3d Gaussian splatting

input

- q_i: rotation 3d Gaussian (quaternion)
- s_i: scale of 3d Gaussian (3d vector)
- h_i: spherical harmonics parameters of3d Gaussian
- α_i: opacity of 3d Gaussian
- p_{w_i}: the location of 3d Gaussian in world frame

where: i is the index of 3d Gaussian.
For ease of reading, i is omitted in the following text.

parameters

- R_{cw}: the rotation of camera (local frame)
- t_{cw}: the translation of camera (local frame)
- K: Intrinsic Parameters (i.e. f_x, f_y, c_x, c_y)

pipeline

1. Transform the location to camera frame.

$$\begin{aligned} p_c &= T_{cw}(p_w) \\ &= R_{cw}p_w + t_{cw} \end{aligned} \tag{1}$$

2. Calcuate the 3d Gaussian from rotation and color

$$\begin{aligned} \sigma &= \sigma(q, s) \\ &= \text{upper\_triangular}(RSS^TR^T) \end{aligned} \tag{2}$$

where:

$$R = \begin{bmatrix} 1-2(q_y^2+q_z^2) & 2(q_xq_y-q_zq_w) & 2(q_xq_z+q_yq_w) \\ 2(q_xq_y+q_zq_w) & 1-2(q_x^2+q_z^2) & 2(q_yq_z-q_xq_w) \\ 2(q_xq_z-q_yq_w) & 2(q_yq_z+q_xq_w) & 1-2(q_x^2+q_y^2) \end{bmatrix} \tag{2.1}$$

$$S = \begin{bmatrix} s_0 & 0 & 0 \\ 0 & s_1 & 0 \\ 0 & 0 & s_2 \end{bmatrix} \tag{2.2}$$

3. Project the 3D Gaussian to 2d image as a 2d Gaussian.

$$\begin{aligned} \sigma' &= \sigma'(\sigma, p_c) \\ &= \text{upper\_triangular}(JW\Sigma W^T J^T) \end{aligned} \tag{3}$$

where:

$$J = \begin{bmatrix} \frac{f_x}{x} & 0 & -\frac{f_x x}{z^2} \\ 0 & \frac{f_y}{z} & -\frac{f_y y}{z^2} \\ 0 & 0 & 0 \end{bmatrix} \tag{3.1}$$

$$p_c = [x, y, z] \tag{3.2}$$

$$W = R_{cw} \tag{3.3}$$

$$\Sigma = \text{symmetric\_matrix}(\sigma) \tag{3.4}$$

4. Cacluate the color from SH.

$$c = c(h, p_c)$$

5. Blend the 2d Gaussian to image.

5.1 inverse cov2d

$$\begin{aligned} \omega &= \omega(\sigma') \\ &= \text{upper\_triangular}(\Sigma'^{-1}) \end{aligned} \tag{5.1}$$

where:

$$\Omega = \Sigma'^{-1} = \frac{1}{ac-b^2}\begin{bmatrix} c & -b \\ -b & a \end{bmatrix} \tag{5.1.1}$$

5.2 Project the pc to image.

$$\begin{aligned} u &= u(p_c) \\ &= Kp_c/(Kp_c).z \end{aligned} \tag{5.2}$$

5.3 Cacluate the opacity for each pixel.

$$\begin{aligned} o_{jk} &= o_{jk}(\alpha, u, \omega) \\ &= g_{jk}(u, \omega)\alpha \end{aligned} \tag{5.3}$$

where:

$$\begin{aligned} g_{jk} &= \exp(-0.5 d_{jk}^T \Omega d_{jk}) \\ &= \exp(-0.5(\omega_0 d_{ij0}^2 + \omega_2 d_{ij1}^2 + 2\omega_1 d_{ij0} d_{ij1})) \end{aligned} \tag{5.3.1}$$

$$d_{jk} = u - x_{jk} \tag{5.3.2}$$

j k are the coordinates of the pixels respectively, j and k are omitted in the following for convenience.

5.4 Cacluate the color for each pixel.

The color on the jk pixel (γ_{jk}) depends on the blend of all 2d Gaussian colors on the jk point.

The symbols $\mathbb{C}$ and $\mathbb{O}$ represent the sets of colors and opacities for all 2D Gaussians, respectively. They are sorted by distance, meaning that the smaller the value of i, the smaller the distance between this 2D Gaussian and the camera.

$$\begin{aligned} \gamma &= \gamma(\mathbb{C}, \mathbb{O}) \\ &= \sum_{i \in N} o_i c_i \tau_i \end{aligned} \tag{5.4}$$

where:

$$\begin{aligned} \mathbb{C} &= \{c_i | i \in N\} \\ \mathbb{O} &= \{o_i | i \in N\} \end{aligned}$$

$$\tau_i = \prod_{j=1}^{i-1}(1 - o_j) \tag{5.4.1}$$