3. Task scheduling in parallel computing
4. Project management in construction

4.9 Monte Carlo Method: Using Randomness to Solve Problems

The Monte Carlo method is a powerful technique that uses random sampling to solve problems that might be deterministic in principle. It's particularly useful for complex systems with many variables or when traditional analytical methods are impractical.

It is a tool that allows us to tackle problems that would be infeasible to solve through direct computation, making it invaluable in fields ranging from physics and finance to computer graphics and artificial intelligence. Some applications of Monte Carlo method are discussed below.

4.9.1 Estimating Pi: A Circle in a Square

Imagine a square with a side length of 1 unit and a circle inscribed within it. The circle's diameter is equal to the square's side, so its radius is 0.5 units. Now, let's conduct a thought experiment.

Throw a large number of pebbles (all assumed to be of the same size) randomly and uniformly at the square. Count how many pebbles land inside the circle versus the total number thrown.

The ratio of pebbles inside the circle to the total number of pebbles will approximate the ratio of the circle's area to the square's area. We can use this to estimate pi:

- Area of the square: $1 \times 1 = 1$ square unit
- Area of the circle: $\pi r^2 = \pi(0.5)^2 = \frac{\pi}{4}$ square units
- The ratio of these areas is $\frac{\left(\frac{\pi}{4}\right)}{1} = \frac{\pi}{4}$

So, if we multiply our pebble ratio by 4, we get an estimate of π. The more pebbles we throw, the more accurate our estimate becomes.

This method demonstrates how random sampling can be used to approximate a value (in this case, π) without direct calculation.

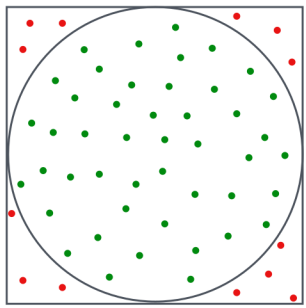

*Figure 4.6: Circle inscribed in a square. There are 47 green dots and 13 red dots. 47/60 is **0.78333...** which is close to $\frac{\pi}{4}$ or **0.78539...***

4.9.2 Raytracing in Computer Graphics

In computer graphics, particularly in rendering realistic lighting, the Monte Carlo method plays a crucial role.

This technique simulates the path of light rays as they interact with objects in a scene.

Figure 4.7: A ray traced image of a 3D model that I created in Blender, rendered using LuxRender

Instead of calculating every possible light path (which would be computationally infeasible), a Monte Carlo approach is used:

- Multiple light rays are randomly cast from each pixel of the image.
- The paths of these rays are traced as they bounce off surfaces, refract through materials, or get absorbed.
- The final colour of a pixel is determined by averaging the results of these random light paths.

Recent advancements in real-time raytracing for video games heavily rely on these Monte Carlo techniques, allowing for more realistic lighting, reflections and shadows in interactive environments.