Background on 3D Reconstruction from Images

Abstract

This document introduces the mathematical foundations of 3D reconstruction from 2D images, a key problem in computer vision. We describe the 3D scene, the information encoded in 2D images, and the perspective projection process that maps 3D points to 2D pixels. The goal is to recover the 3D structure of a scene using only 2D image data, a challenging task due to the loss of depth information in projection.

1 Background

1.1 What is 3D Reconstruction from Images?

The goal of 3D reconstruction is to recover the structure of a 3D scene using only 2D images. Consider a 3D scene represented by a set of points $P(P_x, P_y, P_z) \subset \mathbb{R}^3$, where each point has coordinates (P_x, P_y, P_z). These points might describe the surface of an object, like a statue.

Each image of the 3D scene, denoted I_n (where n is the image index), contains a set of pixel points $p(p_x, p_y) \subset \mathbb{R}^2$, with coordinates (p_x, p_y), which are projections of the 3D points. The process of capturing a 3D point in a 2D image $I\ n$ is modeled by a projection function:

$$\pi_n : \mathbb{R}^3 \to \mathbb{R}^2, \quad \underbrace{(P_x, P_y, P_z)}_{\text{3D point}} \mapsto (p_x, p_y)$$

This function represents how a camera maps a 3D point to a 2D pixel in the n-th image, similar to projecting a point on an object onto a photograph.

To reconstruct the 3D scene, we need to solve the inverse problem:

$$\pi_n^{-1}(p) = P(P_x, P_y, P_z) \in \mathbb{R}^3, \quad \text{given } p(p_x, p_y) \in \mathbb{R}^2$$

This means finding the 3D point P that projects to a given 2D pixel p. However, this problem is typically **ill-posed**, as multiple 3D points may project to the same 2D pixel, creating ambiguity. We will discuss this in detail in Section 1.4.

1.2 Information Encoded in 2D Images

A 2D image $I_n : \mathbb{R}^2 \to [0,1]^3$ provides various types of information about the 3D scene, such as color and texture. Each pixel $p(p_x, p_y) = [p_x, p_y]^\top \in \mathbb{R}^2$ is mapped to an RGB color value $I_n(p) = [r, g, b] \in [0,1]^3$, where $r, g, b \in [0,1]$ represent the red, green, and blue intensities.

The information available from a single image includes:

- **Pixel coordinates**: $p(p_x, p_y) \in \mathbb{R}^2$, representing the location of each pixel in the image plane.
- **Color values**: $I_n(p) = [r, g, b] \in [0,1]^3$, the RGB color at pixel p, describing the appearance of the scene.

- **Image gradient**:

$$\nabla I_n(p) = (\nabla r(p), \nabla g(p), \nabla b(p)) = \left(\underbrace{\left[\frac{\partial r}{\partial p_x}, \frac{\partial r}{\partial p_y}\right]^\top}_{\text{red channel}}, \left[\frac{\partial g}{\partial p_x}, \frac{\partial g}{\partial p_y}\right]^\top, \left[\frac{\partial b}{\partial p_x}, \frac{\partial b}{\partial p_y}\right]^\top \right) \in \mathbb{R}^6$$

  This captures local changes in intensity, indicating edges or texture information in the image.

- **Learned features** (optional): $\phi(I_n)(p) \in \mathbb{R}^d$, high-dimensional features extracted from the image using methods like convolutional neural networks (CNNs) or other feature extractors.

These data result from projecting 3D structures through a camera. For example, the color $I_n(p)$ may correspond to the visible surface of a 3D object, while $\nabla I_n(p)$ may indicate the edges or shape of that object.

1.3 The Forward Projection: From 3D World to 2D Image

We formalize the perspective projection process, which maps a 3D point $P(P_x, P_y, P_z) = [P_x, P_y, P_z]^\top \in \mathbb{R}^3$ to a 2D pixel point $p(p_x, p_y) = [p_x, p_y]^\top \in \mathbb{R}^2$ in the n-th image I_n, using the camera's parameters.

Camera Parameters

The projection depends on two types of camera parameters:
Extrinsics (world-to-camera transformation):

- **Camera center**: $C(C_x, C_y, C_z) \in \mathbb{R}^3$, representing the camera's position in world coordinates.
- **Rotation matrix**: $R \in SO(3)$, a 3x3 matrix that rotates the world coordinate system to align with the camera's orientation.
- **Translation vector**: $t = -RC \in \mathbb{R}^3$, representing the translation to align the world origin with the camera center.

Intrinsics (projection to image plane):

- **Intrinsic matrix**:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{3\times3}$$

  where $f_x, f_y \in \mathbb{R}$ are the focal lengths (in pixels), and $(c_x, c_y) \in \mathbb{R}^2$ is the principal point, where the camera's optical axis intersects the image plane.

Forward Projection Pipeline

The projection π_n maps a 3D point $P(P_x, P_y, P_z)$ to a 2D pixel $p(p_x, p_y)$ in I_n:

$$p_{\text{hom}} = K[R|t] \begin{bmatrix} P \\ 1 \end{bmatrix}, \quad p_{\text{hom}} = \begin{bmatrix} p'_x \\ p'_y \\ z' \end{bmatrix} \tag{1}$$

$$p(p_x, p_y) = \begin{bmatrix} p_x \\ p_y \end{bmatrix} = \begin{bmatrix} p'_x/z' \\ p'_y/z' \end{bmatrix}, \quad z' \neq 0 \tag{2}$$

Here, p_{hom} uses homogeneous coordinates, which include an extra coordinate to handle perspective scaling. The process involves:

1. **World to camera coordinates**: Transform $P(P_x, P_y, P_z)$ to camera coordinates:

$$P_c(P_{cx}, P_{cy}, P_{cz}) = \underbrace{R(P - C)}_{\text{rotate and translate to camera frame}} = \underbrace{[R|t]\begin{bmatrix} P \\ 1 \end{bmatrix}}_{\text{matrix operation}} \in \mathbb{R}^3$$

2. **Perspective projection**: Project P_c to the image plane:

$$p_{\text{hom}} = KP_c = \begin{bmatrix} p'_x \\ p'_y \\ z' \end{bmatrix}$$

3. **Normalization**: Convert to 2D pixel coordinates by dividing by the depth z':

$$p(p_x, p_y) = \begin{bmatrix} p_x \\ p_y \end{bmatrix} = \begin{bmatrix} p'_x/z' \\ p'_y/z' \end{bmatrix}, \quad z' \neq 0$$