in the HACKING file. This project might be over-scoped, but I intend to stay with BRL-CAD ahead of GSoC as well. I’m not getting away without completing this project.

**The ideas for implementation have been arrived at after discussions on IRC, mails and some research, which involves a lot of dumb google-ing and then realizing it’s documented in BRL-CAD already.**

* *Deliverables :*

1. Centroid() function for all primitives in coreInterface
2. Tests for each of the Centroid() functions
3. Volume() function for all primitives in coreInterface
4. Tests for each of the Volume() functions
5. SurfaceArea() function for all primitives in coreInterface
6. Tests for each of the SurfaceArea() functions
7. Algorithm design for finding the :
   * 1. Centroid()
     2. Volume()
     3. SurfaceArea(), of a combination.
8. Centroid(), Volume() and SurfaceArea() functions for Combinations
9. Tests for each of these functions for Combinations
10. Extrude class design scheme
11. Extrude class Constructors, Modifier functions
12. Tests for the Extrude class
13. ARBN class design scheme
14. ARBN class Constructors, Modifier functions
15. Tests for the ARBN class
16. ARS class design scheme
17. ARS class Constructors, Modifier functions
18. Tests for the ARS class

* *Development Schedule :*

**There will be work done between April 1 and May 24. Not 40 hours/week, but close to 24 hours/week. During this time, I intend to do the following tasks =>**

* (April 1 - April 20) : Implementing the tests for already existing Primitive classes, for those who don’t have tests implemented already.

(April 22 - April 29)

**=========== (College) Semester End Exams ===========**

* (May 1 - May 8) : Reviewing the Pipe implementation done by Andrei Popescu last year, applying changes if required and merging it.
* (May 9 - May 24) : NURBS and BREP files are already written in C++. But after taking a look at it, I felt they need to be restructured. So, after discussing with Mentor, if necessary, this will be done.

**Coding period starts from 25th May =>**

* Week-1 (25th May - 31st May) :

1. Discuss the algorithm for finding the Volume of a Combination.
2. Discuss the small tweaks, optimizations that can be done on the standard algorithm discussed.
3. Document the algorithm, note the limiting conditions
4. Implement the Volume function for Combination as well as the interface.

* Week-2 (1st June - 7th June) :

1. Discuss the interface design for feature implementation of Cone, Sphere, Torus, Ellipsoid.
2. Implement the features and hence the interface
3. Write tests for these features and Volume function of Combination.

* Week-3 (8th June - 14th June) :

1. Discuss the algorithm for finding the Surface Area of a Combination.
2. Discuss the small tweaks, optimizations that can be done on the standard algorithm discussed.
3. Document the algorithm, note the limiting conditions
4. Implement the Surface Area function for Combination as well as the interface.

* Week-4 (15th June - 21st June) :

1. Discuss the inteface design for feature implementation of EllipticalTorus, parabolicCylinder and hyperbolicCylinder.
2. Implement the features and hence the interface
3. Write tests for these features and SurfaceArea function of Combination.

* Week-5 (22nd June - 28th June) :

1. Discuss the algorithm for finding the Centroid of the Combination.
2. Discuss the small tweaks, optimizations that can be done on the standard algorithm discussed.
3. Document the algorithm, note the limiting conditions
4. Implement the Centroid function for the Combination as well as the interface.

(29th June - 3rd July)

**============== MID TERM EVALUATIONS =============**

* Week-6 (4th July - 10th July) :

1. Discuss the interface design for feature implementation of Hyperboloid, Paraboloid, BoT and ARB8.
2. Implement the features and hence the interface.
3. Write tests for these features and Centroid function of Combination.

* Week-7 (11th July - 17th July) :

1. Discuss the interface design for feature implementation of halfspace, NMG, particle, Sketch and Pipe
2. Implement the features and hence the interface
3. Write tests for these features

* Week-8 (18th July - 24th July) :

1. INVESTIGATION PERIOD
2. Cover up on any implementation task that is falling behind schedule.
3. Cover up on documentation / testing tasks, if pending

* Week-9 (25th July - 31st July) :

1. Discuss the class design for Extrude primitive
2. Draw the class scheme
3. Implement the constructors, (.h and .cpp)

* Week-10 (1st August - 7th August) :

1. Implement the modifier functions (.h and .cpp)
2. Write the unit tests for Extrude
3. Documentation, cleanup

* Week-11 (8th August - 14th August) :

1. Discuss class design for ARBN primitive
2. Draw the class scheme
3. Implement the constructors (.h and .cpp)

* Week-12 (15th August - 21st August) :

1. Implement the modifier funciton (.h and .cpp)
2. Write unit tests for ARBN
3. Documentation, cleanup

* Week-13 (22nd August - 28th August) :

1. Discuss class design for ARS primitive
2. Draw the class scheme
3. Implement the constructors (.h and .cpp)

**Rest of the ARS and the final touchup would be done before the “soft” deadline (25th September).**

**Milestones :**

1. **Features for Combinations**
2. **Features for all primitives**
3. **New classes for 3 core primitives (Extrude, ARBN, ARS)**

* *Time availability :*

I have to work towards an honors project in the summers, which would require 10 hours of work per week. Hence, I would be able to work for 35 hours/week.

**Why me?**

Well, first of all, I have been searching for something that I would like to work on since I joined college in 2012. Finally, I found myself attracted to Geometry and Visualization, and I would love working towards it. This project is my first step in this direction.

Well, I am not the perfect guy for the assignment, but I can assure you I won’t let you down. This would give me more satisfaction than I have ever found anywhere before.

**Why BRL-CAD?**

Last year, I looked up on BRL-CAD, but was not motivated enough to contribute as I had not obtained experience in Graphics and Visualization (Raytracing, Geometry) yet.

I also looked up on Blender, but the same thing stopped me.

Well, since that day, I had decided that I will work towards contributing to either Blender or BRL-CAD. I tried talking on the IRC in both organizations and honestly, I liked BRL-CAD community better. So, that was it. I decided that I will conrtibute to BRL-CAD, and I haven’t looked back since then -- I don’t regret my decision at all.

Plus, I love the way Sean and others inspire students to keep working and getting better and cooler! I was never disheartened by anything in the community. So, shoot! I’m choosing BRL-CAD.

**I have been searching for an inspiration to work from a long time and I have obtained it from BRL-CAD community. It means a lot to me!**