NeRF Notes

Samuel Kinstlinger

May 2025

* In real-world environments, a robot must avoid obstacles, manipulate objects, and navigate efficiently. All these tasks depend on spatial awareness—the robot needs to know what is where, including semantic and metric data. In this situation, 2D images are not sufficient: 3D structure is essential. For example, a mobile robot needs to know the 3D shape of a room to plan collision-free paths. Additionally, a manipulation robot needs 3D object geometry to grasp correctly.
* However, robots often do not perceive their entire environment at once. This can be due to sensor range limitations, inherently directional aspects of sensors, self-occlusion, or external occlusions in environments that block light of sight to other areas. The robot only has information collected by sensors, meaning that to get a full understanding, robots must often move and perceive from a variety of viewpoints and angles. It must then use the totality of information it senses to construct a representation of the relevant aspects of an environment. This representation is known as a **map**.
* A robot must reason about occlusions rather than just moving to observe them because in many situations, movement is either not immediately possible, not safe, not efficient, or not informative enough. Reasoning fills these gaps with inference, prediction, or planning support.
* Points have inherent geometric relationships that should be stored functionally for the purposes of compactness

NeRF – Neural Radiance Field

Broadest Goals: We need a full map as visual occlusionWe need a compact representation to allow for efficient and effective storage, learning, and planning.

Goal: Find function F(x, d) = (c, σ).

* Inputs: The visual and spatial properties of a point depend specifically on the 3D location of the point. However, appearance can vary by angle. A wall might reflect light differently depending on whether you look at it head-on or at a grazing angle. View-dependent effects like specular reflection require direction-aware modeling.
  + x ∈ R3 - A point in 3D space—represented by its x, y, z coordinates. This tells the network where in space we’re evaluating. NeRF learns a function over the entire 3D volume, and x is the input to that function.
  + d ∈ R3 – A unit vector (direction of length 1) indicating the camera’s viewing direction as it looks at point x. Some surfaces look different depending on the angle you look at them. This is called view-dependent appearance (e.g., specular highlights, reflections).
* Outputs:
  + c ∈ R3 - The red, green, and blue color values emitted from point x in direction d. This is the appearance of the point from the camera’s perspective. It's what the renderer ultimately uses to construct the final image. When the renderer samples many points along a camera ray and combines their c values using volume rendering, it produces the final pixel color.
  + σ ∈ R>=0 - A scalar representing how much material exists at point x—specifically, how likely it is for light to be absorbed or emitted there. σ = 0: The point is completely transparent (like air or empty space). High σ: The point is highly opaque or solid (like the surface of a wall). It determines how much a point along a ray contributes to the final pixel color. This is crucial for volume rendering. NeRF doesn’t directly generate 3D surfaces—instead, it builds them up implicitly by assigning high σ values to points that make up visible surfaces.