Nessa terceira parte do desenvolvimento do nosso Ray Casting foram criadas novas funcionalidades para o mesmo, como por exemplo a câmera pinhole, o carregamento de malhas mais elaboradas e o adicional da cor para os objetos da classe Primitiva.

Para a primeira melhoria citada anteriormente(câmera pinhole), foi criada uma nova classe estendida da classe Câmera, que como diferença nos atributos se olharmos para a câmera ortográfica, não havia nenhuma, a não ser a introdução da distância focal. No que tange o algoritmo de ligação dos raios que partem da câmera para cada pixel da imagem, foi algo relativamente simples de se fazer com a ajuda da distância focal, bastando calcular a distância e a direção dos raios.

Para a segunda melhoria, foi criada uma classe TriangleMesh, que carrega os triângulos a partir de um arquivo. Primeiro, ela itera sobre as malhas presentes no arquivo. Para cada malha, itera-se sobre todas as faces, em busca dos índices dos vértices e dos vetores normais dos triângulos. Obtidos os índices, basta ir criando os triângulos dinamicamente, usando a classe Triangle, e ir armazenando esses triângulos num array. No fim, esse array é passado para a classe Scene, para os triângulos serem renderizados.

Já a última melhoria que foi a distribuição de cores para as primitivas, bastou apenas algumas pequenas modificações para funcionar. Foi criado um atributo chamado de cor para os objetos da classe primitiva, de forma que não importasse qual o tipo de primitiva que fosse ser usada, sempre existiria uma cor(no caso optamos por atribuir cores randomicamente). Além de criar o campo cor, foi alterado o algoritmo que verifica se houve intersecção ou não de cada raio para cada pixel, visto que caso houvesse essa intersecção, não seria mais passado uma cor entre branco e preto para aquele pixel, mas sim a cor da primitiva intersectada no mesmo.