Statistical Formulation of Region-Based Segmentation (2007)

$P(\Omega_1, \dots \Omega_N)$: optimal partition of the image plane Ω (切割)

By maximizing a posteriori probability $p(P(\Omega_1, \dots \Omega_N) | I)$ for a given image I.

$$\because \quad p(P(\Omega_1, \dots \Omega_N), I) = p(P(\Omega_1, \dots \Omega_N) | I)\, p(I) = p(I | P(\Omega_1, \dots \Omega_N))\, p(P(\Omega_1, \dots \Omega_N))$$

$$\therefore \quad p(P(\Omega_1, \dots \Omega_N) | I) \propto \underbrace{p(I | P(\Omega_1, \dots \Omega_N))}_{\text{Image Based cues: observation given a model state.}} \underbrace{p(P(\Omega_1, \dots \Omega_N))}_{\text{geometric properties of the partition.}}$$

The image partition to be composed of N regions without correlation between labellings

$$p(I | P(\Omega_1, \dots \Omega_N)) = p(I | \{\Omega_1, \dots \Omega_N\}) = \prod_{i=1}^{N} p(I | \Omega_i)$$

($p(I|\Omega_i)$: the pro of observing an image I when Ω_i is a region of interest.)

Assume that the observation I consists of a set of feature values $f(x)$ associated with each image location. (image intensity, pixel color, gradient ....)

We make the assumption that the value of f at different locations of the same region can be modeled as independent and identically distributed realizations of the same random process.

Let P_i be the PDF of this random process in Ω_i.

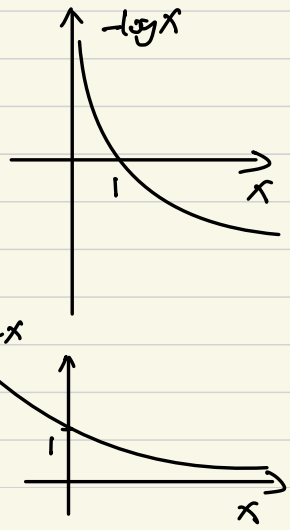
$$\therefore \quad p(I | P(\Omega_1, \dots \Omega_N)) = \prod_{i=1}^{N} \prod_{x \in \Omega_i} (P_i(\underbrace{f(x)}_{\text{特征}}))^{dx},$$

dx is the bin volume.

★ Maximization of the $p(P(\Omega_1, \dots \Omega_N) | I)$ is equivalent to minimizing its negative logarithm

$$\therefore \quad -\log\left(\prod_{i=1}^{N} \prod_{x \in \Omega_i} (P_i(f(x)))^{dx}\right) \qquad \log a^x$$

$$= -\sum_{i=1}^{N} \int_{\Omega_i} \log(P_i(f(x)))\, dx$$



If $p(P(\Omega_1, \dots \Omega_N)) \propto e^{-\nu |C|}$, $\nu > 0$, favors a short length $|C|$ of partition boundary.

$$\therefore \quad E(\{\Omega_1, \ldots \Omega_N\}) = -\sum_i \int_{\Omega_i} \log p_i(f(x))\,dx + \nu|C|$$

$$\therefore \quad \min(E(\{\Omega_i\})) = \min\left(-\sum_i \int_{\Omega_i} p(f(x))\,dx\right)$$

Two-Pase Level Set

Assume that a partioning of the domain Ω such that each pixel is ascribed to one of two possible phases.

$$\therefore E(\phi) = \int_\Omega -H(\phi)\log p(f) - (1-H(\phi))\log p(f) + \nu|\nabla H(\phi)|dx$$

where $H(\phi) = \begin{cases} 1 & \text{if } \phi \geq 0, \\ 0 & 0 \text{ else.} \end{cases}$

Robust Real-Time Visual Tracking Using Pixel-Wise Posteriors (2008)

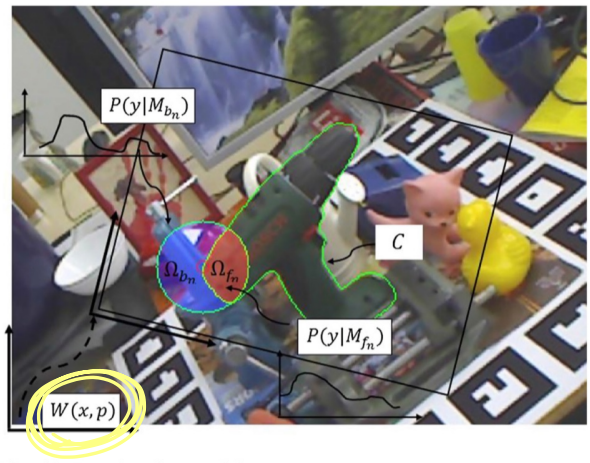
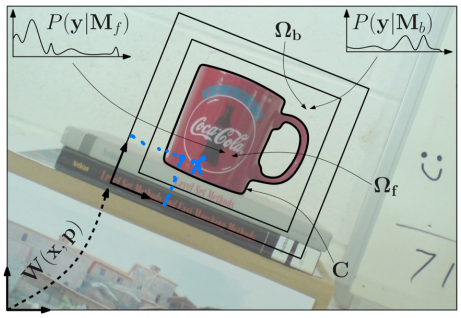


Fig. 12 Local region model

$W(x,p)$: take a pixel location x in the object frame and warps it into the image frame according to parameters p.



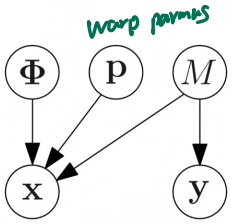


Fig. 1. (Left): Representation of object showing: the contour $\mathbf{C}$, the set of foreground pixels $\mathbf{\Omega}_f$, the set of background pixels $\mathbf{\Omega}_b$, the foreground model $P(\mathbf{y}|M_f)$, the background model $P(\mathbf{y}|M_b)$ and the warp $\mathbf{W}(\mathbf{x},\mathbf{p})$; (Right): Graphical representation of our generative model representing the image as a bag-of-pixels

- $\mathbf{x}$: A pixel location in the object coordinate frame.
- $\mathbf{y}$: A pixel value (in our experiments this is a YUV value).
- $\mathbf{I}$: Image.
- $\mathbf{W}(\mathbf{x},\mathbf{p})$: Warp with parameters $\mathbf{p}$.
- $M = \{M_f, M_b\}$: Model parameter either foreground or background.
- $P(\mathbf{y}|M_f)$: Foreground model over pixel values $\mathbf{y}$.
- $P(\mathbf{y}|M_b)$: Background model over pixel values $\mathbf{y}$.
- $\mathbf{C}$: The contour that segments the foreground from background.
- $\mathbf{\Phi}(\mathbf{x})$: Shape kernel (in our case the level-set embedding function).
- $\mathbf{\Omega} = \{\mathbf{\Omega}_f, \mathbf{\Omega}_b\}$: Pixels in the object frame $[\{\mathbf{x}_0, \mathbf{y}_0\}, \ldots, \{\mathbf{x}_N, \mathbf{y}_N\}]$, which is partitioned into foreground pixels $\mathbf{\Omega}_f$ and background pixels $\mathbf{\Omega}_b$.
- $H_\epsilon(z)$: Smoothed Heaviside step function.
- $\delta_\epsilon(z)$: Smoothed Dirac delta function.

The joint distribution for ce single pixel given the model:

$$P(x,y,\Phi,p,M) = P(x,y|\Phi,p,M)\,P(\Phi,p,M)$$

(x,y: color, counter, warp, model)

$$= P(x|\Phi,p,M)\,P(y|\Phi,p,M)\,P(\Phi)\,P(p)\,P(M)$$

$$= P(x|\Phi,p,M)\,P(y|M)\,P(\Phi)\,P(p)\,P(M)$$

$$= P(x|\Phi,p,M)\,P(y,M)\,P(\Phi)\,P(p)$$

$$\therefore P(x,\Phi,p,M|y)\,P(y) = P(x|\Phi,p,M)\,P(M|y)\,P(y)\,P(\Phi)\,P(p)$$

$$\therefore P(x,\Phi,p,M|y) = P(x|\Phi,p,M)\,P(M|y)\,P(\Phi)\,P(p)$$

$$= P(\Phi)P(p)\sum_{j=f,b} P(x|\Phi,p,M_j)\,P(M_j|y)$$

pixel-wise posterior of the model M given a pixel values y.

$$P(M_j|y) = \frac{P(y|M_j)\,P(M_j)}{P(y)} = \frac{P(y|M_j)\,P(M_j)}{\sum_{i=f,b} P(y|M_i)\,P(M_i)},\ j=\{f,b\}$$

$$= P(x,\Phi,p|y)$$

$$P(x,\Phi,p|y) = P(\Phi,p|x,y)\,P(x|y) = P(\Phi,p|x,y)\,P(x)$$

$$\Rightarrow P(\Phi,p|x,y) = P(x,\Phi,p|y)\,\frac{1}{P(x)}$$

The prob of the shape and the location p given pixel $\{x,y\}$

$$= P(\Phi)P(p)\,\frac{1}{P(x)}\sum_{j=f,b} P(x|\Phi,p,M_j)\,P(M_j|y)$$

$\frac{1}{P(x)}$: it can be drop as it is constant.

$P(x|\Phi,p,y)$

The pixel wise posteriors:

$$P(\Phi, p|\Omega) \propto P(\Phi)P(p)\prod_{i=1}^{N}\{\sum_{j=f,b} P(x_i|\Phi, p, M_j)P(M_j|y_i)\}$$

(N: number of pixels; $P(\Phi)$: Φ is known from model; $P(p)$: constant for all pixels)

$$\therefore P(\Phi, p|\Omega) \propto \prod_{i=1}^{N}\{\sum_{j=f,b} P(x_i|\Phi, p, M_j)P(M_j|y_i)\}$$

$$P(x_i|\Phi, p, M_f) = \begin{cases} \frac{He(\Phi(x_i))}{n_f}, & j=f \\ \frac{1-He(\Phi(x_i))}{n_b}, & j=b \end{cases}$$

如果 x_i 是在 foreground，概率为 $\frac{1}{n_f}$，否则为 0

如果 x_i 是在 background，概率为 $\frac{1}{n_b}$，否则为 0.

$$P(M_j|y) = \frac{P(y|M_j)P(M_j)}{\sum_{\{i=f,b\}}P(y|M_i)P(M_i)}$$

and $$P(M_j) = \begin{cases} \frac{n_f}{n}, & j=f \\ \frac{n_b}{n}, & j=b \end{cases}$$, $n = n_f + n_b = K$

$$n_f = \sum_{i=1}^{K} He(\Phi(x_i))$$

$$n_b = \sum_{i=1}^{K}(1-He(\Phi(x_i)))$$

$$\therefore P(\Phi, p|\Omega) \propto \prod_{i=1}^{N}\{\sum_{j=f,b} P(x_i|\Phi, p, M_j)P(M_j|y_i)\}$$

$$= \prod_{i=1}^{N}\{P(x_i|\Phi, p, M_f)P(M_f|y_i) + P(x_i|\Phi, p, M_f)P(M_b|y_i)\}$$

$$= \prod_{i=1}^{N}\{\frac{He(\Phi(x_i))}{n_f}\frac{P(y_i|M_f)\frac{n_f}{n}}{P(y_i|M_f)\frac{n_f}{n}+P(y|M_b)\frac{n_b}{n}} + \frac{1-He(\Phi(x_i))}{n_b}\frac{P(y_i|M_b)\frac{n_b}{n}}{P(y_i|M_f)\frac{n_f}{n}+P(y|M_b)\frac{n_b}{n}}\}$$

$$= \prod_{i=1}^{N}\{\frac{He(\Phi(x_i))P(y_i|M_f)}{P(y_i|M_f)n_f + P(y|M_b)n_b} + \frac{[1-He(\Phi(x_i))]P(y_i|M_b)}{P(y_i|M_f)n_f + P(y|M_b)n_b}\}$$

$$= \prod_{i=1}^{N}\{p_f + p_b\}$$

2D-3D Pose Estimation of Heterogeneous Objects Using a Region Based Approach (2016)

It is common in region-based segmentation for a closed curve to be evolved such that discrepancy (相差) between the statistics of the foreground and those of the background region is maximised:

$$E=\int_{\Omega_f} r_f(I(x), C)\,d\Omega + \int_{\Omega_b} r_b(I(x), C)\,d\Omega$$

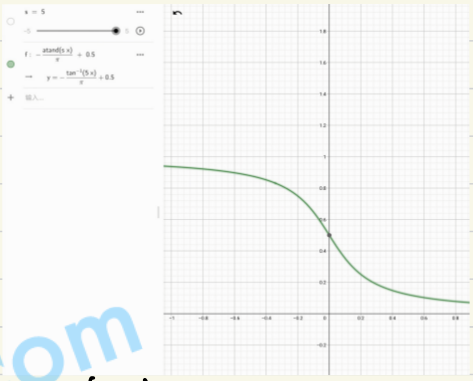
Ω_f: foreground of image $\subset \mathbb{R}^2$, $[\{x_0,y_0\},\dots,\{x_N,y_N\}]$, N is number of pixels.
Ω_b: background
$X=[x,y]$
$I(x)=y$, pixel value
$C=\{(x,y)\in\mathbb{R}^2 \mid \Phi(x)=0\}$: the contour, the zero level-set of the function $\Phi(x)$

where r_f and r_b are two monotonically decreasing functions, measuring the matching quality of image pixels with respect to the foreground and background models.

① Rewrite the energy function:

$$E(\Phi)=\int_{\Omega} He(\Phi)\, r_f(x) + (1-He(\Phi))\, r_b(x)\,d\Omega$$

$$\Phi(x)=\begin{cases} -d(x), & \forall x\in\Omega_f \\ d(x), & \forall x\in\Omega_b \end{cases}, \qquad d(x)=\min_{x_c\in C}|x-x_c|$$



He is smoothed Heaviside step,

$$He(\Phi(x)) = -\frac{1}{\pi}\arctan(s\cdot\Phi(x))+\frac{1}{2},$$

with s determing the pitch of the smoothed transition

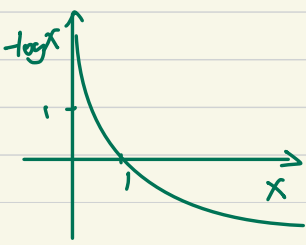
r_f and r_b are given by the likelihood of a pixel property (such as color) under a given model, i.e. $r(x)=P(y|M)$, with $M\in\{M_f, M_b\}$.
PDF of a pixel value y belonging to the foreground or background.

② In contrast, assuming pixel-wise independence, and replacing integration with summation, the energy given by the negative log (posterior) probability of the shape of the contour (Φ) given the image data:

$$P(\Phi|I)\propto\prod_{x\in\Omega}(He(\Phi(x))P_f(y)+(1-He(\Phi(x)))P_b(y))$$

$$\therefore\ E(\Phi)=-\log(P(\Phi|I))$$

$$=-\sum_{x\in\Omega}\log(He(\Phi)P_f+(1-He(\Phi))P_b)$$



$y = I(x)$. pixel value

$$\because \quad P(y) = P(y|M_f)P(M_f) + P(y|M_b)P(M_b)$$

$$\therefore \quad P(M_j|y)P(y) = P(y|M_j)P(M_j), \qquad j = \{f, b\}.$$

$$\Longrightarrow \quad P(M_j|y) = \frac{P(y|M_j)P(M_j)}{\underbrace{\sum_{i=\{f,b\}} P(y|M_i)P(M_i)}_{P(y)}}$$

Let $n_f = \sum_{i=1}^{N} H_e(\Phi(x_i))$, $n_b = \sum_{i=1}^{N} (1 - H(\Phi(x_i)))$, $n = n_f + n_b$

n_f: foreground pixels number; n_b: background pixels number; n: all pixels number

$$\therefore \quad P(M_f) = \frac{n_f}{n}, \qquad P(M_b) = \frac{n_b}{n}$$

$$\therefore \quad P_j(y) = \frac{P(M_j|y)}{n_j} = \frac{1}{n_j} \frac{P(y|M_j)\frac{n_j}{n}}{\sum_{i=\{f,b\}} P(y|M_i)\frac{n_i}{n}} = \frac{P(y|M_j)}{P(y|M_f)n_f + P(y|M_b)n_b}$$

The $P(y|M_j)$ are estimated by calculating the histograms (256 bins) of each color space and smooth using a Gaussian kernel. For simplicity, we assume the RGB channels are independent.

$$\therefore P(y_{RGB}|M) = P(y_R|M)P(y_G|M)P(y_B|M)$$

1. Initialize pose parameters. $\lambda = \lambda_0$
2. For each iteration:
   (a) Apply 3D transformation to object.
   (b) Project 3D model on to image plane.
   (c) For each local region:
      (i) Estimate local region statistics.
      (ii) Calculate local energy gradient with respect to pose parameters, ∇E_n.
   (d) Fuse local region gradients, $\nabla E = f(\nabla E_n)$.
   (e) Find optimal step size, s.
   (f) Update pose parameters $\lambda = \lambda - s\nabla E$.

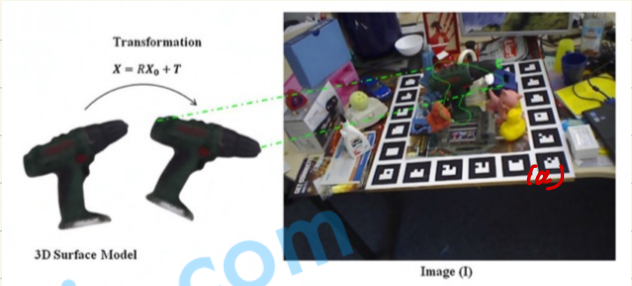


Fig. 5 Driller object projection onto the image plane

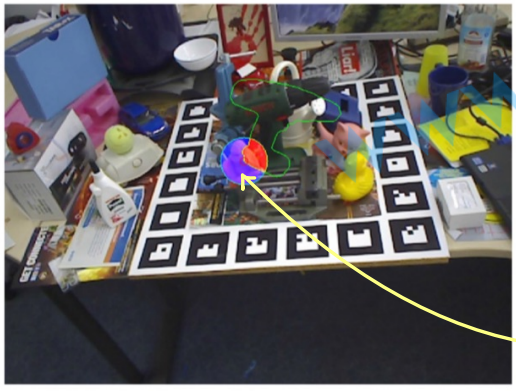
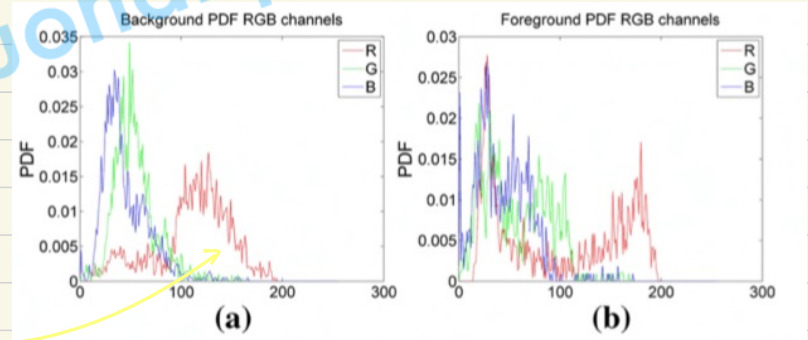


Fig. 8 Example of a local region extraction, divided into the local foreground (*red*) and local background (*blue*) (Color figure online)



Let $\lambda=[\lambda_1,\lambda_2,\lambda_3,\lambda_4,\lambda_5,\lambda_6]^T$ the pose parameters.

$t=[\lambda_1,\lambda_2,\lambda_3]^T$

$R=\exp(\Lambda)$, $\quad \Lambda=\begin{bmatrix} 0 & -\lambda_6 & \lambda_5 \\ \lambda_6 & 0 & -\lambda_4 \\ -\lambda_5 & \lambda_4 & 0 \end{bmatrix}$

$$\therefore \frac{\partial E}{\partial \lambda_i} = \frac{\partial\left(-\sum_{x\in\Omega} \log\left(He(\mathcal{D})P_f + (1-He(\mathcal{D}))P_b\right)\right)}{\partial \lambda_i}$$

$\frac{d\ln x}{dx} = \frac{1}{x}$

$$= -\sum_{x\in\Omega} \frac{1}{He(\mathcal{D})P_f + (1-He(\mathcal{D}))P_b} \cdot (P_f - P_b) \cdot \frac{\partial He(\mathcal{D})}{\partial \lambda_i}$$

$He(\mathcal{D})(P_f-P_b)+P_b$

relay on statistical properties est weight applied to the geo differentials.

geo differentials of the object with respect to the pose parameters.

$$\because \frac{\partial He(\mathcal{D})}{\partial \lambda_i} = \frac{\partial He(\mathcal{D})}{\partial \mathcal{D}}\left[\frac{\partial \mathcal{D}}{\partial u}\frac{\partial u}{\partial \lambda_i} + \frac{\partial \mathcal{D}}{\partial v}\frac{\partial v}{\partial \lambda_i}\right] = \delta_e(\mathcal{D})\left[\frac{\partial \mathcal{D}}{\partial u}\frac{\partial u}{\partial \lambda_i} + \frac{\partial \mathcal{D}}{\partial v}\frac{\partial v}{\partial \lambda_i}\right]$$

$\mathcal{D}(u,v)$ image pixel

$$\because \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \frac{X}{Z} f_x + u_0 \\ \frac{Y}{Z} f_y + v_0 \end{bmatrix}$$

$$\therefore \frac{\partial u}{\partial \lambda_i} = f_x \frac{\partial (X/Z)}{\partial \lambda_i} = f_x\left(\frac{1}{Z}\frac{\partial X}{\partial \lambda_i} - X\frac{1}{Z^2}\frac{\partial Z}{\partial \lambda_i}\right)$$

$$\therefore \frac{\partial v}{\partial \lambda_i} = f_y\left(\frac{1}{Z}\frac{\partial Y}{\partial \lambda_i} - Y\frac{1}{Z^2}\frac{\partial Z}{\partial \lambda_i}\right)$$

Local Region Based Pose Estimation

Defined a mask function $B_n(x_i, x_c)$ which centered at x_c, with a radius d:

$$B_n(x_i, x_c) = \begin{cases} 1 & |x_i - x_c| < d \\ 0 & \text{else} \end{cases}$$

$$\frac{\partial E}{\partial \lambda_i} = -\sum_{x \in \Omega_n} \frac{(P_f - P_b)}{H_e(\Phi) P_f + (1 - H_e(\Phi)) P_b} \cdot \frac{\partial H_e(\Phi)}{\partial \lambda_i}$$

∵ $H_e(\Phi)$ value only change on the counter! So we define a local regions around every point along the object's contour.

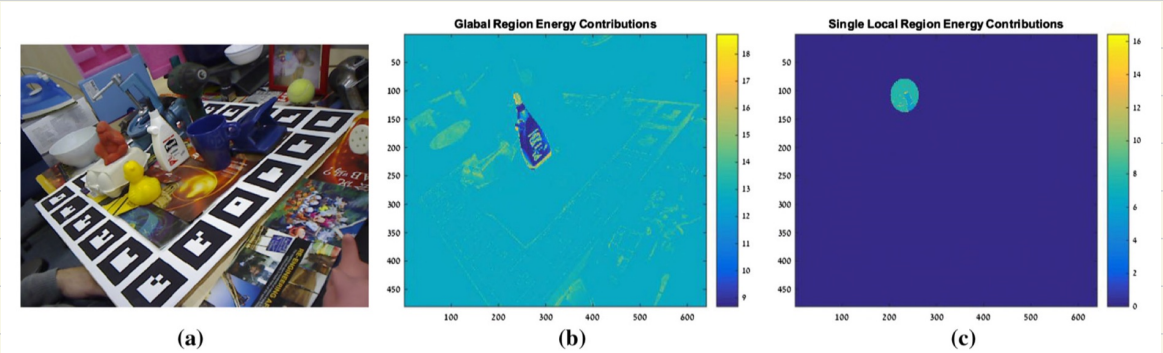


Fig. 13 Energy function evaluation over heterogeneous object

Using large sized regions, the region statistics will be more robust to the initialization of the object's pose. Change in the objects pose will have a lower impact on the statistical models.
Using small sized regions, the ability to capture variations in the region's statistics increases.

A small object or a cluttered (杂乱的) background require a small radius to correctly catch the variation between regions, whereas for a large object and a slowly varying background, a large radius is preferred.

《Real-Time Monocular Pose Estimation of 3D Objects Using Temporally Consistent》(2017)

Result evaluate : $\frac{E}{|\Omega|} > t$, $t \in [0.5, 0.6]$, lost

$$E = -\sum_{x \in \Omega} \log\left(He(\Phi(x))\bar{P}_f(x,y) + (1 - He(\Phi(x)))\bar{P}_b(x,y)\right)$$

we want get the min of E

$\Omega \subset \mathbb{R}^2$ being the image domain, $\Omega \to \{0, 1\}$ (0: background, 1: foreground)

$|\Omega|$: foreground region pixel number

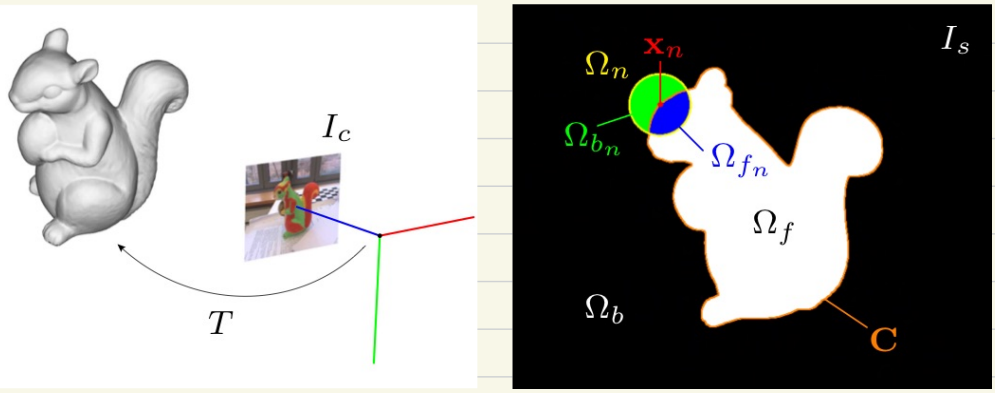


Figure 2. Overview of our pose estimation setting. Left: The object pose T relative to a camera based on color image I_c and a 3D model of the object. Right: Silhouette I_s generated by projecting the surface model into the image plane using an estimated pose T.

tcle-histograms :

那这里又存在一个local采样中心点的时序一致的问题，就是说前后帧的匹配问题，必须得找到 $\mathbf{x}_i(t_l) \leftrightarrow \mathbf{x}_i(t_{l-1})$ 间的匹配关系。Real-Time Monocular Pose Estimation of 3D Objects using Temporally Consistent Local Color Histograms 提出 temporally consistent local color histograms（tclc-histograms）试图解决这一问题。

大致思想是，由于三维顶点是唯一且有序的，其投影到二维平面有像素点与之对应。根据不同角度和视口，每个三维模型顶点 $\mathbf{X}_i$ （都有可能投影成为二维图像边缘上的像素点，也就意味着）都可以有一个关联着的所谓的 local foreground & background histograms，文中也称呼为 tclc-histogram。

那也就是说齐次坐标的 $\tilde{\mathbf{X}}_i$ 其在二维投影平面的前后帧像素关系：

$$\pi(K(T(t_l)\tilde{X}_i)_{3\times1}) \leftrightarrow \pi(K(T(t_{l-1})\tilde{X}_i)_{3\times1})$$

对于第一次投影到二维轮廓线上的三维顶点，都在当前帧，以该点为圆心进行一次tclc-histogram初始化。而对于已保留有前帧histogram信息的点，即说明该像素 $\mathbf{x}_i = \pi(K(T\tilde{X}_i)_{3\times1})$ 其前后帧都落在轮廓线上，即 $\mathbf{x}_i(t_l), \mathbf{x}_i(t_{l-1}) \in \mathbf{C}$ ，其更新方法就是前文提到的：

$$P(\mathbf{y}|M_{f_i}) = (1-\alpha_f)P^{t_{l-1}}(\mathbf{y}|M_{f_i}) + \alpha_f P^{t_l}(\mathbf{y}|M_{f_i}),$$
$$P(\mathbf{y}|M_{b_i}) = (1-\alpha_b)P^{t_{l-1}}(\mathbf{y}|M_{b_i}) + \alpha_b P^{t_l}(\mathbf{y}|M_{b_i}),$$

Pose Detection

In order to cover different scales, each of these 48 base orientations is used to generate a template at a close, an intermediate and a far distance to the camera, resulting in overall 144 base templates.

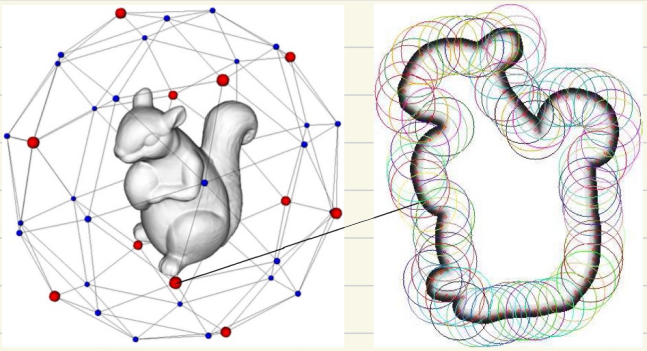


Figure 5. The template views used for pose detection. Left: A subdivided icosahedron generates the outer plane rotation of the template views. Red dots indicate vertices corresponding to the base templates, blue dots indicate those provided by subdivision used for the neighboring templates. Right: An example base template visualized by the corresponding $H_e(\Phi)$ (grey pixels) with the local histogram regions depicted (colored circles) along the contour.

Real-Time 3D Model Tracking in Color and Depth on a Single CPU Core (2017)

3.4. Approximating for real-time tracking

Computing the SDF from Eq. 3.1 has already three costly steps. We need a silhouette rendering Ω_f of the current model pose, an extraction of the contour C and lastly, a subsequent distance transform embedding ϕ. While [29] perform GPU rendering and couple computation of the SDF and its gradient in the same pass to be faster, [17] perform hierarchical ray-tracing on the CPU and extract the contour via Scharr operators. We make two key observations:

1. Only the actual contour points are required
2. Neighboring points provide superfluous information because of similar curvature

We thus propose a cheap yet very effective approximation of the model render space that avoids both online rendering and contour extraction. In an offline stage, we equidistantly sample viewpoints V_i on a unit sphere around the object model, render from each and extract the 3D contour points to store view-dependent **sparse** 3D sampling sets in **local object space** (see Figure 4). Since we will utilize these points in 3D space, we neither need to sample in scale nor for different inplane rotations. Finally, we store for each contour point its 2D gradient orientation and sample a set of interior surface points with their normals (see Figure 5).

In a naive approach, all involved terms from Eq. 7 would be computed densely, *i.e.* $\forall x \in \Omega$, which is prohibitively costly for real-time scenarios. The related work evaluates the energy only in a narrow band around the contour since the residuals decay quickly when leaving the interface. We therefore propose to compute Eq. 8 in a narrow band along a sparse set of selected contour points where we compute ϕ along rays. Each projected contour point shoots a positive and negative ray perpendicularly to the contour, i.e along its normal. Building on that, we introduce the idea of ray integration for 3D contour points such that we do not create pixel-wise but **ray-wise** Jacobians which leads to a smaller reduction step and a better conditioning of the normal system in Eq. 9 than [17] and their approach.

To formalize, we have a model pose $[R, t]$ during tracking and avoid rendering by computing the camera position in object space $O := -R^\intercal t$. We normalize to unit length and find the closest viewpoint V^* quickly via dot products:

$$V^* := \underset{V_i}{\operatorname{argmax}} \langle\, V_i,\ O/||O||\, \rangle. \tag{16}$$

Each local 3D sample point of the contour X_i from V^* is then transformed and projected to a 2D contour sample point $x_i = \pi(RX_i + t)$ which is then used to shoot rays into the object interior and into the opposite direction.

To get the orientation of each ray, we cannot rely anymore on the value during pre-rendering since the current model pose might have an inplane rotation not accounted for. Given a contour point with 2D rotation angle θ during pre-rendering, we could embed it into 3D space via $v = (\cos\theta, \sin\theta, 0)$ and later multiply it with the current model rotation R. Although this works in practice, the projection of $R \cdot v$ onto the image plane can be off at times. We thus propose a new approximation of the inplane rotation where we seek to decompose $R = R_{inplane} \cdot R_{canonical}$ s.t. one part describes a general rotation around the object center in a canonical frame and the other a rotation around the view direction of the camera (*i.e.* inplane) . Although ill-posed in general, we exploit our knowledge about the closest viewpoint by assuming $R_{canonical} \approx R_{V^*}$ and propose to approximate a rotation $\tilde{R}$ on the xy-plane via

$$\tilde{R} := R \cdot R_{V^*}^\intercal. \tag{17}$$

We then extract the angle $\theta = acos(\tilde{R}_{1,1})$ via the first element. With larger viewpoint deviation $||V^* - \frac{O}{||O||}||$, this approximation worsens but our sphere sampling is dense enough to alleviate this in practice. We re-orient each contour gradient $\tilde{g}_i := (g_i + \theta) \mod 2\pi$ and shoot rays to compute the residuals and $\frac{\partial H_\phi}{\partial \phi}$ from Eq. 7 (see Figure 5 to compare the orientations and the bottom row in Figure 2 for the SDF rays).

The final missing building block is the derivative of the SDF $\frac{\partial \phi}{\partial x}$ which cannot be computed numerically since we are missing dense information. We thus compute it geometrically, similar to [17]. Whereas their computation is exact when assuming local planarity by projections onto the principal ray, our approach is faster while incurring a small error which is negligible in practice. Given a ray $r = (r_x, r_y)$ from contour point $p = (p_x, p_y)$ we compute the horizontal derivative at $\phi(p_x + r_x, p_y + r_y)$ as central difference

$$\frac{||(p_x + r_x + 1, p_y + r_y)|| - ||(p_x + r_x - 1, p_y + r_y)||}{2}.$$

The vertical derivative is computed analogously. Like the related work, we perform all computations on three pyramid levels in a coarse-to-fine manner and shoot the rays in a band of 8 steps on each level. Since we shoot two rays per

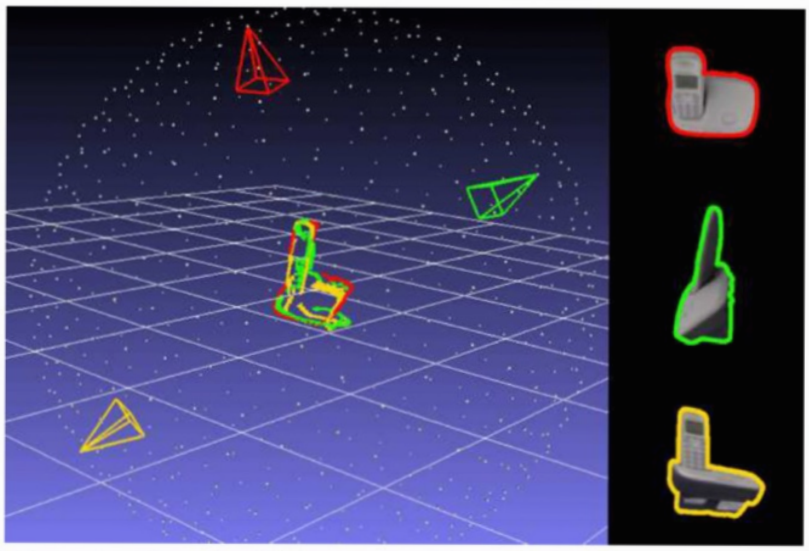


Figure 4. Object-local 3D contour points visualized for three viewpoints on the unit sphere. Each view captures a different contour which is used during tracking to circumvent costly renderings.

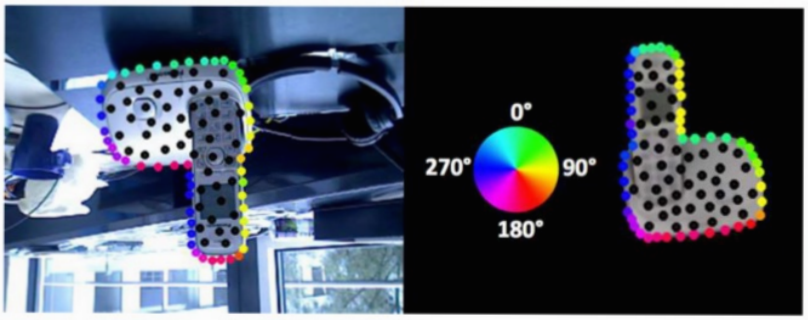


Figure 5. Current tracking and closest pre-rendered viewpoint augmented with contour and interior sampling points. The hue represents the normal orientation for each contour point. Note how we rotate the orientation of each contour point by our approximation of the inplane rotation such that the SDF computation is proper.

A Region-Based Gauss-Newton Approach to Real-Time Monocular Multiple Object Tracking (2019)

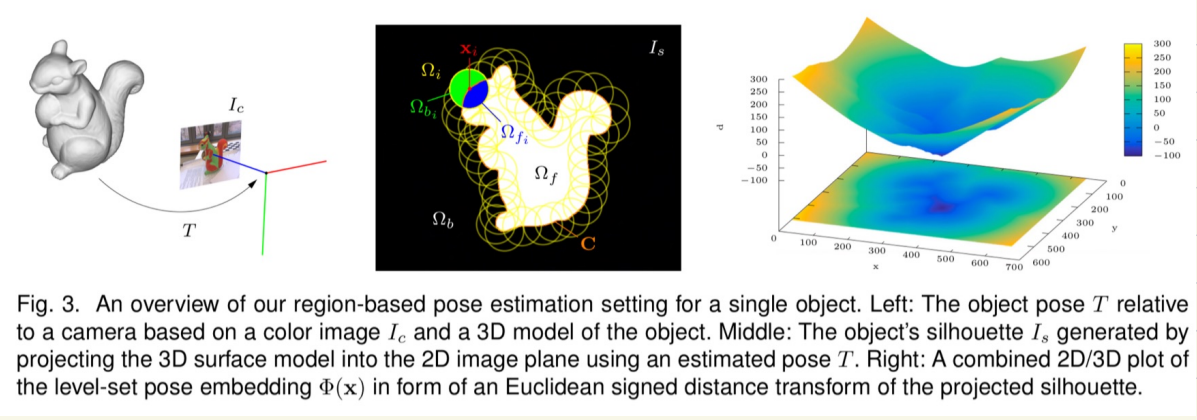


Fig. 3. An overview of our region-based pose estimation setting for a single object. Left: The object pose T relative to a camera based on a color image I_c and a 3D model of the object. Middle: The object's silhouette I_s generated by projecting the 3D surface model into the 2D image plane using an estimated pose T. Right: A combined 2D/3D plot of the level-set pose embedding $\Phi(\mathbf{x})$ in form of an Euclidean signed distance transform of the projected silhouette.

In 2D-3D Pose Estimation of Heterogeneous Objects Using a Region Based Approach (2016)

The local region centers x_i were calculated as arbitrary sets of pixel locations along C for each image. Thus, this approach in general does not allow to establish correspondences of centers across multiple frames ($x_i(t_t) \leftrightarrow x_i(t_{t-1})$) which is required in order to update the respective histograms.

Tdc-histograms: projecting all model vertices into image plane, $x_i = \pi(K(T\tilde{X}_i)_{3\times1})$ and selecting the subset of all $x_i \in C$. center correspondences between frames:

$$\pi_t(K(T(t_t)\tilde{X}_i)_{3\times1}) \leftrightarrow \pi_{t-1}(K(T(t_{t-1})\tilde{X}_i)_{3\times1})$$

This ensures to keep the individual histograms temporally consistent.

① If a vertex projects onto the contour for the first time, its corresponding histograms are inited from the local region around its center in current frame.

② Otherwise:

$$P(y|M_{fi}) = (1-\alpha_f)P^{t_{t-1}}(y|M_{fi}) + \alpha_f P^{t_t}(y|M_{fi})$$
$$P(y|M_{bi}) = (1-\alpha_b)P^{t_{t-1}}(y|M_{bi}) + \alpha_b P^{t_t}(y|M_{bi})$$

More robust results can be obtained by computing the average posteriors from all local histograms instead:

$$\bar{P}_f(x) = \frac{1}{\sum_{i=1}^{n} B_i(x)} \sum_{i=1}^{n} P_{fi}(x)B_i(x), \quad B_i(x) = \begin{cases} 1, & \forall x \in \Omega_i \\ 0, & \forall x \notin \Omega_i \end{cases}$$

$$\bar{P}_b(x) = \frac{1}{\sum_{i=1}^{n} B_i(x)} \sum_{i=1}^{n} P_{bi}(x)B_i(x)$$

$$\therefore \quad E(\xi) = -\sum_{x\in\Omega} \log\left(He(\mathcal{D}(x(\xi)))\right)\bar{P}_f(x) + \left(1-He(\mathcal{D}(x(\xi)))\right)\bar{P}_b(x).$$

Gauss-Newton Strategy

$$E(\xi) = \sum_{x\in\Omega} F(x,\xi), \qquad F(x,\xi) = -\log\left(He(\mathcal{D}(x,\xi))\right)\bar{P}_f(x) + \left(1-He(\mathcal{D}(x(\xi)))\right)\bar{P}_b(x)$$

rewrite it as a nonlinear weighted least-squares problem of the form:

$$E(\xi) = \frac{1}{2}\sum_{x\in\Omega}\frac{1}{F(x,\xi)}F^2(x,\xi) = \frac{1}{2}\sum_{x\in\Omega}\psi(x)F^2(x,\xi)$$

fixed weights $\psi(x)$ by means of Gauss-Newton opt and alternatively using the refined pose for updating the weights $\psi(x)$.

$$\therefore \frac{\partial E(\xi)}{\partial \xi} = \frac{1}{2}\sum_{x\in\Omega}\psi(x)\frac{\partial F^2(x,\xi)}{\partial \xi} = \sum_{x\in\Omega}\psi(x)F\frac{\partial F}{\partial \xi} = \sum_{x\in\Omega}\frac{\partial F}{\partial \xi}$$

with $\psi(x)F(x,\xi) = 1$.

$$\therefore \frac{\partial^2 E(\xi)}{\partial \xi^2} = \sum_{x\in\Omega}\psi(x)\left(\left(\frac{\partial F}{\partial \xi}\right)^T\frac{\partial F}{\partial \xi} + F\frac{\partial^2 F}{\partial \xi^2}\right)$$

Let $\frac{\partial F(x,\xi)}{\partial \xi} = J$

$$\because \quad E(\xi+\Delta\xi) \approx E(\xi) + J\Delta\xi$$

$$\therefore \frac{1}{2}\|E(\xi)+J\Delta\xi\|^2 = \frac{1}{2}\left(\|E(\xi)\|_2^2 + 2E(\xi)^T\sum_{x\in\Omega}J\Delta\xi + \frac{1}{2}\Delta\xi^T\sum_{x\in\Omega}\psi(x)\Delta\xi^T\underbrace{J^TJ}_{H}\Delta\xi\right)$$

Let $\frac{\partial\left(\frac{1}{2}\|E(\xi)+J\Delta\xi\|^2\right)}{\partial\Delta\xi} = 0$

$$\therefore \quad J^TJ\Delta\xi = -J^TE(\xi)$$

$$\therefore \quad \Delta\xi = (J^TJ)^{-1}J^TE(\xi)$$

4.1 Rendering Engine

We use the standard rasterization pipeline of OpenGL in order obtain the silhouette masks I_s. Since we want to process the rendered images on the CPU, we perform offscreen rendering into a *FrameBufferObject*, which is afterwards downloaded to host memory. To generate synthetic views that match the real images, the intrinsic parameters (2) of the camera need to be included. For this, we model the transformation from 3D model coordinates $\mathbf{X}$ to homogeneous coordinates within the canonical view volume of OpenGL as $\tilde{\mathbf{V}} = P(K)LT\tilde{\mathbf{X}} \in \mathbb{R}^4$. Here, L is the so-called *look-at* matrix

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4}, \tag{36}$$

that aligns the principal axes of the real cameras coordinate frame with those of the virtual OpenGL camera and $P(K)$ is a homogeneous projection matrix

$$P(K) = \begin{bmatrix} \frac{2f_x}{w} & 0 & 1-\frac{2c_x}{w} & 0 \\ 0 & -\frac{2f_y}{h} & 1-\frac{2c_y}{h} & 0 \\ 0 & 0 & -\frac{Z_f+Z_n}{Z_f-Z_n} & -\frac{2Z_fZ_n}{Z_f-Z_n} \\ 0 & 0 & -1 & 0 \end{bmatrix} \in \mathbb{R}^{4\times 4}, \tag{37}$$

with respect to the camera matrix K. The scalars w, h are the width and height of the real image I_c and Z_n, Z_f are the near- and far-plane of the view frustum described by $P(K)$.

In case of tracking multiple objects, all 3D models are rendered in the same scene. Each mesh is rendered with a constant and unique color that corresponds to its model index j. This allows to separate the individual foreground regions and identify their contours $\mathbf{C}^j$ as required for computing the different level-sets. Here, mutual occlusions are natively handled by the OpenGL Z-Buffer.

As seen in (34), the derivatives used for pose optimization involve the coordinates of the 3D surface point $\mathbf{X}'$ in the camera's frame of reference, corresponding to each pixel $\mathbf{x}$. In addition to the silhouette mask, we therefore also download the Z-buffer into a per pixel depth map $I_d : \Omega \to [0,1] \subset \mathbb{R}$. Given I_d, the required coordinates are efficiently determined via backprojection as $\mathbf{X}'(\mathbf{x}, I_d) = D(\mathbf{x}, I_d)K^{-1}\tilde{\mathbf{x}}$, with

$$D(\mathbf{x}, I_d) = \frac{Z_nZ_f}{Z_f - I_d(\mathbf{x})(Z_f - Z_n)}, \quad \forall \mathbf{x} \in \Omega_f, \tag{38}$$

where $\tilde{\mathbf{x}} = [x, y, 1]^\top$ is the homogeneous representation of an image point $\mathbf{x} = [x, y]^\top$.

In [26] it has been shown that it is beneficial not only to consider the points on the surface closest to the camera but also the most distant ones (on the backside of the object) for pose optimization. In order to obtain the respective coordinates for each pixel, we compute an additional reverse depth map I_d^r, for which we simply invert the OpenGL depth check used to compute the corresponding Z-buffer (see Figure 6). Given I_d^r, the farthest surface point $\mathbf{X}'(\mathbf{x}, I_d^r)$ corresponding to a pixel $\mathbf{x}$ is also recovered as $\mathbf{X}'(\mathbf{x}, I_d^r) = D(\mathbf{x}, I_d^r)K^{-1}\tilde{\mathbf{x}}$.

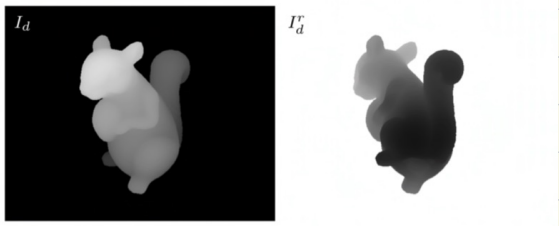


Fig. 6. The two depth map types used within our approach, where brighter pixels are closer to the camera. Left: The usual depth map I_d corresponding to the closest surface points. Right: The reverse depth map I_d^r corresponding to the most distant surface points.

Pixel-Wise Weighted Region-Based 3D Object Tracking using Contour Constraints (2021)

By projecting 3D model into image plane with a given pose, we obtain the silhouette mask I s , in which the projected contour segments the image into the foreground region Ωf and the background region Ω b . At each projected contour point m i , the search line l i is sampled along the direction vector n i which is perpendicular to the projected contour. The j -th sampling pixel of the search line l i is denoted by x ij . s i is the object contour point. The object pose that transforms a 3D model point in the object coordinate to the camera coordinate is represented by a 4 × 4 homogeneous matrix

$$^{cam}_{model}T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$$

$$x = \pi(K(T^{model} X))$$

ΔT is parameterized by $p = [w_1\ w_2\ w_3\ v_1\ v_2\ v_3]^T$

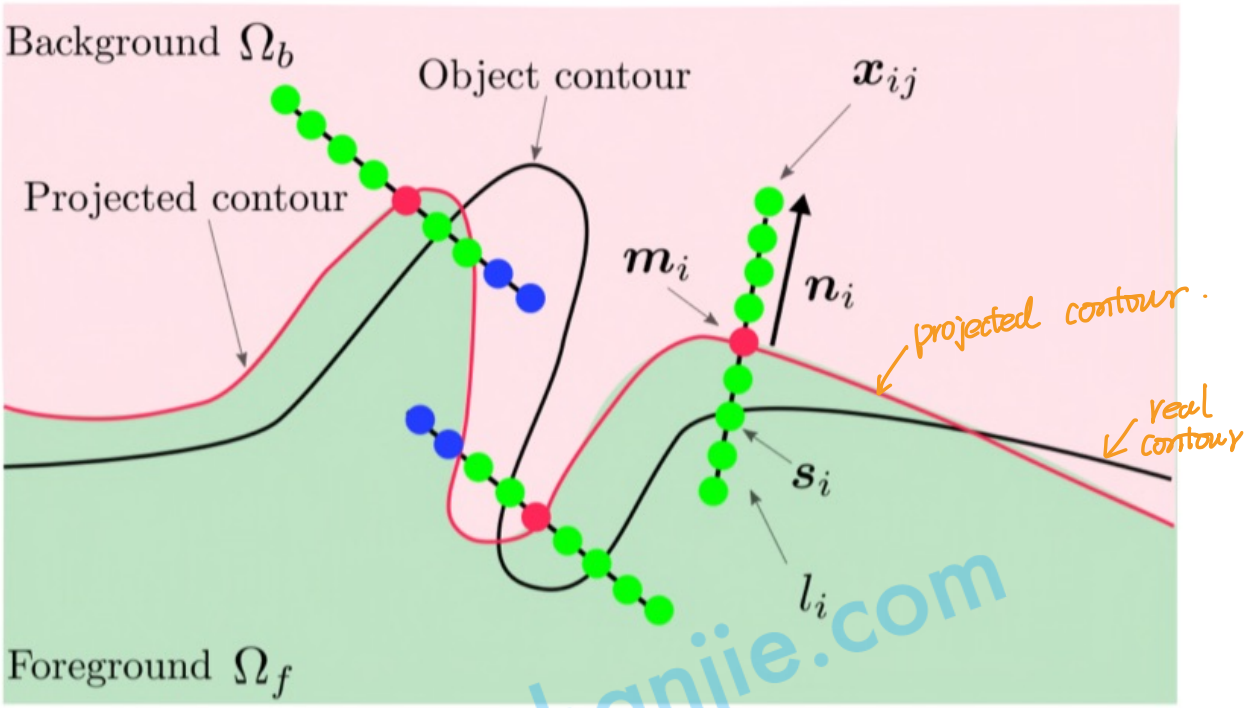


Fig. 2. The silhouette mask I_s generated by projecting the 3D model into the image plane.

$$E(p) = -\sum_{x_{ij} \in L} w(x_{ij}) \log [H_e(d(x_{ij})) P_f(x_{ij}) + (1 - H_e(d(x_{ij}))) P_b(x_{ij})]$$

① $p = [w_1\ w_2\ w_3\ v_1\ v_2\ v_3]^T$

② x_{ij} : The j-th sampling pixel of the search line l_i

③ L : all sampling pixels x_{ij}

④ $d(x_{ij}) = n_i^T (x_{ij} - m_i)$,
m_i : projected contour point.
n_i : perpendicular to the projected contour, pointing to background.

$$d(x_{ij}) \begin{cases} > 0, & x_{ij} \in \Omega_b \\ < 0, & x_{ij} \in \Omega_f \end{cases}$$

⑤ $H_e(d(x_{ij})) = \frac{1}{\pi}(-\arctan(d(x_{ij})s) + \frac{\pi}{2})$

s controls the pitch of the smoothed transition.

⑥ $P_f(x_{ij})$ and $P_b(x_{ij})$: **representing the foreground and background region membership posterior probability of each pixel's color, which are calculated by temporally consistent local color histograms**

Candidate Contour Point:

we build a bundle image I b by simply stacking each search line. As shown in Fig. 3, each row of I b is the sampling pixels x ijof the search line I i.

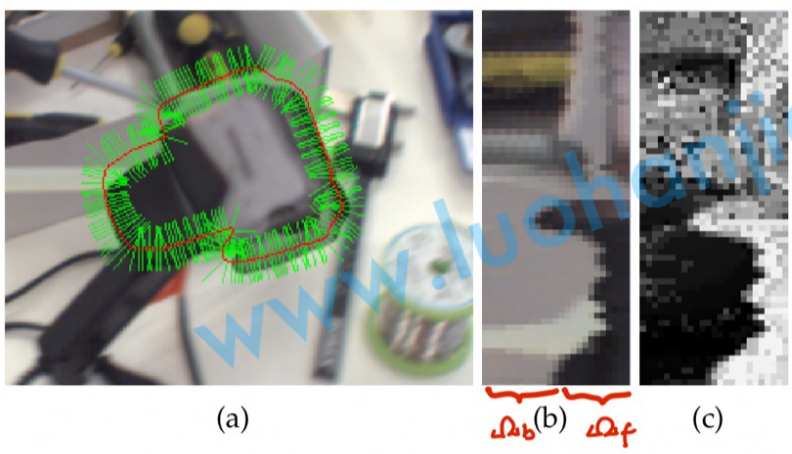


Fig. 3. Example of the search lines and the bundle image. (a) The search lines (green lines) are sampled around the projected contour (red curve). (b) Part of the bundle image I_b, which is built by stacking the search lines. (c) Foreground probability map I_p of the bundle image I_b.

we assume that the object contour point s i is in the location where the foreground probability changes rapidly. Then the candidate point h ij can be computed by 1D convolution of a 1 × 3 filter kernel f = [−1, 0, 1] at each row of I p .

$$h_{ij} = (f * I_p)(l_{ij}) > \epsilon$$

l_{ij} is the location in I_p,

The candidate contour point h ij can be extracted if the convolution value is larger than the threshold .

Probability of Contour Point

For a sampling point x_{ij}, the left area $\Phi^-_{x_{ij}}$ is defined as $(x_{i,j-1}, x_{i,j-2}, x_{i,j-3})$
right $\Phi^+_{x_{ij}}$ $(x_{i,j+1}, x_{i,j+2}, x_{i,j+3})$

$$P(h_{ij}|C) = \prod_{x_{ij} \in \Phi^-_{h_{ij}}} P_b(x_{ij}) \prod_{x_{ij} \in \Phi^+_{h_{ij}}} P_f(x_{ij})$$

the probability that the candidate point h_{ij} belongs to the contour

$$P(h_{ij}|F) = \prod_{x_{ij} \in \Phi^-_{h_{ij}}} P_f(x_{ij}) \prod_{x_{ij} \in \Phi^+_{h_{ij}}} P_f(x_{ij})$$

h_{ij} belongs to the foreground clutter point

$$P(h_{ij}|B) = \prod_{x_{ij} \in \Phi^-_{h_{ij}}} P_b(x_{ij}) \prod_{x_{ij} \in \Phi^+_{h_{ij}}} P_b(x_{ij})$$

h_{ij} belongs to the background clutter point

The normalized probability that the candidate point h_{ij} belongs to the object contour point:

$$P_c(h_{ij}) = \frac{P(h_{ij}|C)}{P(h_{ij}|C) + P(h_{ij}|F) + P(h_{ij}|B)}$$

Weight Function using Contour Constraints

$$W_c(x_{ij}) = \begin{cases} \exp(k_1 (1 - P(s_i|C))), & \text{if } s_i \neq \phi \\ \exp(k_1), & \text{otherwise} \end{cases}$$

If P(s i | C) is small, the search line l i is likely to be influenced by partial occlusions or ambiguous colors, then we reduce the weights of all sampling pixels x ij on the search line l i .

where k 1 is a negative constant that controls the rate of decay, so that the weight function w c (x ij) decays exponentially with 1 – P(s i | C).

$$W_d(x_{ij}) = \begin{cases} \exp(k_2 D(x_{ij}, s_i)), & \text{if } s_i \neq \phi \\ \exp(k_2), & \text{otherwise} \end{cases}, \quad D(x_{ij}, s_i) = \|x_{ij} - s_i\| / N_i$$

N_i: length of search line l_i

Since sampling pixels far from the object contour are more likely to be influenced by partial occlusions and cluttered backgrounds, we reduce the weights of the sampling pixels far from the object contour.

$$w(x_{ij}) = w_c(x_{ij})\, w_d(x_{ij})$$

Pose Optimization

$$E(p) = -\sum_{x_{ij} \in L} w(x_{ij}) \log[He(d(x_{ij}))P_f(x_{ij}) + (1 - He(d(x_{ij})))P_b(x_{ij})]$$

$$= \frac{1}{2} \sum_{x_{ij} \in L} w(x_{ij}) \psi(x_{ij}) F^2(x_{ij}, p)$$

where $\psi(x_{ij}) = \frac{1}{F(x_{ij}, p)}$, $F(x_{ij}, p) = -\log[He(d(x_{ij}))P_f(x_{ij}) + (1 - He(d(x_{ij})))P_b(x_{ij})]$

$$J = \frac{\partial F(x_{ij}, p)}{\partial He(d(x_{ij}))} \frac{\partial He(d(x_{ij}))}{\partial p}$$

$$\frac{\partial F(x_{ij}, p)}{\partial He(d(x_{ij}))} = \frac{P_f - P_b}{He(d(x_{ij}))P_f + (1 - He(d(x_{ij})))P_b}$$

$$\frac{\partial He(d(x_{ij}))}{\partial p} = \frac{\partial He}{\partial d(x_{ij})} \frac{\partial d(x_{ij})}{\partial m_i} \frac{\partial m_i}{\partial p} = \delta_e(d(x_{ij}))\, n_i^T \frac{\partial m_i}{\partial p}$$

$$\delta_e(d(x_{ij})) = \frac{\partial He}{\partial d(x_{ij})} = \frac{s}{\pi(1 + d(x_{ij})^2 s^2)}$$

$$\therefore \Delta p = -\left(\sum_{x_{ij} \in L} w(x_{ij}) \psi(x_{ij}) J^T J\right)^{-1} \sum_{x_{ij} \in L} w(x_{ij}) J^T$$

TABLE 3
Average runtime (in ms) of each step for one frame on RBOT dataset. Please see the text for more explanations.

Method	RD	PP	SL	CS	DT	PO	HU	Total
[25]	18.5	4.6	-	-	2.1	2.2	6.2	33.6
Ours	18.6	4.4	1.5	0.7	-	1.9	6.2	33.3

RD: rendering
PP: pixel-wise posterior probability caculation
SL: search lines sampling
CS: object contour points searching
DT: signed distance transform
PO: pose optimization
HU: histogram up-date

A Sparse Gaussian Approach to Region-Based 6DOF Object Tracking (2021)

$X_i = [X_i\ Y_i\ Z_i]^T$, 3D model Points
$\tilde{X}_i = [X_i\ Y_i\ Z_i\ 1]^T$
$I : \Omega \to \{0, \dots, 255\}^3$, color image I with image domain $\Omega \subset \mathbb{R}^2$.
$y_i = I(x_i)$, color values y_i at image $x_i = [x_i\ y_i]^T$

$$x_i = \pi_L(X_i) = \begin{bmatrix} X_i f_x / Z_i + p_x \\ Y_i f_y / Z_i + p_y \end{bmatrix}$$

$$^{c}\tilde{X}_i = {}^{c}_{M}T\ {}^{M}\tilde{X}_i = \begin{bmatrix} {}^{c}_{M}R & {}^{c}_{M}t \\ 0 & 1 \end{bmatrix} {}^{M}\tilde{X}_i$$

camera ref frame ← $^{c}\tilde{X}_i$; model ref frame ← $^{M}\tilde{X}_i$

$$\because\ R = \exp([r]_\times) = I + [r]_\times + \frac{1}{2!}[r]_\times^2 + \frac{1}{3!}[r]_\times^3 + \cdots$$

$$[r]_\times = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix}_\times = \begin{bmatrix} 0 & r_3 & -r_2 \\ r_3 & 0 & -r_1 \\ -r_2 & r_1 & 0 \end{bmatrix}$$

The linear variation of a 3D model point represented in camera reference frame C:

$$^{c}\tilde{X}_i^{+} = \begin{bmatrix} {}^{c}_{M}R & {}^{c}_{M}t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I + [\theta_r]_\times & \theta_t \\ 0 & 1 \end{bmatrix} {}^{M}\tilde{X}_i$$

$$\theta^T = [\theta_r^T\ \theta_t^T]$$

It is more natural to variate a model point (右乘) in the model frame since object is typically moved significantly more than the camera.

A correspondence line is described by a center $c_i = [c_{xi}\ c_{yi}]^T \in \mathbb{R}^2$ and a normal vector $n_i = [n_{xi}\ n_{yi}]^T \in \mathbb{R}^2$, with $\|n_i\|_2 = 1$ in image.

3D colour point X_i and associated vector normal to Contour N_i —Project to image→ c_i, n_i

pixels on the line : $x_{cl_i}(r) = \lfloor c_i + r n_i + 0.5 \rfloor$

↑ distance from center c_i

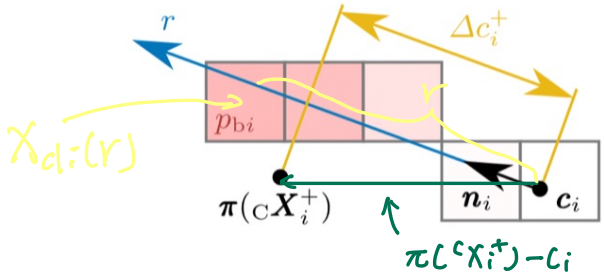


Fig. 2. Correspondence line defined by a center $\boldsymbol{c}_i$ and a normal vector $\boldsymbol{n}_i$, with selected pixels and the projected difference Δc_i^+ from $\boldsymbol{c}_i$ to the variated point ${}^c\boldsymbol{X}_i^+$. The color intensity in red indicates the magnitude of the pixel-wise posterior p_{bi} for each pixel.

During the pose variation in 6Dof, the projected difference Δc_i^+ from the unmoved center c_i to the variated model point ${}^cX_i^+$ is calculated as:

$$\Delta c_i^+ = n_i^T (\pi({}^cX_i^+) - c_i)$$

The color on the line with distance r:

$$y_i(r) = I(x_{cl_i}(r)) \qquad x_{cl_i}(r) = \lfloor c_i + r n_i + 0.5 \rfloor$$

The pixel-wise posterior, of the model m_j, given a pixel value $y_i(r)$:

$$p_{j,i}(r) = p(m_j \mid y_i(r)) = \frac{p(y_i(r) \mid m_j)\, p(m_j)}{p(y_i(r))} = \frac{p(y_i(r) \mid m_j)\, p(m_j)}{p(y_i(r) \mid m_f)\, p(m_f) + p(y_i(r) \mid m_b)\, p(m_b)}, \quad j \in \{f, b\}$$

m_f: foreground model
background model

$$= \frac{p(y_i(r) \mid m_j)}{p(y_i(r) \mid m_f) + p(y_i(r) \mid m_b)}$$

Using the knowledge that foreground and background are equally likely along the correspondence line, i.e. p(m f) = p(m b),

$$= \frac{p(y_i(r) \mid m_j)\, n_j}{p(y_i(r) \mid m_f)\, n_f + p(y_i(r) \mid m_b)\, n_b}$$

n_f: foreground pixels number
n_b: background pixels number

$p_j(y) = \frac{p(M_j \mid y)}{n_j} = \frac{1}{n_j} \frac{p(y \mid M_j)\frac{n_j}{n}}{\sum_{i=\{f,b\}} p(y \mid M_i)\frac{n_i}{n}} = \frac{p(y \mid M_j)}{p(y \mid M_f) n_f + p(y \mid M_b) n_b}$

Define $p_{j,i}(r) = p(m_j \mid y_i(r)) = \frac{p(y_i(r) \mid m_j)\, n_j}{p(y_i(r) \mid m_f)\, n_f + p(y_i(r) \mid m_b)\, n_b}$

model, pixel

Remap $r \in R_i \longrightarrow r_s$ in a discrete space:

$$\Delta C_i^+ = n_i^T (\pi(^C X_i^+) - c_i) \longrightarrow \Delta C_{Si}^+$$

$$r_s = (r - \Delta r_i)\frac{\tilde{n}_i}{S}, \qquad \Delta C_{Si}^+ = (\Delta C_i^+ - \Delta r_i)\frac{\tilde{n}_i}{S}$$

$S \in \mathbb{N}^+$ the scale describes the number of pixels combined into a segment.

$\tilde{n}_i = \max(|n_{xi}|, |n_{yi}|)$ projects a normal vector n_x to the closest horizontal or vertical coordinate.

$\Delta r_i \in \mathbb{R}$ distance from center c_i to defined segment location (chosen such that the center $r_s = 0$ lies on the boarder between two segments).

The likelihood function in scale-space:

$$p(D_i | \Delta\tilde{C}_{Si}) \propto \prod_{r_s \in R_s} (h_f(r_s - \Delta\tilde{C}_{Si}) P_{sfi}(r_s) + h_b(r_s - \Delta\tilde{C}_{Si}) P_{sbi}(r_s))$$

↑ $\Delta\tilde{C}_{Si}$ is a function of θ (pose)

$$p(D_i|\theta) \propto \prod_{r \in R_i} (h_f(r - \Delta C_i^+) P_{fi}(r) + h_b(r - \Delta C_i^+) P_{bi}(r))$$

R_s a set of distance to segment centers, $\Delta\tilde{C}_{Si}$ has to be chosen such that precalculated values aligned with segment centers. For $h_f(x)$, $h_b(x)$, 10 precomputed values $x \in \{-4.5, -3.5, -2.5, -1.5, -0.5, 0.5, 1.5, 2.5, 3.5, 4.5\}$ are used. (x: function)

先将 $r \longrightarrow r_s$，然后通过 offset $\Delta\tilde{C}_{Si}$ 得到 precalculated values 对齐到 segment centers.

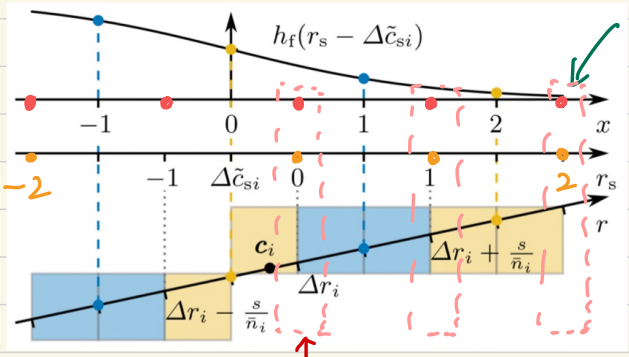
P_{sfi} and P_{sbi} segment-wise posteriors. Assuming pixel-wise independence: $r_s \xrightarrow{S(r_s)} \{r\}$

$$P_{sji}(r_s) = \frac{\prod_{r \in S(r_s)} P(y_i(r)|m_j)}{\prod_{r \in S(r_s)} P(y_i(r)|m_f) + \prod_{r \in S(r_s)} P(y_i(r)|m_b)}, \quad j \in \{f, b\}$$

(all r belongs to r_s)

$$P_{ji}(r) = \frac{P(m_j | y_i(r))}{\eta_j} = \frac{P(y_i(r)|m_j)}{P(y_i(r)|m_f)\eta_f + P(y_i(r)|m_b)\eta_b}$$

where S is a set-valued function that maps r_s to a set of values r that describe the S (scale) closest pixel centers of a segment.



precomputed values of h_f

$h_f(x)$

$h_f(r_s - \Delta\tilde{c}_{si}) P_{sfi}(r_s)$

① $P(D_i | \Delta\tilde{c}_{si})$ is calculated for 11 discrete values $\Delta\tilde{c}_{si} \in \{-5, -4, \ldots 5\}$ distribution_length

② For $h_f(x)$, $h_b(x)$, 10 precomputed values $x \in \{-4.5, -3.5, -2.5, -1.5, -0.5, 0.5, 1.5, 2.5, 3.5, 4.5\}$ are used. function.

③ $h_f(x) = \frac{1}{2} - \frac{1}{2}\tanh(\frac{x}{2s_h})$, $h_b(x) = \frac{1}{2} + \frac{1}{2}\tanh(\frac{x}{2s_h})$

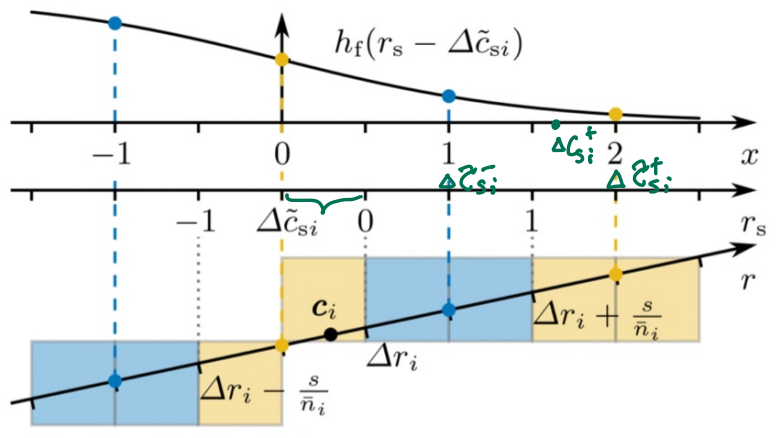
$s_h = 1.3$

1. Δr_i chosen such that the center $r_s = 0$ lies on the boarder and closest to the line center c_i

2. For $h_f(x)$, $h_b(x)$, 10 precomputed values $x \in \{-4.5, -3.5, -2.5, -1.5, -0.5, 0.5, 1.5, 2.5, 3.5, 4.5\}$ are used.

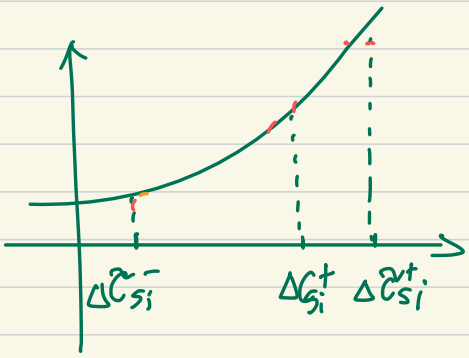
3. $\Delta\tilde{c}_{si}$ has to be chosen such that precomputed values of $h(x)$ are aligned with segment centers of r, making $h(r_s - \Delta\tilde{c}_{si})$ and $P_{si}(r_s)$ and the same position

Due to a limited number of precalculated values for h_f and h_b, $P(D_i|\Delta\tilde{c}_{si})$ can only be evaluated at discrete values $\Delta\tilde{c}_{si}$, the upper and lower neighboring discretized values $\Delta\tilde{c}_{si}^{+}$ and $\Delta\tilde{c}_{si}^{-}$ are used to linearly interpolate:



$$P(D_i|\theta) \approx (\Delta\tilde{c}_{si}^{+} - \Delta c_{si}^{+})\underbrace{P(D_i|\Delta\tilde{c}_{si}^{-})} + (\Delta c_{si}^{+} - \Delta\tilde{c}_{si}^{-})P(D_i|\Delta\tilde{c}_{si}^{+})$$

$$P(D_i|\Delta\tilde{c}_{si}) \propto \prod_{r_s \in R_s} \left(h_f(r_s - \Delta\tilde{c}_{si})P_{sf_i}(r_s) + h_b(r_s - \Delta\tilde{c}_{si})P_{sf_i}(r_s)\right)$$



$$\Delta c_{si}^{+} = (\Delta c_i^{+} - \Delta r_i)\frac{\bar{n}_i}{s}$$

$$P(D_i|\Delta c_{si}^{+}) = P(D_i|\Delta\tilde{c}_{si}^{-}) + (\Delta c_{si}^{+} - \Delta\tilde{c}_{si}^{-})\frac{(P(D_i|\Delta\tilde{c}_{si}^{+}) - P(D_i|\Delta\tilde{c}_{si}^{-}))}{(\Delta\tilde{c}_{si}^{+} - \Delta\tilde{c}_{si}^{-})}$$

$$= \frac{(\Delta\tilde{c}_{si}^{+} - \Delta\tilde{c}_{si}^{-}) - (\Delta c_{si}^{+} - \Delta\tilde{c}_{si}^{-})}{(\Delta\tilde{c}_{si}^{+} - \Delta\tilde{c}_{si}^{-})}P(D_i|\Delta\tilde{c}_{si}^{-})$$

$$+ \frac{(\Delta c_{si}^{+} - \Delta\tilde{c}_{si}^{-})}{(\Delta\tilde{c}_{si}^{+} - \Delta\tilde{c}_{si}^{-})}P(D_i|\Delta\tilde{c}_{si}^{+})$$

$$= \frac{\Delta\tilde{c}_{si}^{+} - \Delta c_{si}^{+}}{\Delta\tilde{c}_{si}^{+} - \Delta\tilde{c}_{si}^{-}}P(D_i|\Delta\tilde{c}_{si}^{-}) + \frac{\Delta c_{si}^{+} - \Delta\tilde{c}_{si}^{-}}{\Delta\tilde{c}_{si}^{+} - \Delta\tilde{c}_{si}^{-}}P(D_i|\Delta\tilde{c}_{si}^{+})$$

∵ $\Delta\tilde{c}_{si}^{+} - \Delta\tilde{c}_{si}^{-}$ is a fixed value, it can be dropped.

Gaussian Equivalence

Newton optimization yields good results for Gaussian distributions.

The goal is to find smoothed step function h_f and h_b that ensure that the likelihood follows a Gaussian distribution. we want the first-order derivative of the logarithm of the likehood has to be equal to that of a normal distribution.

$$\underbrace{P(\mathcal{D}_i|\Theta)}_{P(\Phi,p|\mathcal{L},\Omega)} \propto \prod_{r\in\mathcal{R}_i} \Big(\underbrace{h_f(r-\Delta c_i^+)}_{P(x_i|\Phi,p,M_f)} \underbrace{P_{fi}(r)}_{P(M_f|y_i)} + h_b(r-\Delta c_i^+) P_{bi}(r) \Big)$$

① $\Theta^T = [\Theta_r^T \ \Theta_t^T]$
② $\mathcal{D}_i$ the data specific to a single correspondence line,
③ $\mathcal{R}_i$ a set distance r from the line center to pixel centers.
④ P_{fi} and P_{bi} the pixel-wise posteriors for foreground and background.
⑤ Δc_i^+ the projected difference from correspondence line center c_i to the varyated model point ${}^c x_i^+$

$${}^c\tilde{x}_i^+ = \begin{bmatrix} {}^c_M R & {}^c_M t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I + [\theta_r]_\times & \theta_t \\ 0 & 1 \end{bmatrix} {}^M\tilde{x}_i, \qquad c_i = \pi\, {}^c x_i$$
$$\Delta c_i^+ = n_i^T (\pi({}^c x_i^+) - c_i)$$

For P_{fi} and P_{bi}, perfect segmentation (分割. 正确pose) and a contour at the correspondence line center assumed.

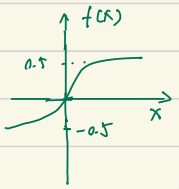
$$P_{fi}(r) = \begin{cases} 1 & \text{if } r \le 0 \\ 0 & \text{else} \end{cases}, \qquad P_{bi}(r) = \begin{cases} 0 & \text{, if } r \le 0 \\ 1 & \text{, else} \end{cases}$$

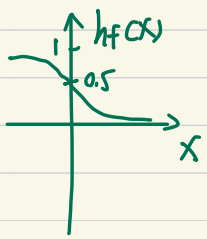
Suppose:
① $h_f + h_b = 1$
② h_f and h_b be symmetric.

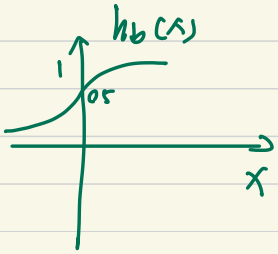
$$\therefore \quad h_f(x) = 0.5 - f(x), \qquad h_b(x) = 0.5 + f(x)$$

where $f(x)$ is an odd function ($-f(x) = f(-x)$) that lies within the interval $[-0.5, 0.5]$ and

$$\lim_{x\to\infty} f(x) = 0.5 \quad \text{and} \quad \lim_{x\to-\infty} f(x) = -0.5$$







Infinitesimally small pixels are assumed to write the likihood from in continuous form (Product Integral)

$$P(D_i|\theta) \propto \prod_{r=-\infty}^{\infty} \left(h_f(r-\Delta C_i^+)\, P_{fi}(r) + h_b(r-\Delta C_i^+)\, P_{bi}(r)\right)^{dr}$$

Convert the product integral to Riemann integral in the perfect pixel-wise posteriors assumption:

$$\prod_a^b f(x)^{dx} = \lim_{\Delta x \to 0} \prod f(x_i)^{\Delta x} = \exp\left(\int_a^b \ln f(x)\, dx\right)$$

$$P(D_i|\theta) \propto \exp\left(\int_{r=-\infty}^{\infty} \ln\left(h_f(r-\Delta C_i^+)\, P_{fi}(r) + h_b(r-\Delta C_i^+)\, P_{bi}(r)\right) dr\right)$$

$$= \exp\left(\int_{r=-\infty}^{\infty} \ln\left(h_f(r-\Delta C_i^+)\, P_{fi}(r)\right) dr + \int_{r=-\infty}^{\infty} \ln\left(h_b(r-\Delta C_i^+)\, P_{bi}(r)\right) dr\right)$$

$$P_{fi}(r) = \begin{cases} 1 & \text{if } r \le 0 \\ 0 & \text{else} \end{cases}, \quad P_{bi}(r) = \begin{cases} 0 & \text{if } r \le 0 \\ 1 & \text{else} \end{cases}$$

$$= \exp\left(\int_{r=-\infty}^{0} \ln\left(h_f(r-\Delta C_i^+)\right) dr + \int_{r=0}^{\infty} \ln\left(h_b(r-\Delta C_i^+)\right) dr\right)$$

let $r - \Delta C_i^+ = x$ $\therefore$ $\int_{r=-\infty}^{0} \Longrightarrow \int_{x=-\infty}^{-\Delta C_i^+}$, $dr = d(x+\Delta C_i^+) = dx$

$$= \exp\left(\int_{x=-\infty}^{-\Delta C_i^+} \ln\left(h_f(x)\right) dx + \int_{x=-\Delta C_i^+}^{\infty} \ln\left(h_b(x)\right) dx\right)$$

To eliminate both constant scaling factors and the integral:

$$\frac{\partial \ln(P(D_i|\theta))}{\partial \Delta C_i^+} = \frac{\partial \ln\left[\exp\left(\int_{x=-\infty}^{-\Delta C_i^+} \ln\left(h_f(x)\right) dx + \int_{x=-\Delta C_i^+}^{\infty} \ln\left(h_b(x)\right) dx\right)\right]}{\partial \Delta C_i^+}$$

$$= \frac{\partial\left(\int_{x=-\infty}^{-\Delta C_i^+} \ln\left(h_f(x)\right) dx + \int_{x=-\Delta C_i^+}^{\infty} \ln\left(h_b(x)\right) dx\right)}{\partial \Delta C_i^+}$$

① $\dfrac{\partial\left(\int_{x=-\infty}^{-\Delta C_i^+} \ln\left(h_f(x)\right) dx\right)}{\partial \Delta C_i^+}$

Leibniz Rule: If $\Phi(t) = \int_{a(t)}^{b(t)} f(x,t)\, dx$, then

$$\frac{d\Phi}{dt} = \int_{a(t)}^{b(t)} \frac{\partial f(x,t)}{\partial t}\, dx + f(b(t),t)\frac{db}{dt} - f(a(t),t)\frac{da}{dt}$$

$$= \int_{x=-\infty}^{-\Delta C_i^+} \frac{\partial \ln\left(h_f(x)\right)}{\partial \Delta C_i^+}\, dx - \ln\left(h_f(-\Delta C_i^+)\right)$$

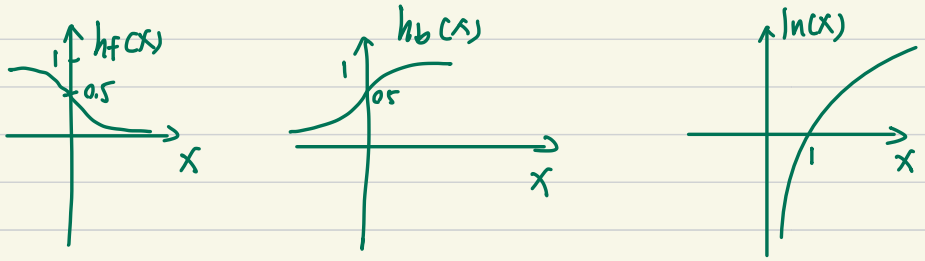
$$= -\ln\left(h_f(-\Delta C_i^+)\right)$$

② $$\frac{\partial\left(\int_{x=-\Delta C_i^+}^{\infty} \ln(h_b(x))\,dx\right)}{\partial \Delta C_i^+}$$

$$= 0 + 0 + \ln(h_b(-\Delta C_i^+))$$

$$\therefore \frac{\partial \ln(P(D_i|\theta))}{\partial \Delta C_i^+} = -\ln(h_f(-\Delta C_i^+)) + \ln(h_b(-\Delta C_i^+))$$

$\because$



$$\therefore \lim_{x\to-\infty} \ln(h_f(x)) = 0 \quad , \quad \lim_{x\to\infty} \ln(h_b(x)) = 0$$

$\because$ $h_f(x) = 0.5 - f(x)$, $h_b(x) = 0.5 + f(x)$

$$\therefore \frac{\partial \ln(P(D_i|\theta))}{\partial \Delta C_i^+} = -\ln(0.5 - f(-\Delta C_i^+)) + \ln(0.5 + f(-\Delta C_i^+))$$

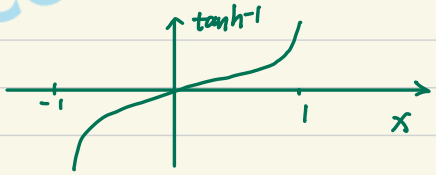
$$= -\ln(0.5(1 - 2f(-\Delta C_i^+))) + \ln(0.5(1 + 2f(-\Delta C_i^+)))$$

$$= -\ln(1 - 2f(-\Delta C_i^+)) - \ln 0.5 + \ln(1 + 2f(-\Delta C_i^+)) + \ln 0.5$$

$$= -\ln(1 - 2f(-\Delta C_i^+)) + \ln(1 + 2f(-\Delta C_i^+))$$

$\because$ The inverse hyperbolic tangent: $2\tanh^{-1}(z) = -\ln(1-z) + \ln(1+z)$

$$\frac{\partial \ln(P(D_i|\theta))}{\partial \Delta C_i^+} = 2\tanh^{-1}(2f(-\Delta C_i^+))$$



We want to enforce our likelihood function follows a normal distribution $X \sim N(\mu, \sigma^2)$

The PDF: $f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$

$\therefore$ let $\Delta C_i^+ \sim N(0, S_h) \propto \exp(-\frac{1}{2S_h}(\Delta C_i^+)^2)$

We attain a probabilistic model that follows a Gaussian distribution. Although assumptions, such as perfect pixel-wise posteriors and infinitesimally small pixels, are not an exact description of reality. Experiments demonstrate that we achieve excellent convergence for used regularized Newton optimization.

Regularized Newton Method

To maximize the likelihood, we estimate the variation vector θ and iteratively update the pose. We use newton optimization with Tikhonov regularization:

$$\hat{\theta} = (-H + \begin{bmatrix} \lambda_r I_3 & 0 \\ 0 & \lambda_t I_3 \end{bmatrix})^{-1} g$$

where $g^T = \left.\frac{\partial \ln(P(D|\theta))}{\partial \theta}\right|_{\theta=0}$, $H = \left.\frac{\partial^2 \ln(P(D|\theta))}{\partial \theta^2}\right|_{\theta=0}$

λ_r and λ_t the regularization parameters for rotation and translation.

$\arg\min_x (\|Ax-b\|^2 + \|\Gamma x\|^2)$, Γ is chosen as a $\Gamma = \alpha I$, $\hat{x} = (A^TA + \Gamma^T\Gamma)^{-1}A^Tb$
$= (A^TA + (\alpha I)^T(\alpha I))^{-1}A^Tb$
$= (A^TA + \alpha^2 I)A^Tb$

Finally, the pose can be updated according to:

$${}^C_M T = {}^C_M T \begin{bmatrix} \exp([\hat{\theta}_r]_\times) & \hat{\theta}_t \\ 0 & 1 \end{bmatrix}$$

The full likelihood: n_{cl} is the number of correspondence lines.

$$P(\mathcal{D}|\theta) \propto \prod_{i=1}^{n_{cl}} P(\mathcal{D}_i|\theta)$$

$$\therefore \ln P(\mathcal{D}|\theta) \propto \sum_{i=1}^{n_{cl}} \ln P(\mathcal{D}_i|\theta)$$

$$\Delta\tilde{C}_{si}^{-} \le \Delta C_{si}^{+} \le \Delta\tilde{C}_{si}^{+}$$

$$P(\mathcal{D}_i|\theta) \propto (\Delta\tilde{C}_{si}^{+} - \Delta C_{si}^{+}) P(\mathcal{D}_i|\Delta\tilde{C}_{si}^{-}) + (\Delta C_{si}^{+} - \Delta\tilde{C}_{si}^{-}) P(\mathcal{D}_i|\Delta\tilde{C}_{si}^{+})$$

$$P(\mathcal{D}_i|\Delta\tilde{C}_{si}) \propto \prod_{r_s \in R_s} (h_f(r_s - \Delta\tilde{C}_{si}) P_{sf_i}(r_s) + h_b(r_s - \Delta\tilde{C}_{si}) P_{sf_i}(r_s))$$

$$\Delta C_i^{+} = n_i^T (\pi(^{C}X_i^{+}) - c_i)$$

$$^{C}\tilde{X}_i^{+} = \begin{bmatrix} {}^{C}_{M}R & {}^{C}_{M}t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I + [\theta_r]_\times & \theta_t \\ 0 & 1 \end{bmatrix} {}^{M}\tilde{X}_i$$

$$\Delta C_{si}^{+} = (\Delta C_i^{+} - \Delta r_i)\frac{\bar{n}_i}{s}$$

$$g^T = \frac{\partial \ln(P(\mathcal{D}|\theta))}{\partial\theta}\bigg|_{\theta=0} = \sum_{i=1}^{n_{cl}} \frac{\partial \ln(P(\mathcal{D}_i|\theta))}{\partial \Delta C_{si}^{+}} \frac{\partial \Delta C_{si}^{+}}{\partial\, {}^{C}X_i^{+}} \frac{\partial\, {}^{C}X_i^{+}}{\partial\theta}\bigg|_{\theta=0}$$

linear interpdcate

$$\frac{\partial \ln(P(\mathcal{D}_i|\theta))}{\partial \Delta C_{si}^{+}}\bigg|_{\theta=0} \approx \frac{\partial(\ln[(\Delta\tilde{C}_{si}^{+} - \Delta C_{si}^{+}) P(\mathcal{D}_i|\Delta\tilde{C}_{si}^{-}) + (\Delta C_{si}^{+} - \Delta\tilde{C}_{si}^{-}) P(\mathcal{D}_i|\Delta\tilde{C}_{si}^{+})])}{\partial \Delta C_{si}^{+}}\bigg|_{\theta=0}$$

$$= \frac{-P(\mathcal{D}_i|\Delta\tilde{C}_{si}^{-}) + P(\mathcal{D}_i|\Delta\tilde{C}_{si}^{+})}{(\Delta\tilde{C}_{si}^{+} - \Delta C_{si}^{+}) P(\mathcal{D}_i|\Delta\tilde{C}_{si}^{-}) + (\Delta C_{si}^{+} - \Delta\tilde{C}_{si}^{-}) P(\mathcal{D}_i|\Delta\tilde{C}_{si}^{+})}$$

$$\because P(\mathcal{D}_i|\Delta\tilde{C}_{si}) \propto \mathcal{N}(\Delta\tilde{C}_{si}|\mu_{\Delta\tilde{C}_{si}}, \sigma_{\Delta\tilde{C}_{si}})$$

$$\because P(\mathcal{D}_i|\theta) \propto (\Delta\tilde{C}_{si}^{+} - \Delta C_{si}^{+}) P(\mathcal{D}_i|\Delta\tilde{C}_{si}^{-}) + (\Delta C_{si}^{+} - \Delta\tilde{C}_{si}^{-}) P(\mathcal{D}_i|\Delta\tilde{C}_{si}^{+})$$

The linear interpdcation of two Gaussian distribution $P(\mathcal{D}_i|\Delta\tilde{C}_{si}^{-})$ and $P(\mathcal{D}_i|\Delta\tilde{C}_{si}^{+})$ is also a Gaussian distribution, and

$$P(\mathcal{D}_i|\Delta\tilde{C}_{si}^{-}) \sim \mathcal{N}(\Delta C_{si}^{+}|\mu_{\Delta\tilde{C}_{si}}, \sigma_{\Delta\tilde{C}_{si}}) \sim P(\mathcal{D}_i|\Delta\tilde{C}_{si}^{+})$$

$$\therefore P(\mathcal{D}_i|\theta) \sim \mathcal{N}(\Delta C_{si}^{+}|\mu_{\Delta\tilde{C}_{si}}, \sigma_{\Delta\tilde{C}_{si}})$$

$$\frac{\partial \ln(P(\mathcal{D}_i|\Delta\tilde{c}_{si}))}{\partial \Delta c_{si}^{+}} = \frac{\partial \ln(\mathcal{N}(\Delta c_{si}^{+} | \mu_{\Delta\tilde{c}_{si}}, \sigma_{\Delta\tilde{c}_{si}}))}{\partial \Delta c_{si}^{+}}$$

$$\frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

$$= \frac{\partial \left(\ln \frac{1}{\sigma_{\Delta\tilde{c}_{si}}\sqrt{2\pi}} + \frac{-(\Delta c_{si}^{+} - \mu_{\Delta\tilde{c}_{si}})^2}{2\sigma_{\Delta\tilde{c}_{si}}^2} \right)}{\partial \Delta c_{si}^{+}}$$

$$= \frac{-2(\Delta c_{si}^{+} - \mu_{\Delta\tilde{c}_{si}})}{2\sigma_{\Delta\tilde{c}_{si}}^2} = -\frac{(\Delta c_{si}^{+} - \mu_{\Delta\tilde{c}_{si}})}{\sigma_{\Delta\tilde{c}_{si}}^2}$$

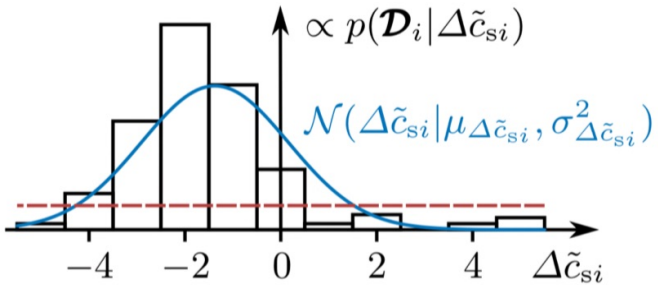


Fig. 5. Example showing normalized values of a noisy discrete likelihood $p(\mathcal{D}_i|\Delta\tilde{c}_{si})$ and the normal distribution $\mathcal{N}(\Delta\tilde{c}_{si}|\mu_{\Delta\tilde{c}_{si}}, \sigma^2_{\Delta\tilde{c}_{si}})$ that approximates that likelihood. The red line indicates a threshold for the probability values. To avoid errors from image noise and invalid pixel-wise posteriors, for values below this threshold, the normal distribution is used in the calculation of partial derivatives of the log-likelihood.

$$p(\mathcal{D}_i|\Delta\tilde{c}_{si}) \propto \prod_{r_s \in \mathcal{R}_s} (h_f(r_s - \Delta\tilde{c}_{si}) p_{sf_i}(r_s) + h_b(r_s - \Delta\tilde{c}_{si}) p_{sf_i}(r_s))$$

$$\propto \mathcal{N}(\Delta\tilde{c}_{si} | \mu_{\Delta\tilde{c}_{si}}, \sigma_{\Delta\tilde{c}_{si}})$$

and $$\frac{\partial^2 \ln(P(\mathcal{D}_i|\Delta\tilde{c}_{si}))}{\partial \Delta c_{si}^{+2}} = -\frac{1}{\sigma_{\Delta\tilde{c}_{si}}^2}$$

$${}^{c}\tilde{X}_i^{+} = \begin{bmatrix} {}^{c}_{M}R & {}^{c}t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I + [\theta_r]_\times & \theta_t \\ 0 & 1 \end{bmatrix} {}^{M}\tilde{X}_i$$

$$\left.\frac{\partial {}^{c}X_i^{+}}{\partial \theta}\right|_{\theta=0} = {}^{c}_{M}R \begin{bmatrix} -[{}^{M}X_i]_\times & I \end{bmatrix}$$

$$\Delta C_i^+ = n_i^T \left(\pi\left({}^C X_i^+\right) - C_i\right) \qquad \Delta C_{si}^+ = \left(\Delta C_i^+ - \Delta r_i\right)\frac{\bar{n}_i}{s}$$

$$x_i = \pi(X_i) = \begin{bmatrix} X_i f_x / Z_i + p_x \\ Y_i f_y / Z_i + p_y \end{bmatrix}$$

$$\Delta C_{si}^+ = \left(\Delta C_i^+ - \Delta r_i\right)\frac{\bar{n}_i}{s} = \left(n_i^T\left(\pi\left({}^C X_i^+\right) - C_i\right) - \Delta r_i\right)\frac{\bar{n}_i}{s}$$

$$= \underbrace{\frac{\bar{n}_i}{s}}_{\text{scalar}}\left(\begin{bmatrix} n_{ix} & n_{iy} \end{bmatrix}\begin{bmatrix} {}^C X_i f_x / {}^C Z_i + p_x - c_{ix} \\ {}^C Y_i f_y / {}^C Z_i + p_y - c_{iy} \end{bmatrix} - \underbrace{\Delta r_i}_{\text{scalar}}\right)$$

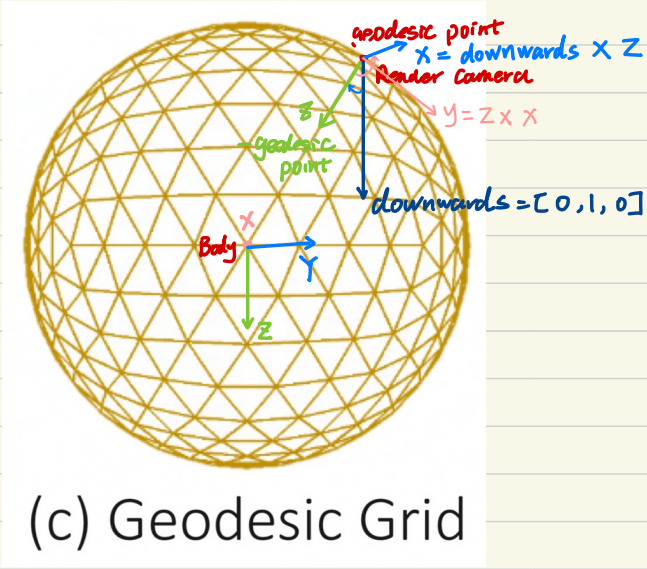
$$= \frac{\bar{n}_i}{s}\left(n_{ix}\left({}^C X_i f_x / {}^C Z_i + p_x - c_{ix}\right) + n_{iy}\left({}^C Y_i f_y / {}^C Z_i + p_y - c_{iy}\right) - \Delta r_i\right)$$

$$\frac{\partial \Delta C_{si}^+}{\partial {}^C X_i^+} = \frac{\bar{n}_i}{s}\left[\frac{n_{ix} f_x}{{}^C Z_i} \quad \frac{n_{iy} f_y}{{}^C Z_i} \quad -\frac{n_{ix}\,{}^C X_i f_x + n_{iy}\,{}^C Y_i f_y}{{}^C Z_i^2}\right]$$

$$H \propto \sum_{i=1}^{n_{c1}} \frac{\partial^2 \ln\left(P\left(D_i \mid \theta\right)\right)}{\partial \Delta C_{si}^{+2}}\left(\frac{\partial \Delta C_{si}^+}{\partial {}^C X_i^+}\frac{\partial {}^C X_i^+}{\partial \theta}\right)^T\left(\frac{\partial \Delta C_{si}^+}{\partial {}^C X_i^+}\frac{\partial {}^C X_i^+}{\partial \theta}\right)\Bigg|_{\theta=0}$$

Geodesic Grid

The 3D model of an object is thereby render from 2562 different viewpoints that placed on the vertices of a geodesic grid with a distance of 0.8m to the object center. For each rendering, we randomly sample $n_{cl}=2000$ points from the object contour.



(c) Geodesic Grid

Code

Init

1. ReadPosesRBOTDataset
2. Init Tracker( )

NormalImageViewer { RenderGeometry : holds rendering data for all assigned bodies as well as the glfw context for the render — make render camera has the same intrinsics as real camera
DatasetRBOTCamera : load images from the RBOT dataset.

Renders all bodies into a normal image that is displayed on top of a color camera image

Body : holds all body info such as name, geometry data, pose and occlusion ids

Model : stores a model of a body that consists of template views with multiple contour points. It includes all functionality to generate, save and load the model.

RegionModality : { Body, Model, DatasetRBOTCamera }

correspondence search, calculation of the gradient vector and hessian matrix. pose optimation, color histograms, visulization. occlusions using an occlusion mask

Tracker : { NormalImageViewer { RenderGeometry, DatasetRBOTCamera }, RegionModality : { Body, Model, DatasetRBOTCamera } }

Evaluate

1. Init Bodies

① Model → GenerateModel

a. Generate camera poses (GenerateGeodesicPoses) see Geodesic Grid

b. NormalImageRender : Render both a depth image and an image where the normal vector of the surface is encoded in the color of each pixel

setUpRender → RenderGeometry. AddBody → LoadMeshIntoVertices( )

C. Generate template views

normal_image : CV_8UC4
depth_image : CV_16U

For each view :

① Render image by GL

② FetchNormalImage (BGRA, 8UC4)

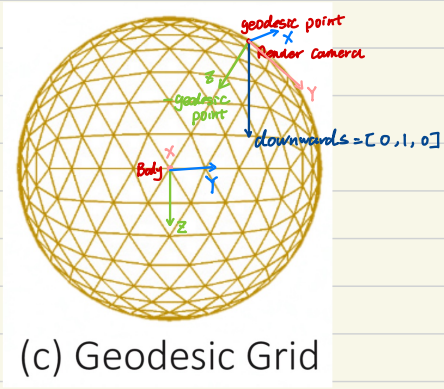
③ FetchDepthImage (16U)

④ GeneratePointData

1. get A channel of BGRA image.
2. findContours
3. Calculate data for contour points
   a. Randomly sample point on contour
   b. compute the point 3D position in the model body frame

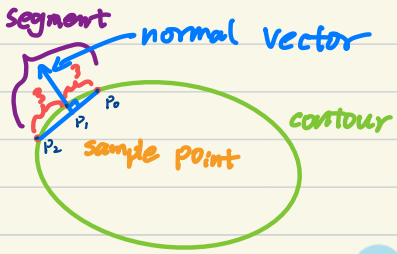
2D pixel → 3D in Cam frame → 3D in Body frame. The 3D point of model of the contour.

A channel of BGRA image.



(c) Geodesic Grid

C. Calculate contour segment and approximate normal vector



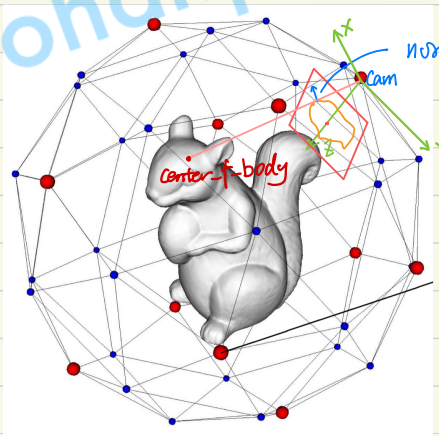
$\vec{n} \cdot (P_0 - P_1) = 0$

$= \vec{n} \cdot \vec{v} = 0$

$= n_x v_x + n_y v_y = 0$

$\therefore \vec{n} = [-v_y, v_x]$

D. normal vector of the sample contour in Body frame

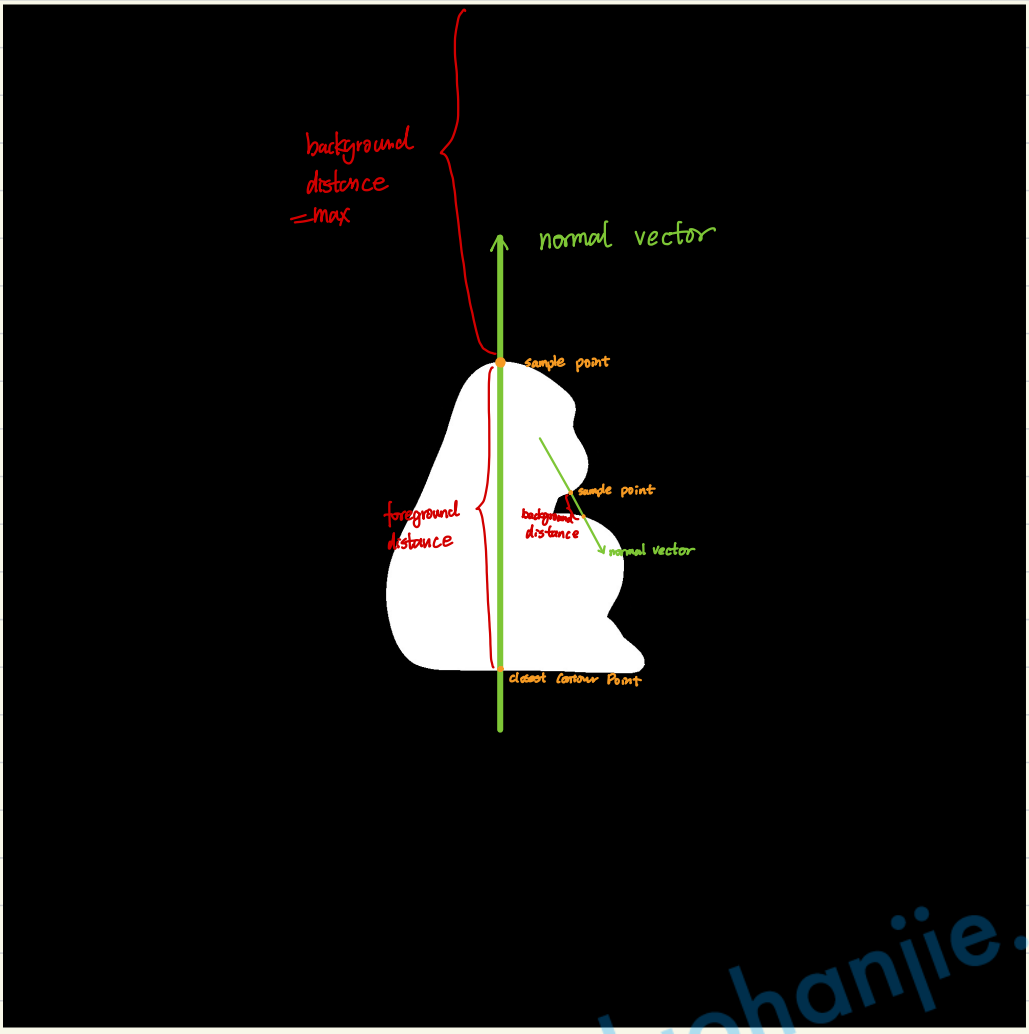


E. compute continuous distances for the foreground and background where the corresponding regions are not interrupted by the other. starting from a coordinate, those distance are measured along the normal vector.

$$\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} f_x x + c_x z \\ f_y y + c_y z \\ z \end{bmatrix} = \begin{bmatrix} f_x \frac{x}{z} + c_x \\ f_y \frac{y}{z} + c_y \\ 1 \end{bmatrix} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

$$f_x \frac{x}{z} + c_x = u \Rightarrow \frac{z}{f_x} = \frac{x}{c_x - u}, \quad x \frac{f_x}{z} = c_x - u$$

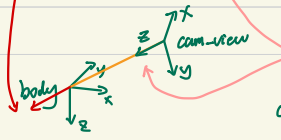
$$\therefore \text{pixel\_to\_meter} = \frac{z}{f_x}$$



Finally, we get these from GenerateModel:

the contour point in the body of body frame

- view_orientation
  - z axis of cam_view in body
- data_point → center_f_body
- data_point → normal_f_body
- data_point → foreground_distance
- data_point → background_distance



$$T_{\text{cam\_view}}^{\text{body}} = \begin{bmatrix} 1 & 1 & 0 & 1 \end{bmatrix}$$

```
version_id
1
sphere_radius
0.8
n_divides
4
n_points
200
image_size
2000
n_template_views
2562
geometry_path
/media/luohanjie/Hanjie/RBOT_dataset/ape/ape.obj

geometry_unit_in_meter
0.001
geometry_defined_counterclockwise
1
geometry_enable_culling
0
geometry2body_pose
1 0 0 0
0 1 0 0
0 0 1 0
0 0 0 1
maximum_body_diameter
0.3
```

Camera2body_pose : ${}^{body}_{view\_cam}T$ from GenerateRenderData

body2world_pose : ${}^{world}_{body}T = {}^{cam}_{body}T$ from model to real Camera.

RegionModality->body2camera_pose : camera->world2camera · body->body2world = $I\ {}^{cam}_{body}T$

geometry2body_pose :

NormalImageRenders -> camera2world_pose : ${}^{body}_{view\_cam}T$, = Camera2body_pose

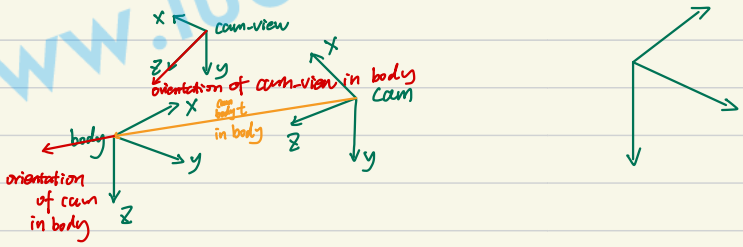
2. ResetBody (0)

① region_modality_ptr_->StartModality()

a. Init histograms

AddLinePixelColorsToTempHistograms()

① GetClosestTemplateView (${}^{cam}_{body}T$)



$({}^{cam}_{body}R)^{-1}\ {}^{cam}_{body}t = {}^{body}_{cam}t$, orientation = ${}^{body}_{cam}t\ /\ \|{}^{body}_{cam}t\|$

get template_view which ori of cam^T ori of cam_view is maximum.

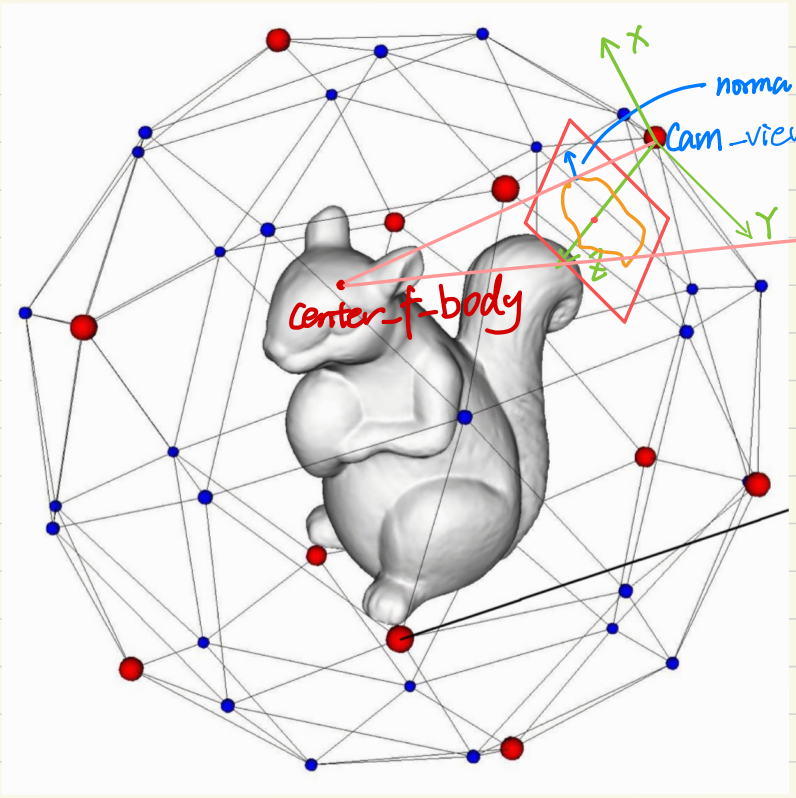
n_histogram_bins = 32

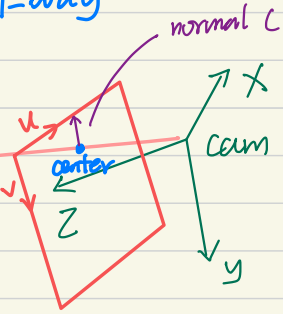
histogram_f (32^3)

histogram_b (32^3)

For each data points of the matched template_view

$$\underset{\text{cam}}{T}_{\text{body}} \cdot \underset{\text{3D point in the body which is projected become contour.}}{\text{center\_f\_body}} = \underset{\text{project point in camera frame}}{\text{center\_f\_camera}}$$





compute: data in cam frame.

center : in image

normal : in cam frame

foreground_distance : in image

background_distance : in image

compute the foreground and background histogram along the line, which maximum is 10 pixels.

Histogram (32^3) = [B $\gg 3 \times 32^2$ + G $\gg 3 \times 32$ + R $\gg 3$]

B: $\frac{R}{8}$

white: [255, 255, 255]

$= \frac{255}{8} \cdot 32 \times 32 + \frac{255}{8} 32 + \frac{255}{8}$

$= 255 \cdot 4 \times 32 + 255 \cdot 4 +$

[B (32) G (32) R (32)]

n_histogram_bins_: 32

temp_histogram_f_: n_histogram_bins_^{3}

temp_histogram_b_: n_histogram_bins_^{3}

3. Iterate over all frames

3.1 ExecuteMeasuredTrackingCycle

To find the final pose, the process is repeated 7 times, starting each time with the retrieval of a new template view. The first and second iterations use a scale of s = 5 and s = 2, respectively. For all other iterations, the scale is set to s = 1. Examples of different scales are shown in Fig. 1. The specified scales have the effect that in the first iteration a large area with low resolution is considered while short lines with high resolution are used in later iterations.

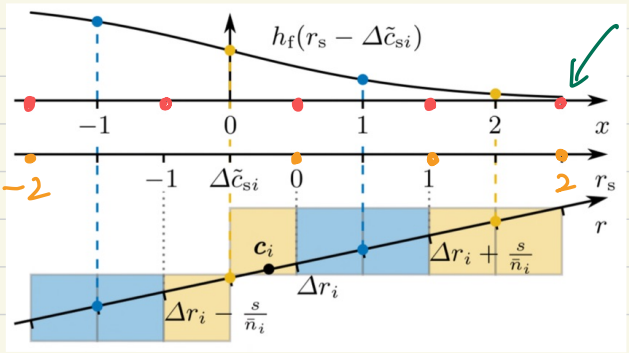
For each iteration: Scale = [5, 2, 1, 1, ...]

3.1.1 Calculate Correspondence

body2camera_pose = camera→world2camera · body→body2world = I $^{cam}_{body}T$

line_length_in_segments = function_length + distribution_length − 1

20 segments = 10 + 11



precomputed values of hf

$h_f(x)$

$\Delta\tilde{c}_{si}$

① $P(D_i | \Delta\tilde{c}_{si})$ is calculated for 11 discrete values $\Delta\tilde{c}_{si} \in \{-5, -4, \dots 5\}$ (distribution_length)

② For $h_f(x)$, $h_b(x)$, 10 precomputed values $x \in \{-4.5, -3.5, -2.5, -1.5, -0.5, 0.5, 1.5, 2.5, 3.5, 4.5\}$ are used. (function)

③ $h_f(x) = \frac{1}{2} - \frac{1}{2}\tanh\left(\frac{x}{2s_h}\right)$, $h_b(x) = \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{x}{2s_h}\right)$

$s_h = 1.3$

PrecalculateFunctionLookup: ① functionLength = 10, $x \in \{-4.5, -3.5, -2.5, -1.5, -0.5, 0.5, 1.5, 2.5, 3.5, 4.5\}$

$h_f(x) = \frac{1}{2} - \frac{1}{2}\tanh\left(\frac{x}{2 \cdot 1.3}\right)$

$h_b(x) = 1 - h_f(x)$

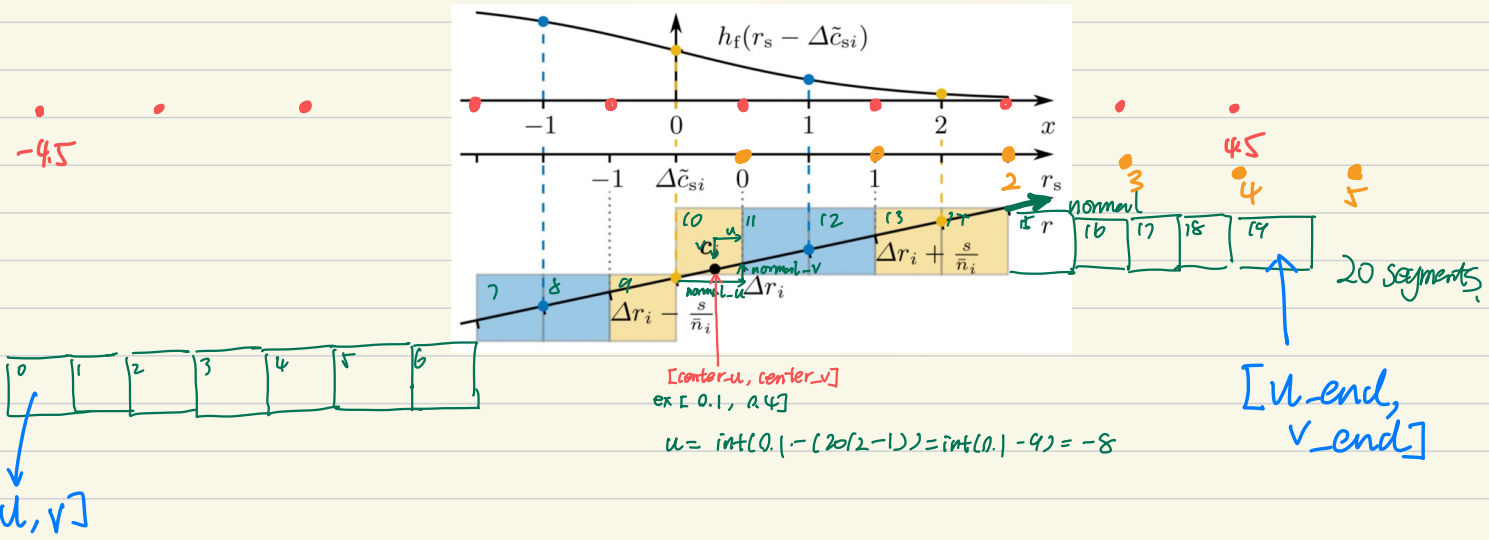
line_length = line_length_in_segments × scale

Search closest template view

For each contour point of matched template_view :

compute : center_f_body
center_f_camera
center_u, center_v
normal_u, normal_v
contrinuous : in pixels × scale

① CalculateSegmentProbabilities :



if normal_u > normal_v → v_step = $\frac{\text{normal\_v}}{\text{normal\_u}}$ 根据 scale=1

starting point u = int(center_u - ($\frac{\text{line\_length}}{2}$ - 0.5) + 0.5)

= int(center_u - (line_length/2 - 1))

u_end = u + line_length - 1

$$\Delta\tilde{c}_{si}^{-} \leq \Delta c_{si}^{+} \leq \Delta\tilde{c}_{si}^{+}$$

$$p(D_i|\theta) \propto (\Delta\tilde{c}_{si}^{+} - \Delta c_{si}^{+})\, p(D_i|\Delta\tilde{c}_{si}^{-}) + (\Delta c_{si}^{+} - \Delta\tilde{c}_{si}^{-})\, p(D_i|\Delta\tilde{c}_{si}^{+})$$

$$p(D_i|\Delta\tilde{c}_{si}) \propto \prod_{r_s \in R_s} (h_f(r_s - \Delta\tilde{c}_{si})\, p_{sfi}(r_s) + h_b(r_s - \Delta\tilde{c}_{si})\, p_{sbi}(r_s))$$

$p(D_i|\Delta\tilde{c}_{si})$ is calculated for 11 discrete values $\Delta\tilde{c}_{si} \in [-5, -4, \ldots 5]$ (distribution_length)

R_s a set of distance to segment centers

Iterate over all pixels of line and calculate probabilities $p_{sfi}(r_s)$ and $p_{sbi}(r_s)$

for $x \in [u, u\_end]$, $y \in [v, v\_end]$,

get ① pixel_color_probability_f = histogram_f_[x,y] : $p(y_i(r)|m_f)$

pixel_color_probability_b = histogram_b_[x,y] : $p(y_i(r)|m_b)$

$$\tilde{p}(y_i(r)|m_f) = \frac{p(y_i(r)|m_f)}{p(y_i(r)|m_f) + p(y_i(r)|m_b)}$$

$$\tilde{p}(y_i(r)|m_b) = \frac{p(y_i(r)|m_b)}{p(y_i(r)|m_f) + p(y_i(r)|m_b)}$$

Then:

$$segment\_probability\_f[segment\_idx] = \prod_{r \in S(r_s)} \tilde{p}(y_i(r)|m_f)$$

$$segment\_probability\_b[segment\_idx] = \prod_{r \in S(r_s)} \tilde{p}(y_i(r)|m_b)$$

where S is a set-valued function that maps r_s to a set of values r that describe the S (scale) closest pixel centers of a segment.

$$normal\_component\_to\_scale = \bar{n}_i / S$$
$$delta\_r = \Delta r_i$$

$\bar{n}_i = \max(|n_{xi}|, |n_{yi}|)$ projects a normal vector n_x to the closest horizontal or vertical coordinate.

$\Delta r_i \in \mathbb{R}$ distance from center c_i to defined segment location (chosen such that the center $r_s = 0$ lies on the boarder between two segments).

② Calculate Distribution

distribution_length = $\| \Delta \tilde{c}_{si} \in [-5, -4, \ldots 5] \|$, $\|$

Loop over entire distribution and start values of segment probabilities:

Loop over values of segment probabilities and corresponding lookup values:

Lookup {
functionLength = 10, $x \in \{-4.5, -3.5, -2.5, -1.5, -0.5, 0.5, 1.5, 2.5, 3.5, 4.5\}$

$h_f(x) = \frac{1}{2} - \frac{1}{2}\tanh\left(\frac{x}{2 \cdot 1.3}\right)$

$h_b(x) = 1 - h_f(x)$

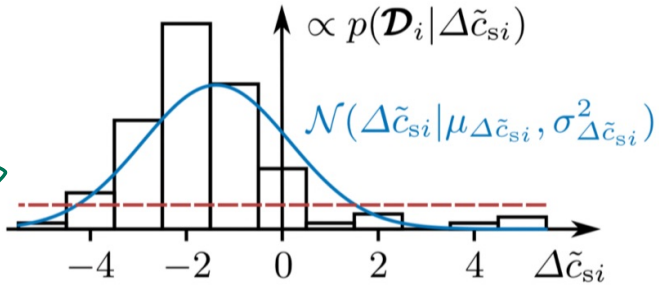
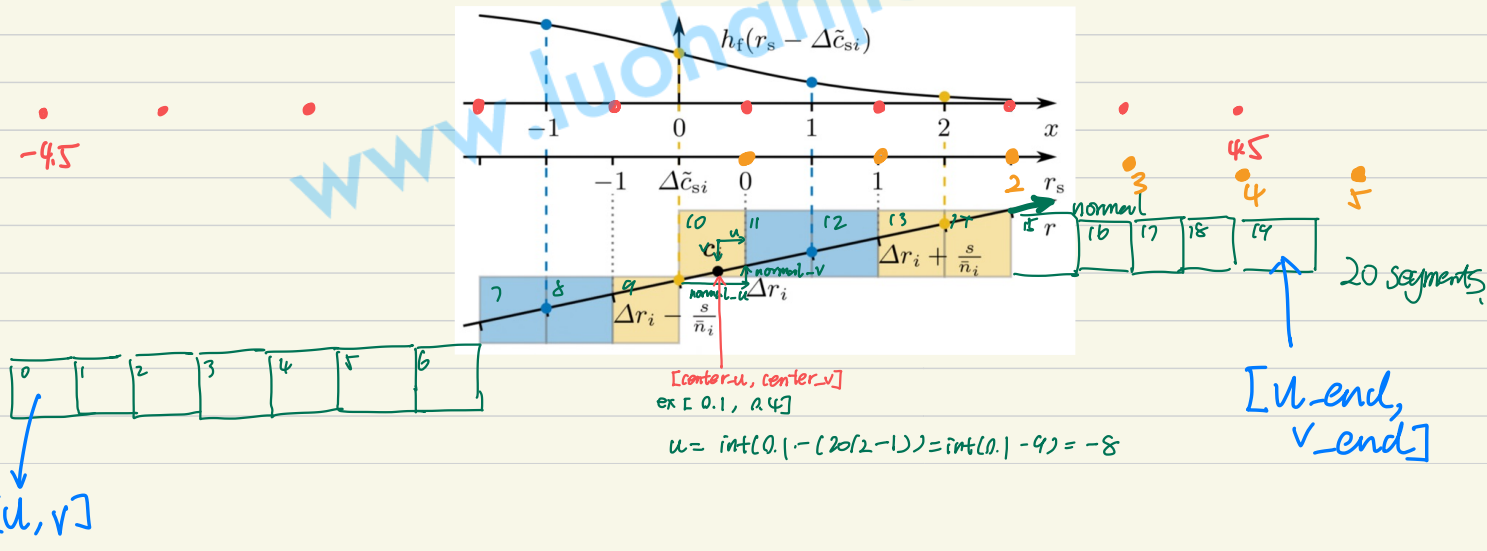


Fig. 5. Example showing normalized values of a noisy discrete likelihood $p(\boldsymbol{\mathcal{D}}_i|\Delta\tilde{c}_{si})$ and the normal distribution $\mathcal{N}(\Delta\tilde{c}_{si}|\mu_{\Delta\tilde{c}_{si}}, \sigma^2_{\Delta\tilde{c}_{si}})$ that approximates that likelihood. The red line indicates a threshold for the probability values. To avoid errors from image noise and invalid pixel-wise posteriors, for values below this threshold, the normal distribution is used in the calculation of partial derivatives of the log-likelihood.

$$p(D_i|\Delta\tilde{c}_{si}) \propto \prod_{r_s \in R_s} \left(h_f(r_s - \Delta\tilde{c}_{si}) P_{sf_i}(r_s) + h_b(r_s - \Delta\tilde{c}_{si}) P_{sf_i}(r_s)\right)$$

compute $p(D_i|\Delta\tilde{c}_{si})$ for each $\Delta\tilde{c}_{si}$



③ Calculate mean and var of $p(CD_i|\Delta\tilde{C}_{si})$

3.1.2 For each update iteration : 2

① CalculatePoseUpdate()

body2camera_pose_ : ${}^{cam}_{body}T$

$$\Delta C_i^+ = n_i^T(\pi({}^cX_i^+) - c_i) \qquad \Delta C_{si}^+ = (\Delta C_i^+ - \Delta r_i)\frac{\bar{n}_i}{s}$$

$$\frac{\partial \ln(p(CD_i|\Delta\tilde{C}_{si}))}{\partial \Delta C_{si}^+} = \begin{cases} -\dfrac{(\Delta C_{si}^+ - \mu_{\Delta\tilde{C}_{si}})}{\sigma^2_{\Delta\tilde{C}_{si}}}, & p(CD_i|\Delta\tilde{C}_{si}^-) \text{ and } p(CD_i|\Delta\tilde{C}_{si}^+) < 0.01 \\ \dfrac{-p(CD_i|\Delta\tilde{C}_{si}^-) + p(CD_i|\Delta\tilde{C}_{si}^+)}{(\Delta\tilde{C}_{si}^+ - \Delta C_{si}^+)p(CD_i|\Delta\tilde{C}_{si}^-) + (\Delta C_{si}^+ - \Delta\tilde{C}_{si}^-)p(CD_i|\Delta\tilde{C}_{si}^+)} \end{cases}$$

$$\frac{\partial \Delta C_{si}^+}{\partial {}^cX_i^+} = \frac{\bar{n}_i}{s}\left[\frac{n_{ix} f_x}{{}^cZ_i} \quad \frac{n_{iy} f_y}{{}^cZ_i} \quad -\frac{n_{ix}{}^cX_i f_x + n_{iy}{}^cY_i f_y}{{}^cZ_i^2}\right]$$

$$\left.\frac{\partial {}^cX_i^+}{\partial \theta}\right|_{\theta=0} = {}^c_M R\,[-[{}^MX_i]_\times \quad I]$$

$$\frac{\partial^2 \ln(p(CD_i|\Delta\tilde{C}_{si}))}{\partial \Delta C_{si}^{+2}} = -\frac{1}{\sigma^2_{\Delta\tilde{C}_{si}}}$$

$$g^T = \left.\frac{\partial \ln(p(CD|\theta))}{\partial\theta}\right|_{\theta=0} = \sum_{i=1}^{n_{cl}} \left.\frac{\partial \ln(p(CD_i|\theta))}{\partial \Delta C_{si}^+}\frac{\partial \Delta C_{si}^+}{\partial {}^cX_i^+}\frac{\partial {}^cX_i^+}{\partial\theta}\right|_{\theta=0}$$

$$H = \left.\frac{\partial^2 \ln(p(CD|\theta))}{\partial\theta^2}\right|_{\theta=0}$$

$$\approx \sum_{i=1}^{n_{cl}} \left.\frac{\partial^2 \ln(p(CD_i|\theta))}{\partial \Delta C_{si}^{+2}}\left(\frac{\partial \Delta C_{si}^+}{\partial {}^cX_i^+}\frac{\partial {}^cX_i^+}{\partial\theta}\right)^T\left(\frac{\partial \Delta C_{si}^+}{\partial {}^cX_i^+}\frac{\partial {}^cX_i^+}{\partial\theta}\right)\right|_{\theta=0}$$

$$\hat{\theta} = (-H + \begin{bmatrix}\lambda_r I_3 & 0 \\ 0 & \lambda_t I_3\end{bmatrix})^{-1} g$$

delta-cs: ΔC_{si}^{+}

dloglikelihood_ddelta_cs: $\frac{\partial \ln(PCD_i|\Delta\tilde{C}_{si})}{\partial \Delta C_{si}^{+}}$

ddelta-cs_dcenter: $\frac{\partial \Delta C_{si}^{+}}{\partial {}^{c}X_i^{+}}$

ddelta_cs_dtheta: $\frac{\partial \Delta C_{si}^{+}}{\partial \theta}$

gradient: $\sum_{i=1}^{nc1} \frac{\partial \ln(PCD_i|\theta))}{\partial \Delta C_{si}^{+}} \frac{\partial \Delta C_{si}^{+}}{\partial {}^{c}X_i^{+}} \frac{\partial {}^{c}X_i^{+}}{\partial \theta}\Big|_{\theta=0}$

hessian:

$$\sum_{i=1}^{nc1} -\frac{1}{6\tilde{\Delta C}_{si}^{2}} \left(\frac{\partial \Delta C_{si}^{+}}{\partial {}^{c}X_i^{+}} \frac{\partial {}^{c}X_i^{+}}{\partial \theta}\right)^{T} \left(\frac{\partial \Delta C_{si}^{+}}{\partial {}^{c}X_i^{+}} \frac{\partial {}^{c}X_i^{+}}{\partial \theta}\right)\Big|_{\theta=0}$$

$$\hat{\theta} = (-H + \begin{bmatrix} \lambda_{rl3} & 0 \\ 0 & \lambda_{tl3} \end{bmatrix})^{-1} g$$

$${}^{cam}_{body}T = {}^{cam}_{body}T \cdot \Delta T$$

SRT3D: A Sparse Region-Based 3D Object Tracking Approach for the Real World (2021)

Images: $I: \Omega \to \{0, \dots, 255\}^3$

$\Omega \subset \mathbb{R}^2$ image domain

correspondence line: $l: \omega \to \{0, \dots, 255\}^3$

$\omega \subset \mathbb{R}$, line domain

image value: $y = I(x)$, $x = [x \; y]^T$

$y = l(r)$, $r \subset \mathbb{R}$ in line.

$= I(c + rn)$

$c = [c_x, c_y]^T$ center, $n = [n_x \; n_y]^T$ normal vector

$\|n\|_2 = 1$

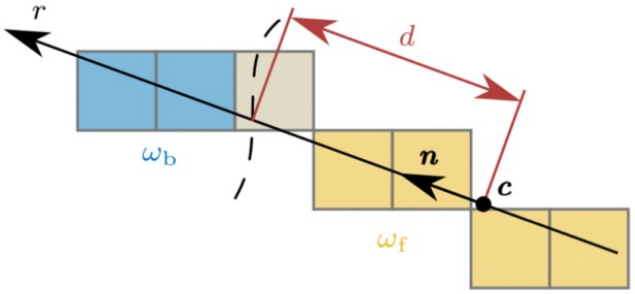
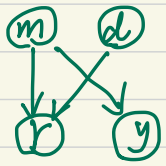


Fig. 2 Correspondence line defined by a center $\boldsymbol{c}$ and a normal vector $\boldsymbol{n}$. The illustration shows pixels along the correspondence line as well as the foreground region ω_f in yellow and the background region ω_b in blue. The contour distance d points from the correspondence line center to an estimated contour, indicated by a dashed line.



{r, y} is a sample

1. For a single pixel

$$P(r, y, d, m) = P(r, y \mid d, m) P(d, m) = P(r \mid d, m) P(y \mid d, m) P(d, m)$$
$$= P(r \mid d, m) P(y \mid m) P(d) P(m)$$

color ↑ distance ↑ model

distance from the correspondence line center c to an estimated contour.

$$P(r,y,d,m) = P(r|d,m)\,P(y|m)\,P(d)\,P(m)$$

$$P(r,d,m|y)\,P(y) = P(r|d,m)\,P(y|m)\,P(d)\,P(m)$$

$$P(r,d,m|y) = P(r,|d,m)\left[\frac{P(y|m)\,P(m)}{P(y)}\right]P(d)$$

$P(m|y)\,P(y) = P(y|m)\,P(m)$

$$= P(r,|d,m)\,P(m|y)\,P(d)$$

conditional line coordinate probabilities

$$\because\ P(m_i|y) = \frac{P(y|m_i)\,P(m_i)}{\sum_{j\in\{f,b\}} P(y|m_j)\,P(m_j)},\quad i\in\{f,b\}$$

$P(m_i|y)$: the probability distributions that describe how likely it is that a specific color value is part of the foreground / background region.

Foreground and background are equally likely along the correspondence line (locally), $P(m_f) = P(m_b)$

$$\therefore\ P(m_i|y) = \frac{P(y|m_i)}{P(y|m_f)+P(y|m_b)},\quad i\in\{f,b\}$$

$$P(x) = \sum_y P(x, Y=y) = \sum_y P(x|Y=y)\,P(Y=y)$$

$$P(d|r,y)\,P(r,y) = P(d,r,y) = P(d|r,y)\,P(r)\,P(y)$$

$$P(d|r,y) = \frac{1}{P(r)}\frac{P(d,r,y)}{P(y)} = \frac{1}{P(r)}\frac{P(d,r|y)\,P(y)}{P(y)}$$

$$= \frac{1}{P(r)}P(d,r|y) = \frac{1}{P(r)}\sum_{i\in\{f,b\}} P(d,r,m_i|y)$$

$$= \frac{1}{P(r)}\sum_{i\in\{f,b\}} P(r,|d,m_i)\,P(m_i|y)\,P(d)$$

$P(r,|d,m_i)$: d, m_i yield r is probability

Posterior probability over the entire correspondence line domain w:

$$P(d|w,l) = \prod_{r\in w}\sum_{i\in\{f,b\}}\frac{1}{P(r)}P(r|d,m_i)\,P(m_i|y)\,P(d)$$

The probability of the contour distance d given data from a correspondence line.

$$\propto \prod_{r\in w}\sum_{i\in\{f,b\}} P(r|d,m_i)\,P(m_i|l(r))$$

$l(r) = I(c+rn) = y$, $P(r)$ and $P(d)$ are considered to be uniform and constant and are thus dropped.

Estimating the $P(d|w,l)$ is computationally expensive since, for each distance d, the product of $P(d|w,l)$ has to be computed over the entire domain w.

2. Discrete scaled space formulation combines multiple pixels into segments and projects from continuous space along the correspondence line onto a discrete space that is indepent of a correspondence line's location and orientation.

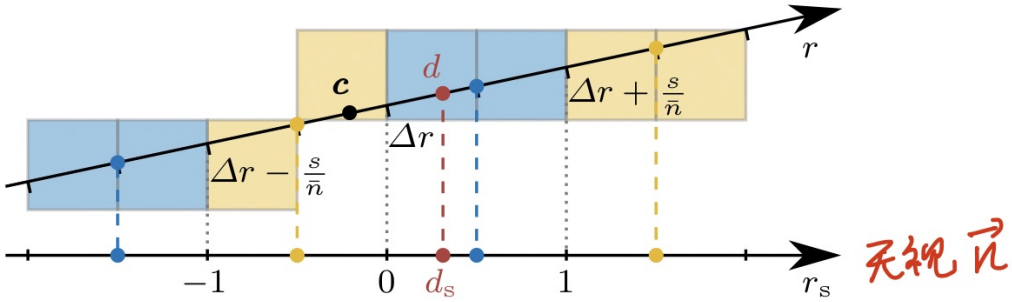


Fig. 3 Example of the relation between the unscaled space r along the correspondence line and the scale-space r_s. Neighboring pixels that are combined into segments are visualized by the same color in blue or yellow. Blue and yellow dots indicate the center of each segment and the corresponding discretized value in the scale-space. An example of the contour distance is illustrated in red. The offset Δr is chosen in a way that ensures that discretized values in the scale-space are the same for all correspondence lines. In this example, Δr points to the closest edge between pixels.

$$r_s = (r - \Delta r)\frac{\bar{n}}{s}, \qquad s \in \mathbb{N}^+ : \text{the number of pixels combined into a segment.}$$
$$d_s = (d - \Delta r)\frac{\bar{n}}{s} \qquad \bar{n} = \max(|n_x|, |n_y|)$$

∴ The posterior probability in the discrete scale-space:

$$P(d|w,l) \propto \prod_{r \in w} \sum_{i \in \{f,b\}} P(r|d,m_i)\, P(m_i|l(r))$$

$$\Rightarrow P(d_s|w_s,l_s) \propto \prod_{r_s \in w_s} \sum_{i \in \{f,b\}} P(r_s|d_s,m_i)\, P(m_i|\underbrace{l_s(r_s)}_{=s})$$

scaled correspondence line domain

$s = l_s(r_s)$ is a set-valued function that maps from the scaled line coordinate r_s to the segment s, which is a set of the closest s pixel values y.

$$\because\ P(m_i|y) = \frac{P(y|m_i)}{\sum_{j\in\{f,b\}} P(y|m_j)},\quad i\in\{f,b\}$$

$\because P(m_b)=P(m_f)$

$$P(m_i|y_1,y_2,y_3) = \frac{P(y_1,y_2y_3|m_i)P(m_i)}{P(y_1,y_2,y_3)} = \frac{P(y_1|m_i)P(y_2|m_i)P(y_3|m_i)P(m_i)}{\sum_{j\in\{f,b\}} P(y_1y_2y_3|m_j)P(m_j)}$$

$$= \frac{P(y_1|m_i)P(y_2|m_i)P(y_3|m_i)}{P(y_1y_2y_3|m_f)+P(y_1y_2y_3|m_b)}$$

$$= \frac{P(y_1|m_i)P(y_2|m_i)P(y_3|m_i)}{P(y_1y_2y_3|m_f)+P(y_1y_2y_3|m_b)}$$

$$= \frac{P(y_1|m_i)P(y_2|m_i)P(y_3|m_i)}{P(y_1|m_f)P(y_2|m_f)P(y_3|m_f)+P(y_1|m_b)P(y_2|m_b)P(y_3|m_b)}$$

$$\therefore P(m_i|S) = \frac{\prod_{y\in S} P(y|m_i)}{\prod_{y\in S}P(y|m_f)+\prod_{y\in S}P(y|m_b)}$$

$P(m_i|S)$: S — a set of the closest s pixel values y

$$P(d_S|w_S,l_S) \propto \prod_{r_S\in w_S}\sum_{i\in\{f,b\}} P(r_S|d_S,m_i)\,P(m_i|\underbrace{s(r_S)}_{S})$$

① $P(m_i|y_j) = \frac{P(y_j|m_i)P(m_i)}{P(y_j|m_f)P(m_f)+P(y_j|m_b)P(m_b)}$ If color y_j do not sample, $P(m_i|y_j)=0.5$

② $P_{m_i} = P(m_i|y_1)P(m_i|y_2)P(m_i|y_3)\cdots$

③ $$P(m_i|S) = \frac{P_{m_i}}{P_{m_f}+P_{m_b}}$$

$$= \frac{\frac{P(y_1|m_i)}{[P(y_1|m_f)+P(y_1|m_b)]}\,\frac{P(y_2|m_i)}{[P(y_2|m_f)+P(y_2|m_b)]}\,\frac{P(y_3|m_i)}{[P(y_3|m_f)+P(y_3|m_b)]}}{\frac{P(y_1|m_f)}{[P(y_1|m_f)+P(y_1|m_b)]}\,\frac{P(y_2|m_f)}{[P(y_2|m_f)+P(y_2|m_b)]}\,\frac{P(y_3|m_f)}{[P(y_3|m_f)+P(y_3|m_b)]} + \frac{P(y_1|m_b)}{[P(y_1|m_f)+P(y_1|m_b)]}\,\frac{P(y_2|m_b)}{[P(y_2|m_f)+P(y_2|m_b)]}\,\frac{P(y_3|m_b)}{[P(y_3|m_f)+P(y_3|m_b)]}}$$

$$= \frac{\prod_{y\in S} P(y|m_i)}{\prod_{y\in S}P(y|m_f)+\prod_{y\in S}P(y|m_b)}$$

① If $P(m_b|y_1)$ $P(m_b|y_2)$ $P(m_b|y_3)$
0.5 1 0.1

$P(m_f|y_1)$ $P(m_f|y_2)$ $P(m_f|y_3)$
0.5 0 0.9

$$P(m_b|S) = \frac{0.5 \cdot 1 \cdot 0.1}{0.5 \cdot 1 \cdot 0.1 + 0} = 1$$

② $$P(m_b|S) = \frac{0.0001 \times 0.4 \times 0.1}{0.0001 \times 0.4 \times 0.1 + 0.0001 \times 0.0001 \times 0.3} = \frac{0.4}{0.4 + 0.001 \times 0.3} = \frac{0.1}{0.1 + 1e^{-4} \times 0.2} \approx 1$$

Laplace smoothing

Laplace smoothing is a technique to smooth categorical data. Given a set of observation counts $X = \{x_1, x_2, \ldots x_d\}$ from a d-dim multinomial distribution with N trials, a "smoothed" version of counts gives the est:

$$\hat{\theta}_i = \frac{x_i + \alpha}{N + \alpha d} \quad (i = 1, \ldots, d)$$

It can address the problem when we encounter a unseen data which the posterior probility will become 0.

ex. x_i is the number of color y observation times.

$\therefore P(y|m_i) =$ the number of pixel which value y (x_i) / total sample pixel numbers (N)

→ smooth $\hat{P}(y|m_i) =$ the number of pixel which value y $(x_i) + 1$ / total sample pixel numbers (N) + the total number of y values $(32768 = (256/8)^3)$

Underflow

If $P(y|m_i)$ is small, $\prod_{y \in S} P(y|m_i)$ can produce numbers too small for floating point storage.

$$P(m_i|S) = \frac{\prod_{y \in S} P(y|m_i)}{\prod_{y \in S} P(y|m_f) + \prod_{y \in S} P(y|m_b)}$$

$$\prod_{y \in S} P(y|m_i) = \exp(\log(\prod_{y \in S} P(y|m_i))) = \exp(\sum_{y \in S} \log(P(y|m_i)))$$

3. $p(r|d, m_f)$, $p(r|d, m_b)$ models a local uncertainty with respect to the exact location of the foreground and background transition.

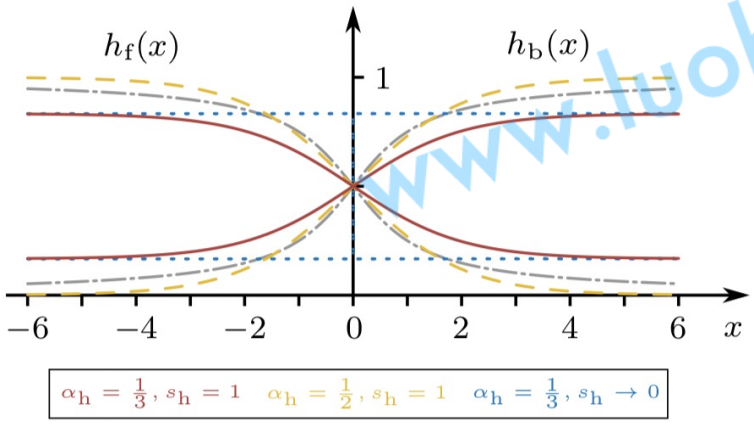


Fig. 4 Smoothed step functions h_f and h_b that model the conditional line coordinate probabilities $p(r \mid d, m_f)$ and $p(r \mid d, m_b)$. The functions $h_f(x) = \frac{1}{2} - \frac{1}{\pi}\tan^{-1}\left(\frac{x}{s_h}\right)$ and $h_b(x) = \frac{1}{2} + \frac{1}{\pi}\tan^{-1}\left(\frac{x}{s_h}\right)$ used by Zhong et al. (2020b) and Tjaden et al. (2018) are illustrated by dash-dotted gray lines. The definitions $h_f(x) = \frac{1}{2} - \frac{1}{2}\tanh\left(\frac{x}{2s_h}\right)$ and $h_b(x) = \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{x}{2s_h}\right)$ from our previous work (Stoiber et al., 2020) are shown as dashed yellow lines. In both plots, the slope parameter $s_h = 1$ is used. For the proposed functions in Eq. (12) and (13), solid red lines correspond to $\alpha_h = \frac{1}{3}$ and $s_h = 1$, while dotted blue lines show the functions for $\alpha_h = \frac{1}{3}$ and $s_h \to 0$. In addition to visualizing the definitions from our previous work, the dashed yellow lines illustrate the proposed functions for $\alpha_h = \frac{1}{2}$ and $s_h = 1$.

$h_i(x) = P(r \mid d, m_i)$

$$\begin{cases} h_f(x) = \frac{1}{2} - a_n \tanh(x/2s_n) \\ h_b(x) = \frac{1}{2} + a_n \tanh(x/2s_n) \end{cases}$$

$a_n \in [0, 0.5]$
$s_n \in R^+$

$P(m_f) = P(m_b) = a_n$ ☆ a_n的含义

$P(m_n) = 1 - 2a_n$

Suppose $m \in \{m_f, m_b, m_n\}$ (m_n: noise)

$P(m_f) = P(m_b) = a_n$, $a_n \in [0, 0.5]$, and the probability of a noise m_n

$P(m_n) = 1 - 2a_n$

$$\because P(m_i \mid y) = \frac{P(y \mid m_i) P(m_i)}{\sum_{j \in \{f,b,n\}} P(y \mid m_j) P(m_j)} \quad i \in \{f, b, n\}$$

We define the conditional color probability given the noise model:

$$P(y \mid m_n) = \frac{1}{2}(P(y \mid m_f) + P(y \mid m_b))$$

$$\therefore P(m_i \mid y) = \frac{P(y \mid m_i) P(m_i)}{P(y \mid m_n) P(m_n) + P(y \mid m_f) P(m_f) + P(y \mid m_b) P(m_b)}$$

$$= \frac{P(y \mid m_i) P(m_i)}{P(y \mid m_n)(1 - 2a_n) + P(y \mid m_f) a_n + P(y \mid m_b) a_n}$$

$$= \frac{P(y \mid m_i) P(m_i)}{P(y \mid m_n)(1 - 2a_n) + P(y \mid m_f) a_n + P(y \mid m_b) a_n}$$

$$= \frac{P(y \mid m_i) P(m_i)}{\frac{1}{2}(P(y \mid m_f) + P(y \mid m_b))(1 - 2a_n) + a_n(P(y \mid m_f) + P(y \mid m_b))}$$

$$= \frac{2P(y \mid m_i) P(m_i)}{P(y \mid m_f) + P(y \mid m_b)}$$

For foreground and background:

$$P(m_i \mid y) = \frac{2a_n P(y \mid m_i)}{P(y \mid m_f) + P(y \mid m_b)}, \quad i \in \{f, b\}.$$

$$= 2a_n P_i(r)$$

For noise:

$$P(m_n|y) = \frac{2P(y|m_n)P(m_n)}{P(y|m_f)+P(y|m_b)} = \frac{2\cdot\frac{1}{2}(P(y|m_f)+P(y|m_b))P(m_n)}{P(y|m_f)+P(y|m_b)}$$

$$= P(m_n)$$
$$= 1-2a_n$$

$\because P(d|r,y) \propto \sum_{i\in\{f,b\}} P(r,|d,m_i)\,P(m_i|y)$, $h_i(x) = P(r|d,m_i)$

$$P_i(r) = \frac{P(y|m_i)}{P(y|m_f)+P(y|m_b)}$$

we define $P(r|d,m_n) = \frac{1}{2}$ for noise.

$\therefore P(d|r,y) \propto \sum_{i\in\{f,b,n\}} P(r|d,m_i)P(m_i|y)$

$$= 2a_n h_f(r-d)P_f(r) + 2a_n h_b(r-d)P_b(r) + \frac{1}{2}(1-2a_n)$$

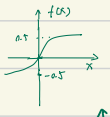
Let ① $h_f(x) = \frac{1}{2} - f(x)$
② $h_b(x) = \frac{1}{2} + f(x)$
③ $P_f(r) + P_b(r) = 1$

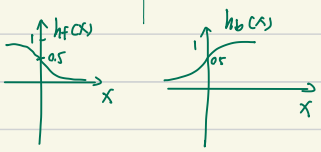
Suppose:
① $h_f + h_b = 1$
② h_f and h_b be symmetric.

$\therefore h_f(x) = 0.5 - f(x)$, $h_b(x) = 0.5 + f(x)$

where $f(x)$ is an odd function ($-f(x) = f(-x)$) that lies within the interval $[-0.5, 0.5]$ and

$\lim_{x\to\infty} f(x) = 0.5$ and $\lim_{x\to-\infty} f(x) = -0.5$





$\therefore P(d|r,y) \propto 2a_n(\frac{1}{2} - f(r-d))P_f(r)$
$+ 2a_n(\frac{1}{2} + f(r-d))P_b(r)$
$+ \frac{1}{2}(1-2a_n)(P_f(r)+P_b(r))$

$= a_n P_f(r) - 2a_n f(r-d)P_f(r)$
$a_n P_b(r) + 2f(r-d)P_b(r)$
$\frac{1}{2}(P_f(r)+P_b(r)) - a_n(P_f(r)+P_b(r))$

$= (\frac{1}{2} - 2a_n f(r-d))P_f(r) +$
$(\frac{1}{2} + 2a_n f(r-d))P_b(r)$

$\because f(x) = \frac{1}{2}\tanh\left(\frac{x}{2s_h}\right)$

$\therefore p(d|r,y) \propto \left(\frac{1}{2} - a_h \tanh\left(\frac{r-d}{2s_h}\right)\right) p_f(r) + \left(\frac{1}{2} + a_h \tanh\left(\frac{r-d}{2s_h}\right)\right) p_b(r)$

Extending the probabilistic model with a noise model m_n with a defined constant uncertainty and using the foreground and background probabilities $p(m_f) = p(m_b) = a_h$ is equivalent to the introduction of a simple amplitude parameter a_h into the smoothed step functions.

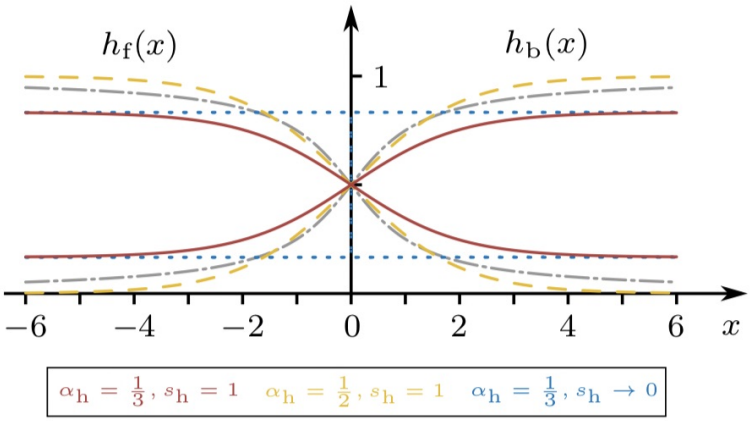


Fig. 4 Smoothed step functions h_f and h_b that model the conditional line coordinate probabilities $p(r \mid d, m_f)$ and $p(r \mid d, m_b)$. The functions $h_f(x) = \frac{1}{2} - \frac{1}{\pi}\tan^{-1}\left(\frac{x}{s_h}\right)$ and $h_b(x) = \frac{1}{2} + \frac{1}{\pi}\tan^{-1}\left(\frac{x}{s_h}\right)$ used by Zhong et al. (2020b) and Tjaden et al. (2018) are illustrated by dash-dotted gray lines. The definitions $h_f(x) = \frac{1}{2} - \frac{1}{2}\tanh\left(\frac{x}{2s_h}\right)$ and $h_b(x) = \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{x}{2s_h}\right)$ from our previous work (Stoiber et al., 2020) are shown as dashed yellow lines. In both plots, the slope parameter $s_h = 1$ is used. For the proposed functions in Eq. (12) and (13), solid red lines correspond to $\alpha_h = \frac{1}{3}$ and $s_h = 1$, while dotted blue lines show the functions for $\alpha_h = \frac{1}{3}$ and $s_h \to 0$. In addition to visualizing the definitions from our previous work, the dashed yellow lines illustrate the proposed functions for $\alpha_h = \frac{1}{2}$ and $s_h = 1$.

4. $\because$ ① $p(d|w,I) \propto \prod_{r \in w} \sum_{i \in \{f,b\}} p(r|d,m_i)\, p(m_i|I(r))$

② $h_i(r-d) = p(r|d,m_i)$ $\quad \begin{cases} h_f(x) = \frac{1}{2} - a_h \tanh(x/2s_h) \\ h_b(x) = \frac{1}{2} + a_h \tanh(x/2s_h) \end{cases}$

③ $p_i(r) = p(m_i|I(r)) = p(m_i|y) = \frac{p(y|m_i)}{p(y|m_f) + p(y|m_b)}, \; i \in \{f,b\}$

$I(r) = I(c+rn) = y$

$\therefore p(d|w,I) \propto \prod_{r \in w} h_f(r-d)\, p_f(r) + h_b(r-d)\, p_b(r)$

To analyze the posterior probability $P(d|w,L)$ distribution,

we assume a contour at the line center c and perfect step function $p_i(x)$ for the pixel-wise posterior probabilities defined by:

$$\begin{cases} p_f(r) = \frac{1}{2} - \frac{1}{2}\,\mathrm{sgn}(r) \\ p_b(r) = \frac{1}{2} + \frac{1}{2}\,\mathrm{sgn}(r) \end{cases}$$

we consider infinitesimally samall pixels and write the posterior probability distribution $p(d|w,L)$ in continuous form for an infinite correspondance line:

The classical Riemann integral of a function $f: [a,b] \to \mathbb{R}$:

$$\int_a^b f(x)\,dx = \lim_{\Delta x \to 0} \sum f(x_i)\Delta x$$

Product integrals are similar, but take the limit of a product instead of the limit of sum. They can be thought of as "continuous" version of "discrete" products.

One kind of Product Integrals is Geometric Integral:

$$\prod_a^b f(x)^{dx} = \lim_{\Delta x \to 0} \prod f(x_i)^{\Delta x} = \exp\left(\int_a^b \ln f(x)\,dx\right)$$

$$p(d|w,L) \propto \prod_{r=-\infty}^{\infty} \left(h_f(r-d)\,p_f(r) + h_b(r-d)\,p_b(r)\right)^{dr}$$

$$= \exp\left(\int_{r=-\infty}^{\infty} \ln\left(h_f(r-d)\,p_f(r) + h_b(r-d)\,p_b(r)\right)dr\right)$$

$$= \exp\left(\int_{r=-\infty}^{0} \ln\left(h_f(r-d)\,\underbrace{p_f(r)}_{1} + h_b(r-d)\,\underbrace{p_b(r)}_{0}\right)dr\right.$$

$$\left. + \int_{r=0}^{\infty} \ln\left(h_f(r-d)\,\underbrace{p_f(r)}_{0} + h_b(r-d)\,\underbrace{p_b(r)}_{1}\right)dr\right)$$

$$= \exp\left(\int_{r=-\infty}^{0} \ln\left(h_f(r-d)\right)dr + \right.$$

$$\left. \int_{r=0}^{\infty} \ln\left(h_b(r-d)\right)dr\right)$$

Let $x = r - d$ (↙ $b(t)$, $t =$)

$$\therefore p(d|w,L) \propto \exp\left(\int_{x=-\infty}^{-d} \ln(h_f(x))\,dx + \int_{x=-d}^{\infty} \ln(h_b(x))\,dx\right)$$

(↑ $a(t)$)

$$\therefore \frac{\partial \ln(P(d|w,L))}{\partial d} = \frac{\partial\left(\int_{x=-\infty}^{-d} \ln(h_f(x))\,dx\right)}{\partial d} + \frac{\partial\left(\int_{x=-d}^{\infty} \ln(h_b(x))\,dx\right)}{\partial d}$$

Leibniz Rule: If $\Phi(t) = \int_{a(t)}^{b(t)} f(x,t)\,dx$, then

$$\frac{d\Phi}{dt} = \int_{a(t)}^{b(t)} \frac{\partial f(x,t)}{\partial t}\,dx + f(b(t),t)\frac{db}{dt} - f(a(t),t)\frac{da}{dt}$$

Let $d \leftarrow t$, $-d \leftarrow b(t)$, $x = -\infty \leftarrow a(t)$

$\ln(h_f(x)) \leftarrow f(x,t)$

$$\therefore \frac{\partial\left(\int_{x=-\infty}^{-d} \ln(h_f(x))\,dx\right)}{\partial d}$$

$$= \int_{x=-\infty}^{-d} \frac{\partial \ln(h_f(x))}{\partial d}\,dx - \ln(h_f(-d)) + 0$$

$$= -\ln(h_f(-d))$$

$$\therefore \frac{\partial\left(\int_{x=-d}^{\infty} \ln(h_b(x))\,dx\right)}{\partial d}$$

$$= 0 + 0 + \ln(h_b(-d))$$

$$\frac{\partial \ln(P(d|w,L))}{\partial d} = -\ln(h_f(-d)) + \ln(h_b(-d))$$

$\because \quad h_f(x) = \frac{1}{2} - a_h \tanh(x/2s_h)$

$h_b(x) = \frac{1}{2} + a_h \tanh(x/2s_h)$

$$\therefore \frac{\partial \ln(P(d|w,L))}{\partial d} = -\ln\left(\frac{1}{2} - a_h \tanh\left(\frac{-d}{2s_h}\right)\right) + \ln\left(\frac{1}{2} + a_h \tanh\left(\frac{-d}{2s_h}\right)\right)$$

$\tanh^{-1} = \operatorname{arctanh}$

$\because 2\tanh^{-1}(x) = -\ln(1-x) + \ln(1+x)$

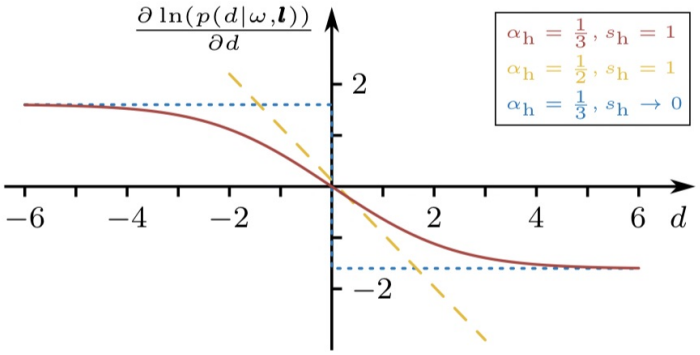
$\tanh(-x) = -\tanh(x)$

$$\therefore \frac{\partial \ln(P(d|w,L))}{\partial d} = -\ln\left(\frac{1}{2}\left(1 - 2a_h \tanh\left(\frac{-d}{2s_h}\right)\right)\right) + \ln\left(\frac{1}{2}\left(1 + 2a_h \tanh\left(\frac{-d}{2s_h}\right)\right)\right)$$

$$= -\ln\left(1 - 2a_h \tanh\left(\frac{-d}{2s_h}\right)\right) + \ln\left(1 + 2a_h \tanh\left(\frac{-d}{2s_h}\right)\right)$$

$$= 2\tanh^{-1}\left(2a_h \tanh\left(\frac{-d}{2s_h}\right)\right)$$

$$\frac{\partial \ln(P(d|w,L))}{\partial d} = -2\tanh^{-1}\left(2a_h \tanh\left(\frac{d}{2s_h}\right)\right)$$



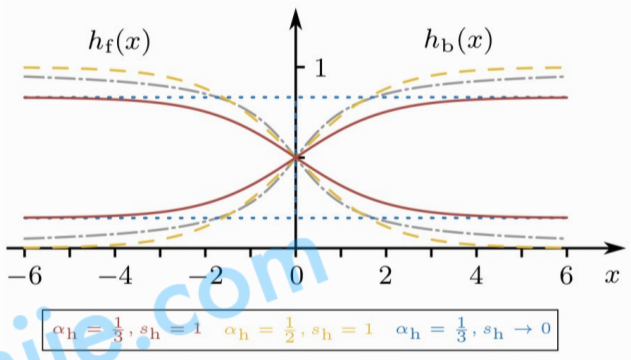


Fig. 5 First-order derivatives of the log-posterior with respect to the contour distance d for different slope and amplitude parameters s_h and α_h. The solid red line shows the derivative for $\alpha_h = \frac{1}{3}$ and $s_h = 1$, which yields a function with a smooth transition from an upper bound to a lower bound. The dashed yellow line shows the function for $\alpha_h = \frac{1}{2}$ and $s_h = 1$. This produces a linear first-order derivative. Finally, using $\alpha_h = \frac{1}{3}$ and $s_h \to 0$ results in a perfect step function illustrated by the dotted blue line.

① If $a_n = \frac{1}{2}$

PDF: $f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$

$$\frac{\partial \ln(P(d|w,L))}{\partial d} = -\frac{d}{S_n} = \frac{\partial \ln(N(d|0,S_n))}{\partial d}$$

$$N(d|0,S_n) = \frac{1}{\sqrt{2\pi S_n}} \exp\left(-\frac{d^2}{2S_n}\right)$$

The first-order derivative of the log-likelihood of $P(d|w,L)$ is equal to the first-order derivative of the logarithm of the normal distribution $N(d|0,S_n)$

$$\therefore P(d|w,L) \propto \exp\left(-\frac{d^2}{2S_n}\right)$$

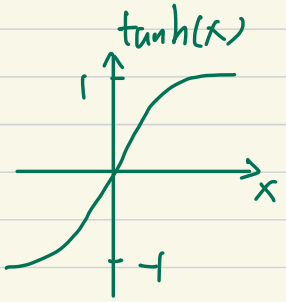
Because the PDF of $P(d|w,L)$ that integrates to one, the final solution can only be the Gaussian distribution:

$$\text{If } a_n = \frac{1}{2}, \ln P(d|w,L) = \frac{1}{\sqrt{2\pi S_n}} \exp\left(-\frac{d^2}{2S_n}\right)$$

② If $S_n \to 0$

$$\frac{\partial \ln(P(d|w,L))}{\partial d} = -2\tanh^{-1}\left(2a_n \tanh\left(\frac{d}{2S_n}\right)\right)$$

$$\Rightarrow = -2\tanh^{-1}(2a_n)\,\mathrm{sgn}(d)$$



$$P(d|w,L) \propto \exp\left(\int \frac{\partial \ln(P(d|w,L))}{\partial d} \Delta d\right)$$

$$= \exp\left(\int -2\tanh^{-1}(2a_n)\,\mathrm{sgn}(d)\,\Delta d\right)$$

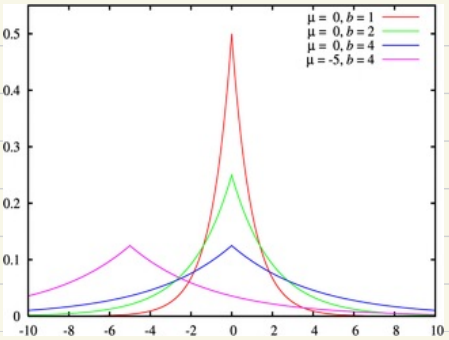
$$= \exp\left(-2\tanh^{-1}(2a_n)\,|d|\right) + C$$

$$= \exp\left(-\frac{|d|}{\frac{1}{2\tanh^{-1}(2a_n)}}\right) + C$$

$\frac{d|x|}{dx} = \mathrm{sgn}(x)$ for $x \neq 0$

Laplace distribution:

$$f(x|\mu,b) = \frac{1}{2b}\exp\left(-\frac{|x-\mu|}{b}\right) = \frac{1}{2b}\begin{cases}\exp\left(-\frac{\mu-x}{b}\right), & \text{if } x < \mu \\ \exp\left(-\frac{x-\mu}{b}\right), & \text{if } x \geq \mu\end{cases}$$



With the exception of a constant scaling factor, the function is equal to a Laplace distribution, which again is a valid PDF that integrates to one.

If $s_h \to 0$, $p(d|w,l) = \frac{1}{2b}\left(-\frac{|d|}{b}\right)$, $b = \frac{1}{2\tanh^{-1}(2a_h)}$

The variance of Laplace distribution : $2b^2$

$$= \frac{1}{2\left(\tanh^{-1}(2a_h)\right)^2}$$

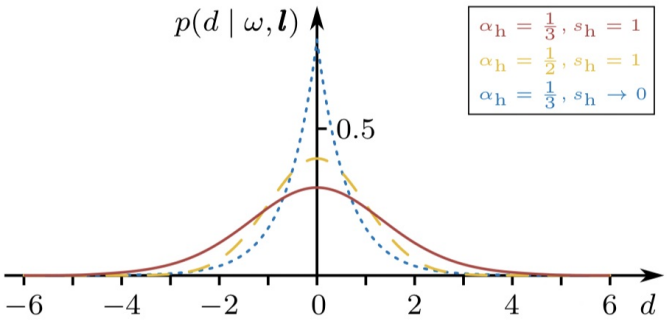


Fig. 6 Posterior probability distributions for different slope and amplitude parameters s_h and α_h. The solid red line shows the function for $\alpha_h = \frac{1}{3}$ and $s_h = 1$, which leads to a very flat distribution. Note that the function was computed numerically. Using $\alpha_h = \frac{1}{2}$ and $s_h = 1$ results in a Gaussian distribution shown by the dashed yellow line. The parameters $\alpha_h = \frac{1}{3}$ and $s_h \to 0$ yield a Laplace distribution for the posterior probability that is illustrated by a dotted blue line.

The slope parameter s_h controls local uncertainty, allowing multiple values d to be almost equally likely.

The amplitude parameter a_h controls the size of the peak compared to its surroundings, thereby controlling global uncertainty.

In the prefect situation,

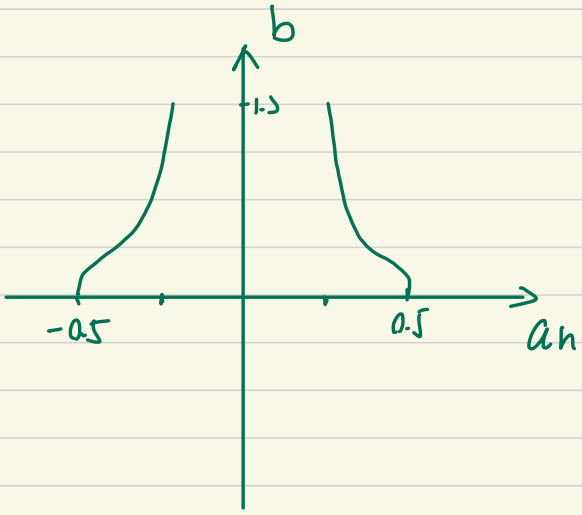
If $a_h = \frac{1}{2}$, $\ln p(d|w,l) = N(d|0, s_h) = \frac{1}{\sqrt{2\pi s_h}} \exp\left(-\frac{d^2}{2s_h}\right)$

$\sigma^2_{Gauss} = s_h$

If $s_h \to 0$, $p(d|w,l) = \frac{1}{2b}\left(-\frac{|d|}{b}\right)$, $b = \frac{1}{2}\tanh(2a_h)$

$\sigma^2_{Laplace} = 2b^2 = \frac{1}{2}\tanh^2(2a_h)$

∴ In the normal situation, the variance of $N(d_{s_i} | \mu_i, \sigma_i^2)$, σ_i^2 should larger than σ^2_{Gauss} and $\sigma^2_{Laplace}$.



5. 3D model points : $X=[X\ Y\ Z]^T$

$\tilde{X}=[X\ Y\ Z\ 1]^T$

$$x=\pi(X)=\begin{bmatrix}\frac{X}{Z}f_x+p_x\\ \frac{Y}{Z}f_y+p_y\end{bmatrix}$$

$$X=\pi^{-1}(x,d_z)=d_z\begin{bmatrix}\frac{x-p_x}{f_x}\\ \frac{y-p_y}{f_y}\\ 1\end{bmatrix}$$

$$ {}_c\tilde{X} = {}_cT_M \; {}_M\tilde{X} = \begin{bmatrix} {}_cR_M & {}_ct_M \\ 0 & 1 \end{bmatrix} {}_M\tilde{X} $$

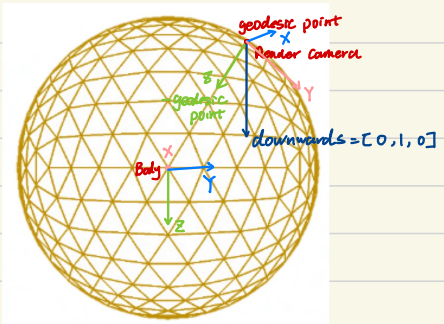
$$ R = \exp([r]_\times) = I + [r]_\times + \frac{1}{2!}[r]_\times^2 + \frac{1}{3!}[r]_\times^3 + \cdots $$

$$ {}_c\tilde{X}(\theta) = \begin{bmatrix} {}_cR_M & {}_ct_M \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I + [\theta_r] & \theta_t \\ 0 & 1 \end{bmatrix} {}_M\tilde{X} $$

$\theta_r \in \mathbb{R}^3, \; \theta_t \in \mathbb{R}^3, \; \theta^T = [\theta_r^T \; \theta_t^T]$

6. The 3D geometry is rendered from a number of n_v viewpoints all around the object. Virtual cameras are placed on the vertices of a geodesic grid. For each rendering, n_c points $x_i \in \mathbb{R}^2$ are randomly sampled from the contour. $n_i \in \mathbb{R}^2$ is the normal is computed, $\|n_i\|_2 = 1$. 3D vectors with respect to the model frame:

$$ {}_M\tilde{X}_i = {}_MT_C \, \tilde{\pi}^{-1}(x_i, d_{Z_i}) $$
$$ {}_MN_i = {}_MR_C \begin{bmatrix} n_i \\ 0 \end{bmatrix} $$



(c) Geodesic Grid

orientation vector $M^v = {}_MR_C \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

Given a specific pose ${}_MR_C$, ${}_ct_M$. the closest precomputed view i_v:

$$ i_v = \underset{i \in \{1,\ldots,n_v\}}{\arg\max} \; (M_{V_i}^T \, {}_MR_C \, {}_ct_M) $$

7. 3D model points and normal vectors from the closest view of the sparse view are projected into the image:

$$ c_i = \pi({}_cT_M \, {}_M\tilde{X}_i) $$
$$ n_i \propto ({}_cR_M \, {}_MN_i)_{2\times 1}, \quad \|n_i\|_2 = 1 $$

Contour distances are calculated as the distances along normal vectors n_i from correspondence line centers c_i to projected 3D model points X_i

$$ d_i(\theta) = n_i^T[\pi({}_cX_i(\theta)) - c_i) $$

$$ {}_c\tilde{X}(\theta) = \begin{bmatrix} {}_cR_M & {}_ct_M \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I + [\theta_r] & \theta_t \\ 0 & 1 \end{bmatrix} {}_M\tilde{X} $$

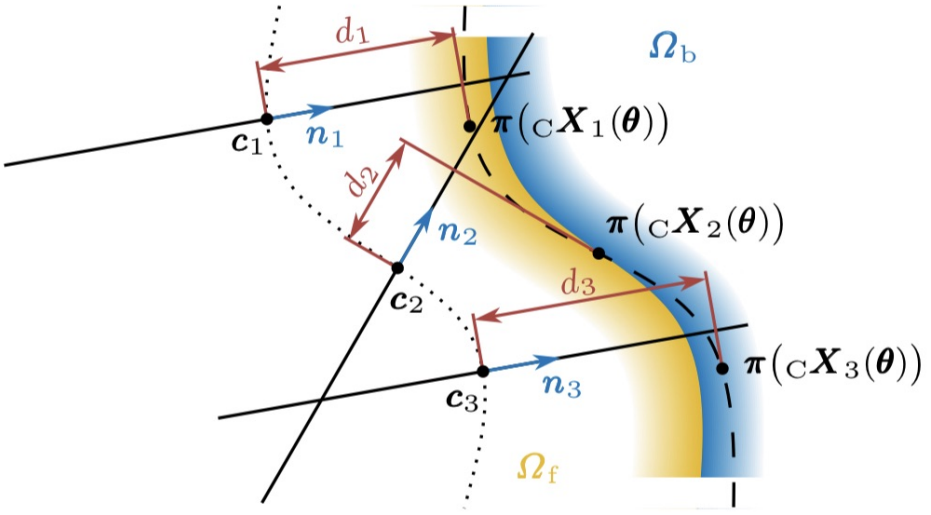


Fig. 8 Correspondence lines defined by a center $\boldsymbol{c}_i$ and a normal vector $\boldsymbol{n}_i$. Variated contour distances d_i are measured along the correspondence lines from the centers $\boldsymbol{c}_i$ to the projected 3D model points ${}_C\boldsymbol{X}_i$ that depend all on the same pose variation $\boldsymbol{\theta}$. The object contour of the original pose estimate, which was used to define the correspondence lines, is indicated by a dotted line. The current estimate of the contour that depends on the pose variation vector $\boldsymbol{\theta}$ is shown by a dashed line. The ground truth segmentation that we try to estimate is given by the foreground region $\boldsymbol{\Omega}_f$ in yellow and the background region $\boldsymbol{\Omega}_b$ in blue. Note that while contours are illustrated as continuous lines, in our method, they are represented by points and normal vectors from the closest view of the sparse viewpoint model.

Assuming a number of n_c independent correspondence lines.

$P(\theta|D) \propto \prod_{i=1}^{n_c} P(d_{s_i}(\theta)|w_{s_i}, l_{s_i})$ (24), (n_c: num of sample points)

D : the data from all correspondence lines $= S$

① $P(d_S|w_S, l_S) \propto \prod_{r_S \in w_S} \sum_{i \in \{f,b\}} P(r_S|d_S, m_i)\, P(m_i|l_S(r_S))$ (7) (10)

$w \to \{0, \dots, 255\}$, $w \subset \mathbb{R}$, line domain. w_S is the scaled correspondence line domain.

② $P(r_S|d_S, m_i) = h_i(r_S - d_S) = \begin{cases} \frac{1}{2} - a_h \tanh[(r_S - d_S)/2s_h], & i = f \\ \frac{1}{2} + a_h \tanh[(r_S - d_S)/2s_h], & i = b \end{cases}$ (3)

③ $r_S = (r - \Delta r)\frac{\bar{n}}{s}$, $s \in \mathbb{N}^+$: the number of pixels combined into a segment. (different from S)

$d_S = (d - \Delta r)\frac{\bar{n}}{s}$ (4)

$\bar{n} = \max(|n_x|, |n_y|)$

$l(r) = l(c + rn)$ (unknown)

④ $d(\theta) = n^T[\pi(\underbrace{{}^c\hat{X}(\theta)}_{⑤}) - c]$

⑤ ${}^c\hat{X}(\theta) = \begin{bmatrix} {}^cR_m & {}^ct_m \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I + [\theta_r] & \theta_t \\ 0 & 1 \end{bmatrix} {}^m\tilde{X}$

③ $P(m_i|S) = \dfrac{\prod_{y \in S} \overbrace{P(y|m_i)}^{⑥}}{\prod_{y \in S} P(y|m_f) + \prod_{y \in S} P(y|m_b)}$

$P(m_i|S)$: S = a set of the closest s pixel values y; $y \in S$ ⑦

⑥ $P(m_i|y) = \dfrac{P(y|m_i)}{P(y|m_f) + P(y|m_b)}$, $i \in \{f, b\}$

⑦ $S = l_S(r_s)$ is a set-valued function that maps from the scaled line coordinate r_s to the segment S, which is a set of the closest s pixel values y.

$P(\theta|D)$ describes how well the current pose ${}^cX(\theta)$ estimate explains the segmentation of the image into a foreground region and background region.

Pose ${}^cX(\theta)$ ⟶ contour distance: $d(\theta)$ ⟶ $P(\theta|D)$

$$\ln(P(\theta|D)) \propto \sum_{i}^{i=nc} \ln[P(d_{s_i}(\theta)|w_{s_i}, l_s)]$$

$$= \sum_{i}^{i=nc} \sum_{r_s}^{r_s=w_s} \ln\left[\sum_{j \in \{f,b\}} P(r_s|d_s, m_j) P(m_j|l_s(r_s))\right]$$

8. To maximize the joint posterior probability $\ln(P(\theta|D))$, we estimate the variation vector δ and iteratively update the pose

$$\delta = (-H + \begin{bmatrix} \lambda_r I_3 & 0 \\ 0 & \lambda_t I_3 \end{bmatrix})^{-1} g$$

$$g^T = \left.\frac{\partial \ln(P(D|\theta))}{\partial \theta}\right|_{\theta=0}, \quad H = \left.\frac{\partial^2 \ln(P(D|\theta))}{\partial \theta^2}\right|_{\theta=0}$$

$$\therefore \; {}^cT_M = {}^cT_M \begin{bmatrix} \exp([\hat{\theta}_r]_\times) & \hat{\theta}_t \\ 0 & 1 \end{bmatrix}$$

$$d_s = (d - \Delta r)\frac{\bar{n}}{s}, \quad d(\theta) = n^T(\pi(cX(\theta)) - c)$$

① $$g^T = \sum_{i=1}^{n_c} \frac{\partial \ln(p(d_{si}|w_{si}, l_{si}))}{\partial d_{si}} \frac{\partial d_{si}}{\partial {}_cX_i} \frac{\partial {}_cX_i}{\partial \theta}\Big|_{\theta=0}$$

6×1 $\quad 1\times 3$ $\quad 3\times 6$

$$\because \; {}_c\bar{X}(\theta) = \begin{bmatrix} {}_cR_m & {}_ct_m \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I + [\theta_r] & \theta_t \\ 0 & 1 \end{bmatrix} {}_m\bar{X}$$

$$\theta_r \in \mathbb{R}^3, \quad \theta_t \in \mathbb{R}^3, \quad \theta^T = [\theta_r^T \;\; \theta_t^T]$$

$$\delta\xi = [\delta\rho, \; \delta\phi]^T$$

$$\frac{\partial(TP)}{\partial\delta\xi} = \lim_{\delta\xi\to 0} \frac{\exp(\xi^\wedge)\exp(\delta\xi^\wedge)P - \exp(\xi^\wedge)P}{\delta\xi}$$

$$\approx \lim_{\delta\xi\to 0} \frac{\exp(\xi^\wedge)(I + \delta\xi^\wedge)P - \exp(\xi^\wedge)P}{\delta\xi}$$

$$= \lim_{\delta\xi\to 0} \frac{\exp(\xi^\wedge)\delta\xi^\wedge P}{\delta\xi} \qquad \begin{bmatrix} \delta\phi^\wedge P + \delta\rho \\ 1 \end{bmatrix}$$

$$= \lim_{\delta\xi\to 0} \frac{\begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}\begin{bmatrix} \delta\phi^\wedge & \delta\rho \\ 0 & 1 \end{bmatrix}\begin{bmatrix} P \\ 1 \end{bmatrix}}{\delta\xi}$$

$$= \lim_{\delta\xi\to 0} \frac{\begin{bmatrix} R\delta\phi^\wedge P + R\delta\rho + t \\ 1 \end{bmatrix}}{\delta\xi} = A$$

$$= \begin{bmatrix} \frac{A}{\delta\rho} & \frac{A}{\delta\phi} \end{bmatrix} \qquad R\delta\phi^\wedge P = -RP^\wedge\delta\phi$$

$$= \begin{bmatrix} R & -RP^\wedge \\ 0 & 0 \end{bmatrix}$$

$$\frac{\partial {}_cX_i}{\partial\theta} = {}_cR_m\begin{bmatrix} -[{}_mX_i]_\times & I_3 \end{bmatrix} = [{}_mX_i \times {}_cR_m, \; {}_cR_m]$$

3×6 $\quad 3\times 3$ $\quad 3\times 6$

$[1\times 3]$ $\quad [X^\wedge(2)]$ $\quad \begin{bmatrix} -[{}_mX_i]_\times & I_3 \\ 3\times 3 & 3\times 3 \end{bmatrix}$ 3×6

$$d_s = (d - \Delta r)\frac{\bar{n}}{s}, \quad d(\theta) = n^T[\pi(c_c X(\theta)) - c]$$

$$x = \pi(X) = \begin{bmatrix} \frac{X}{Z} f_x + p_x \\ \frac{Y}{Z} f_y + p_y \end{bmatrix}$$

$$d_s = \left[\left(\frac{c_X}{c_Z} f_x + p_x - c_x\right) n_x + \left(\frac{c_Y}{c_Z} f_y + p_y - c_y\right) n_y\right] - \Delta r \Big] \frac{\bar{n}}{s}$$

$$\frac{1}{c_Z}(c_X f_x n_x + c_Y f_y n_y)$$

$$\therefore \underset{1\times 3}{\frac{\partial d_{si}}{\partial c_{Xi}}} = \frac{\bar{n}_i}{s}\left[\frac{f_x n_{xi}}{c_{Zi}} \quad \frac{f_y n_{yi}}{c_{Zi}} \quad -\frac{c_{Xi} f_x n_{xi} + c_{Yi} f_y n_{yi}}{c_{Zi}^2}\right]$$

(2) Suppose $f: R^n \to R$, $g: R \to R$, and $h(x) = g(f(x))$

$$\therefore \nabla^2 h(x) = \nabla[g'(f(x))\nabla f(x)]$$

$$= \nabla[g'(f(x))]\nabla f(x) + g'(f(x))\nabla[\nabla f(x)]$$

$$= g''(f(x))\underset{n\times 1}{\nabla f(x)}\underset{1\times n}{\nabla f(x)^T} + g'(f(x))\underset{n\times n}{\nabla^2 f(x)}$$

Let $g: R \to R$, $\ln(p(d_{si}|w_{si}, l_{si})) = g(d_{si})$

$f: R^n \to R$, $f(\underset{b\times 1}{\theta}) = d_{si}$

$$\therefore H = \sum_{i=1}^{nc} \frac{\partial^2 \ln(p(d_{si}|w_{si}, l_{si}))}{\partial d_{si}^2} \underbrace{\left(\frac{\partial d_{si}}{\partial \theta}\right)^T}_{b\times 1}\left(\frac{\partial d_{si}}{\partial \theta}\right)$$

$$+ \frac{\partial \ln(p(d_{si}|w_{si}, l_{si}))}{\partial d_{si}}\left(\frac{\partial^2 d_{si}}{\partial \theta^2}\right)$$

$$\approx \sum_{i=1}^{nc} \frac{\partial^2 \ln(p(d_{si}|w_{si}, l_{si}))}{\partial d_{si}^2}\left(\frac{\partial d_{si}}{\partial \theta}\right)^T\left(\frac{\partial d_{si}}{\partial \theta}\right)$$

① For global optimization : $\ln(p(d_{si}|w_{si}, l_{si})) \sim \mathcal{N}(d_{si}|\mu_i, \sigma_i^2)$

The required mean μ_i and standard deviation σ_i are thereby estimated from a set of discretized contour distances d_{si} and their corresponding probability values.

$$\therefore \frac{\partial(\ln p(d_{si}|w_{si}, l_{si}))}{\partial d_{si}} \approx -\frac{1}{\sigma_i^2}(d_{si} - \mu_i) \qquad \text{PDF: } f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

$$\frac{\partial^2(\ln p(d_{si}|w_{si}, l_{si}))}{\partial d_{si}^2} \approx -\frac{1}{\sigma_i^2}$$

If $a_n = \frac{1}{2}$, $\ln P(d|w,l) = \frac{1}{N\sqrt{2\pi}\sigma_n} \exp(-\frac{d^2}{2\sigma_n})$

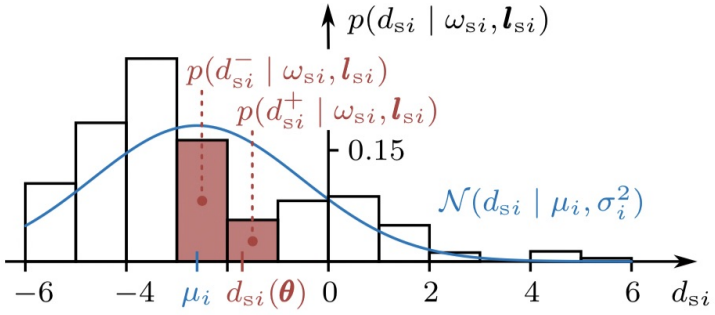


Fig. 9 Discrete posterior probability distribution with noisy measurements. For global optimization, the distribution is approximated by a normal distribution $\mathcal{N}(d_{si} \mid \mu_i, \sigma_i^2)$. The normal distribution and its mean μ_i are illustrated in blue. In the case of local optimization, only two discrete probability values that are closest to the current estimate of the contour distance $d_{si}(\boldsymbol{\theta})$ are considered. The two discrete probability values $p(d_{si}^- \mid \omega_{si}, \boldsymbol{l}_{si})$ and $p(d_{si}^+ \mid \omega_{si}, \boldsymbol{l}_{si})$, which are used to approximate the first-order derivative, are colored in red.

The approximated derivatives direct the optimization towards the mean μ_i using the variance σ_2^2 to consider uncertainty. In the real world, the mean does not exactly coincide with the maximum. Using the approximation has the advantage of fast convergence and that the opt avoids local minima resulting from invalid pixel-wise posteriors and image noise.

② Once the opt is closer to the maximum, the algorithm switches to local opt.

We use the probability d_{si}^- and d_{si}^+ that are closest to the current estimate $d_{si}(\theta)$ and :

$$\frac{\partial(\ln p(d_{si}|w_{si}, l_{si}))}{\partial d_{si}} \approx \frac{d_s}{\sigma_i^2} \ln\left(\frac{p(d_{si}^+|w_{si}, l_{si})}{p(d_{si}^-|w_{si}, l_{si}}\right)$$

$$\frac{\partial^2 \left(p(ds_i \mid ws_i, ls_i)\right)}{\partial ds_i^2} \approx -\frac{1}{\sigma_i^2}$$

If $E(\theta) = \sum_{i=1}^{nc} \frac{ds}{\sigma_i^2} \ln\left(p(ds_i(\theta) \mid ws_i, ls_i)\right)$

The weighting $\frac{1}{\sigma_i^2}$ improves robustness because correspondence lines with high uncertainty are considered less important.
The step size ds helps to balance the weight and specifies how far the opt proceeds, directly scaling the $\hat{\theta}$.

$$\therefore g^T = \sum_{i=1}^{nc} \frac{ds}{\sigma_i^2} \frac{\partial \ln\left(p(ds_i \mid ws_i, ls_i)\right)}{\partial ds_i} \frac{\partial ds_i}{\partial \theta}\Big|_{\theta=0}$$

$$H \approx \sum_{i=1}^{nc} \frac{ds}{\sigma_i^2} \frac{\partial^2 \ln\left(p(ds_i \mid ws_i, ls_i)\right)}{ds_i^2} \left(\frac{\partial ds_i}{\partial \theta}\right)^T \left(\frac{\partial ds_i}{\partial \theta}\right)\Big|_{\theta=0}$$

Central difference:

$$\because \frac{\partial \ln\left(p(ds_i \mid ws_i, ls_i)\right)}{\partial ds_i} \approx \frac{\ln\left(p(ds_i^+ \mid ws_i, ls_i)\right) - \ln\left(p(ds_i^- \mid ws_i, ls_i)\right)}{\Delta ds = 1}$$

$$= \ln\left(\frac{p(ds_i^+ \mid ws, ls_i)}{p(ds_i^- \mid ws_i, ls_i)}\right)$$

$$\therefore \frac{\partial E(\theta)}{\partial \theta} = \frac{1}{\sigma_i^2} \ln\left(\frac{p(ds_i^+ \mid ws, ls_i)}{p(ds_i^- \mid ws_i, ls_i)}\right) \left(\frac{\partial ds_i}{\partial \theta}\right)^T \left(\frac{\partial ds_i}{\partial \theta}\right)$$

Let $\frac{\partial^2 \ln\left(p(ds_i \mid ws_i, ls_i)\right)}{ds_i^2} \approx \frac{1}{ds}$

$$\therefore H \approx \sum_{i=1}^{nc} \left(\frac{1}{\sigma_i^2}\right) \left(\frac{\partial ds_i}{\partial \theta}\right)^T \left(\frac{\partial ds_i}{\partial \theta}\right)\Big|_{\theta=0}$$

↑ same as global opt

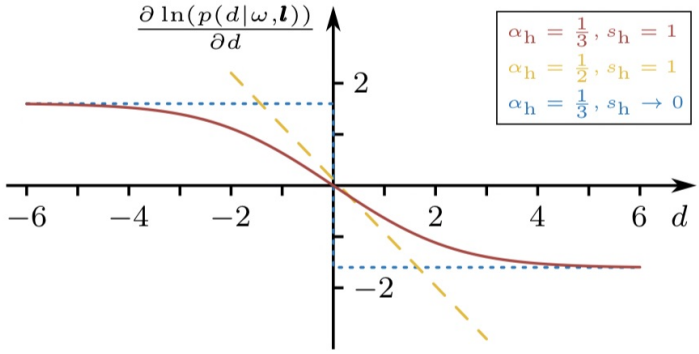


Fig. 5 First-order derivatives of the log-posterior with respect to the contour distance d for different slope and amplitude parameters s_h and α_h. The solid red line shows the derivative for $\alpha_h = \frac{1}{3}$ and $s_h = 1$, which yields a function with a smooth transition from an upper bound to a lower bound. The dashed yellow line shows the function for $\alpha_h = \frac{1}{2}$ and $s_h = 1$. This produces a linear first-order derivative. Finally, using $\alpha_h = \frac{1}{3}$ and $s_h \to 0$ results in a perfect step function illustrated by the dotted blue line.

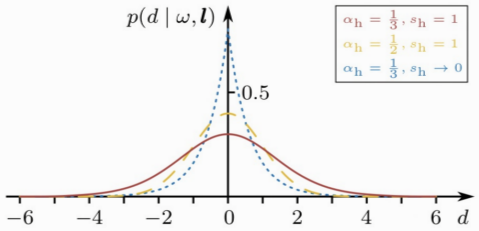


Fig. 6 Posterior probability distributions for different slope and amplitude parameters s_h and α_h. The solid red line shows the function for $\alpha_h = \frac{1}{3}$ and $s_h = 1$, which leads to a very flat distribution. Note that the function was computed numerically. Using $\alpha_h = \frac{1}{2}$ and $s_h = 1$ results in a Gaussian distribution shown by the dashed yellow line. The parameters $\alpha_h = \frac{1}{3}$ and $s_h \to 0$ yield a Laplace distribution for the posterior probability that is illustrated by a dotted blue line.

① For $\alpha_h = \frac{1}{2}$, $s_h = 1$ where $\frac{\partial \ln(p(d|\omega,\boldsymbol{l}))}{\partial d}$ is a linear first-order derivative. It leads to the estimation of the weighted mean over all correspondce lines.

$$g_T = \sum_{i=1}^{n_c} \frac{\partial \ln(p(ds_i|ws_i, ls_i))}{\partial ds_i} \frac{\partial ds_i}{\partial c x_i} \frac{\partial c x_i}{\partial \theta}\Big|_{\theta=0}$$

② For $\alpha_h = \frac{1}{3}$, $s_h \to 0$, $\frac{\partial \ln(p(d|\omega,\boldsymbol{l}))}{\partial d}$ is a binary derivatives guide the opt toward the weighted median.

9.

Algorithm 1 Tracking Step

1: Update camera image
2: **for** $i = 1, 2, \ldots, 7$ **do** $S=[5,2,1,1,1]$
3: **Optional:** Render occlusion mask
4: Find closest view of the sparse viewpoint model
5: Define correspondence lines in the image
6: Compute discrete distributions $p(d_{si} \mid \omega_{si}, \boldsymbol{l}_{si})$
7: **for** $j = 1, 2$ **do**
8: Calculate gradient $\boldsymbol{g}$ and Hessian $\boldsymbol{H}$
9: Estimate variation $\hat{\boldsymbol{\theta}}$ and update pose ${}_{\mathrm{C}}\boldsymbol{T}_{\mathrm{M}}$
10: **end for**
11: **end for**
12: Update color histograms $p(\boldsymbol{y} \mid m_{\mathrm{f}})$ and $p(\boldsymbol{y} \mid m_{\mathrm{b}})$

1. continuous distances from the sparse viewpoint model are used to reject correspondence lines with distances that are below 6 segments.

2.an offset of one pixel, the first 18 pixels are considered in both the positive and negative direction of the normal vector.

3. the posterior probability distribution p(d si | ω si ,l i) is evaluated at 12 discrete values d si ∈ { –5.5, –4.5, . . . , 5.5 } . distribution_num = 12

4. In the calculation, we use 8 precomputed values for the smoothed step functions h f and h b , corresponding to x ∈ { –3.5, –2.5, . . . , 3.5 } . function_num = 8

$x = r_{si} - d_{si}$

5.Also, a minimal offset Δri is chosen such that the line coordinates r i point to pixel centers while the scaled line coordinates r si ensure matching values for x = r si – d si . In our case, this means that r si∈ Z.

6. For the first iteration, the global optimization is used to quickly converge towards a rough pose estimate. In the second iteration, the local optimization is employed to refine this pose, using a step size of α s = 1.3. As regularization parameters, we use λr =5000 and λt = 500000.

10. Parameter Analysis

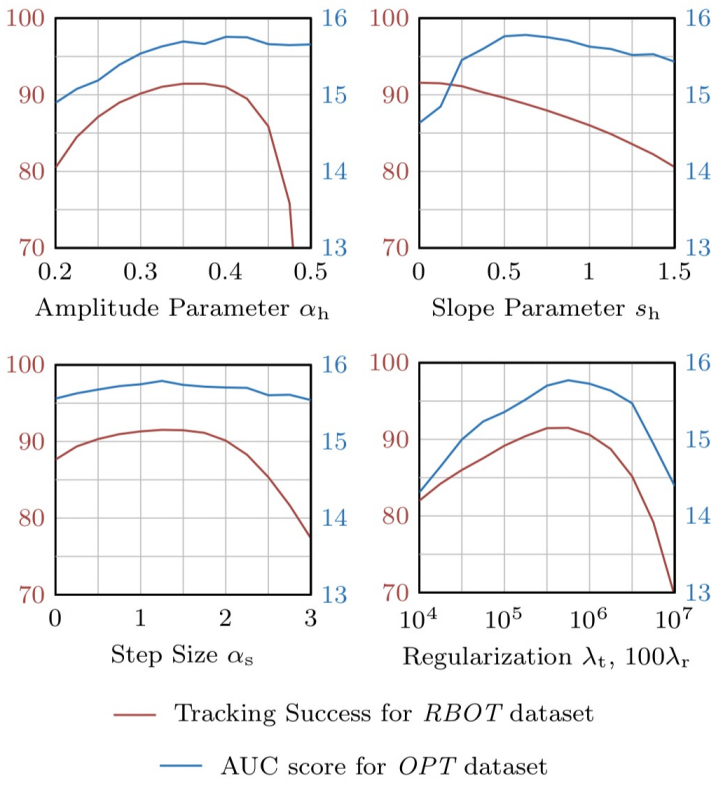


Fig. 13 Average tracking success for the *RBOT* dataset and average AUC score for the *OPT* dataset over different values of the amplitude parameter α_h, slope parameter s_h, step size α_s, and the rotational and translational regularization parameters λ_r and λ_t. For the evaluation of the regularization parameters, we set $\lambda_t = 100\lambda_r$.

α_h: a constant level of noise

s_h: local uncertainty

RBOT: $\alpha_h = 0.36$
$s_h \longrightarrow 0$

OPT: $\alpha_h = 0.42$
$s_h = 0.5$
$\lambda_r = 500000$ for soda

$\alpha_s = 1.3$

$\lambda_t = 100\lambda_r$,
while $\lambda_r = \lambda_t$ for soda

If λ_r, λ_t too small, the opt is unstabe for directions in which no or very little information is available.
If $\lambda_r \lambda_t$ too large. the opt is slowed down and the final pose might not be reached.

Use regularization parameters that are in the same order of magnitude as the maximum rotational and translational diagonal elements of the Hessian matrix.

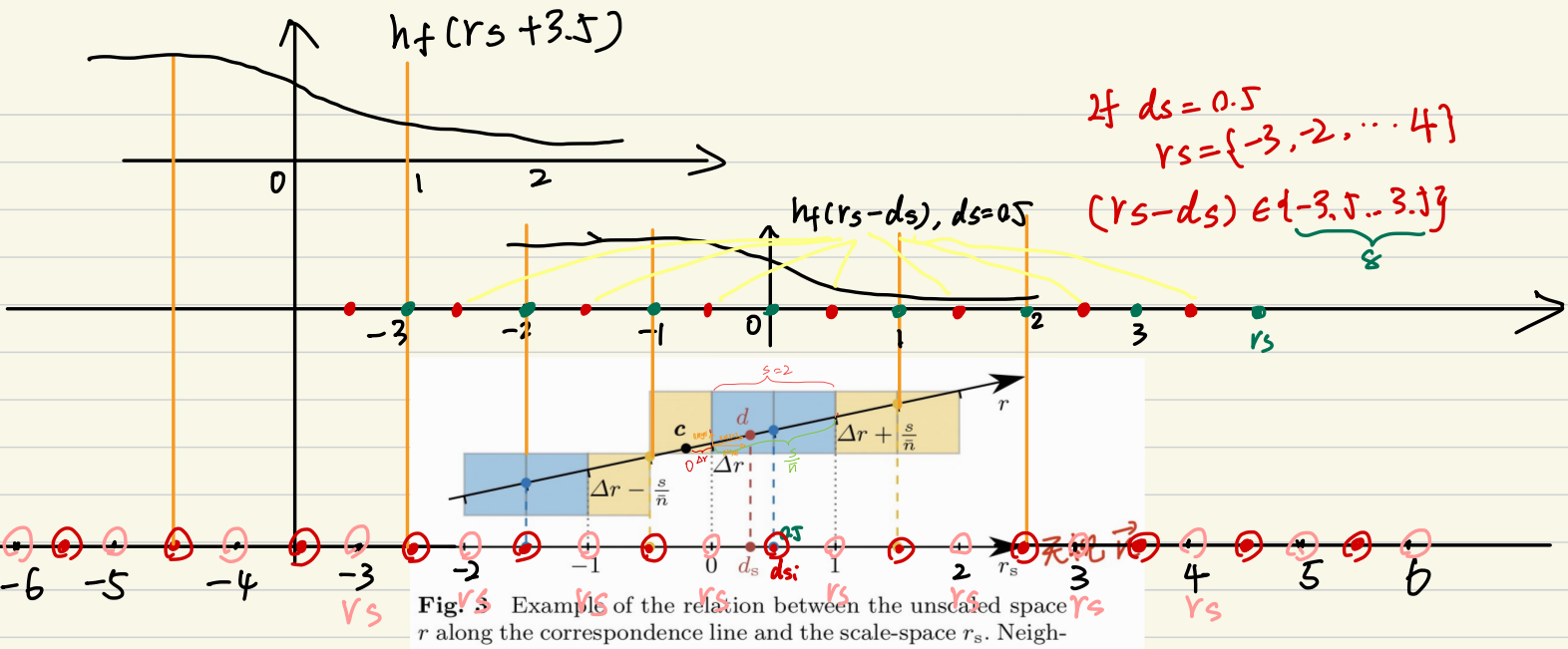


Fig. 3 Example of the relation between the unscaled space r along the correspondence line and the scale-space r_s. Neighboring pixels that are combined into segments are visualized by the same color in blue or yellow. Blue and yellow dots indicate the center of each segment and the corresponding discretized value in the scale-space. An example of the contour distance is illustrated in red. The offset Δr is chosen in a way that ensures that discretized values in the scale-space are the same for all correspondence lines. In this example, Δr points to the closest edge between pixels.

② $P(r_s \mid d_s, m_i) = h_i(r_s - d_s) = \begin{cases} \frac{1}{2} - a_h \tanh[(r_s - d_s)/2s_h] \\ \frac{1}{2} + a_h \tanh[(r_s - d_s)/2s_h] \end{cases}$ (③)

③ $r_s = (r - \Delta r)\frac{\bar{n}}{s}$, $d_s = (d - \Delta r)\frac{\bar{n}}{s}$ (④)

$s \in \mathbb{N}^+$: the number of pixels combined into a segment. (different from s)

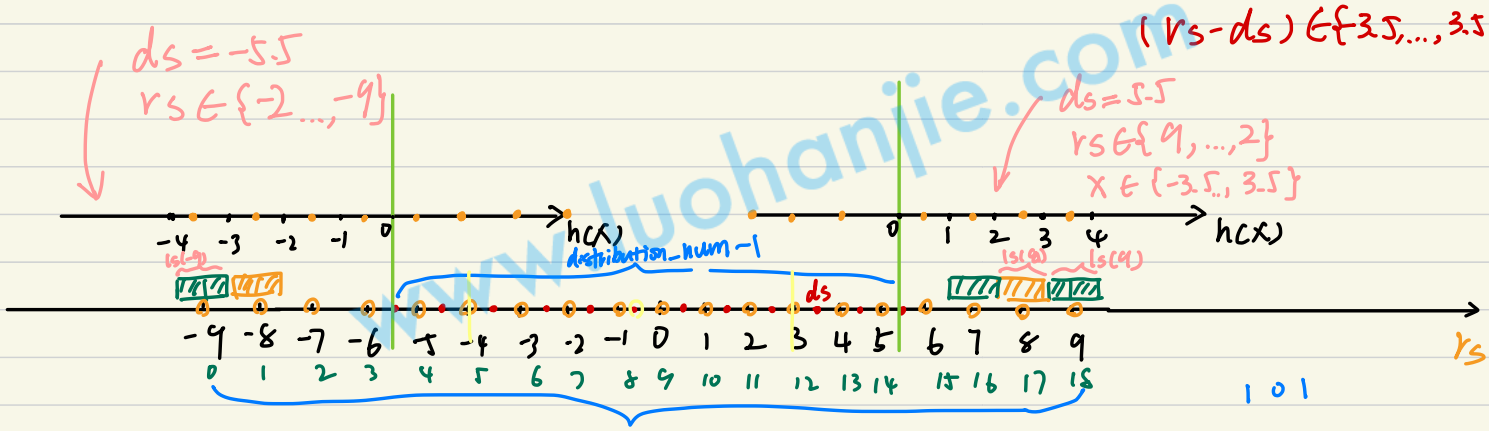
$\bar{n} = \max(|n_x|, |n_y|)$

$r_s - d_s$

$$P(d_s \mid w_s, l_s) \propto \prod_{r_s \in w_s} \sum_{i \in \{f,b\}} P(r_s \mid d_s, m_i)\, P(m_i \mid l_s(r_s))$$

$d_s \in \{-5.5, \ldots, 5.5\}$, $l_s = s$

$(r_s - d_s) \in \{-3.5, \ldots, 3.5\}$



line_length_segment = (function_num − 1) + (distribution_num − 1)

19 = $x \in \{-3.5, 3.5\}$ (8) + $s_i \in \{-5.5, 5.5\}$ (12)

line_length_pixel = line_length_segment × scale.

$\Delta r_x = \text{round}(C_x - \text{line\_length\_pixel\_half}) - (C_x - \text{line\_length\_pixel\_half})$

$\Delta r = \Delta r_x / n_x$

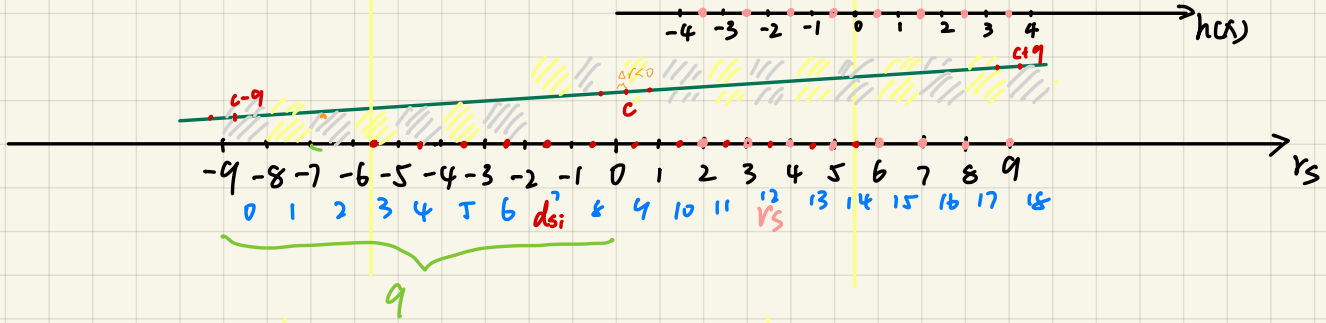
$d_s = 55$

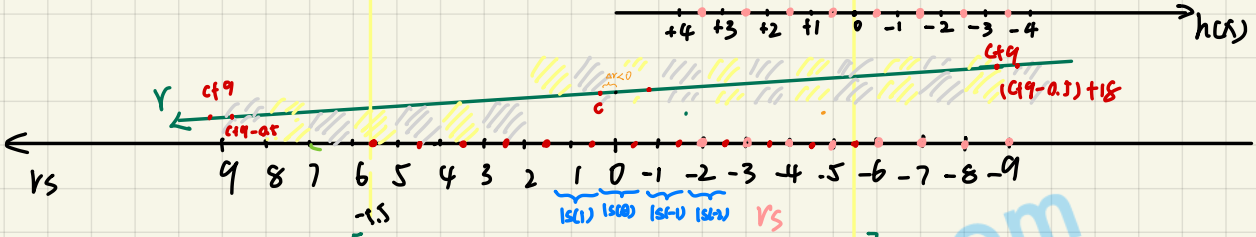
$r_s = \{2, \dots 9\}$

$r_s - d_s \in \{-3.5, \dots 3.5\}$

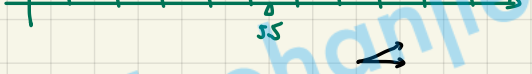
$$P(d_s \mid w_s, l_s) \propto \prod_{r_s \in w_s} \sum_{i \in \{f,b\}} h_i(r_s - d_s)\, p(m_i \mid l_s(r_s)) \overset{s}{=}$$

$r_s = (r - \Delta r)\frac{n}{s}$

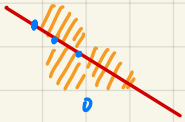


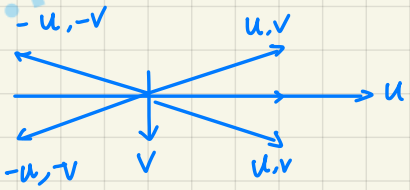


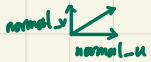
$P(d_s \mid w_s, l_s) = [11\ 10\ 9\ 8\ 7\ 6\ 5\ 4\ 3\ 2\ 1\ 0]$



$r_s = (r - \Delta r)\frac{n}{s}$





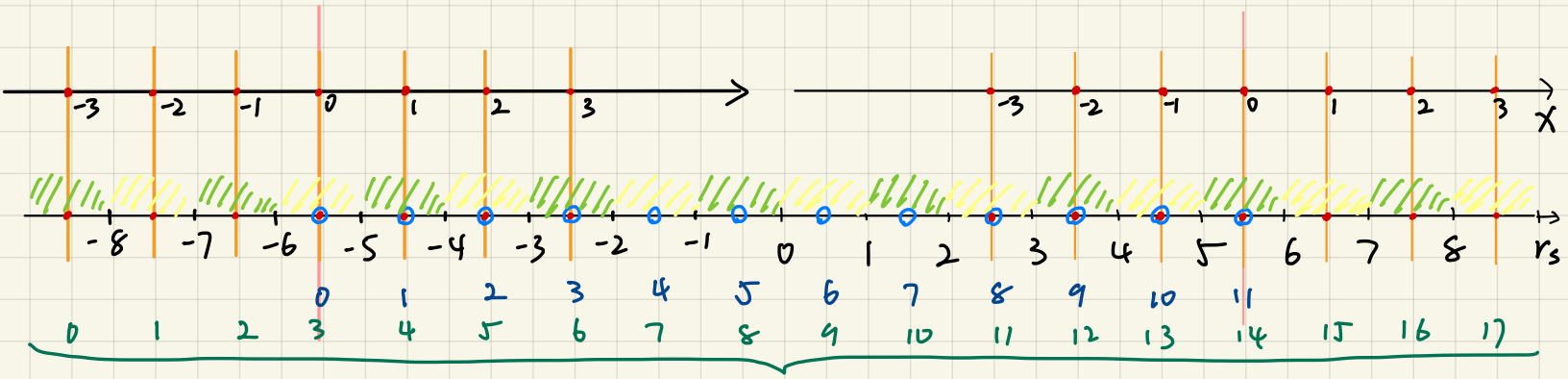


My version

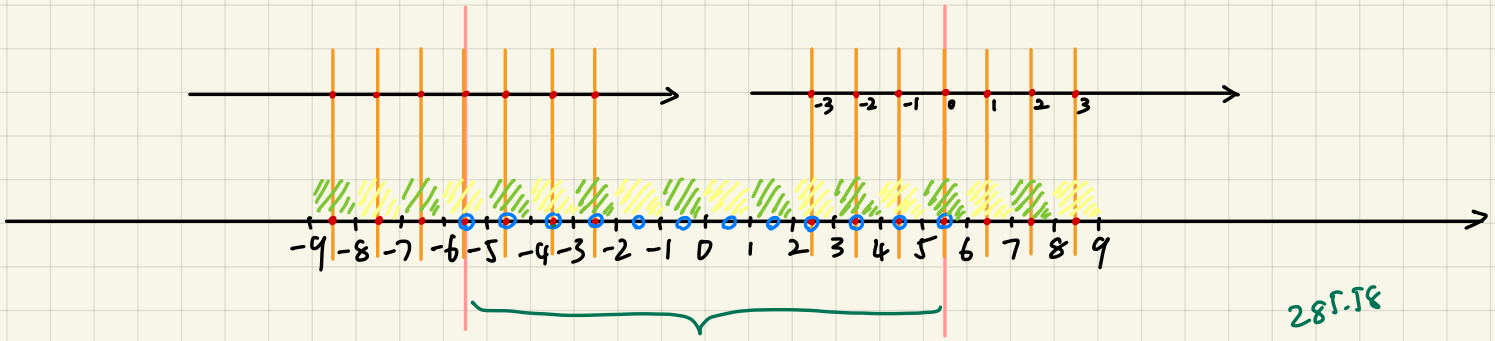
$d_s = 5.5$

$r_s = \{2.5, 3.5, \ldots, 7.5, 8.5\}$

$x = r_s - d_s = \{-3, -2, \ldots, 3\}$



$$\text{line\_length\_pixel} = (\text{line\_length\_segment} + 1) \times \text{scale}$$
$$= 2 \times 18 = 36$$

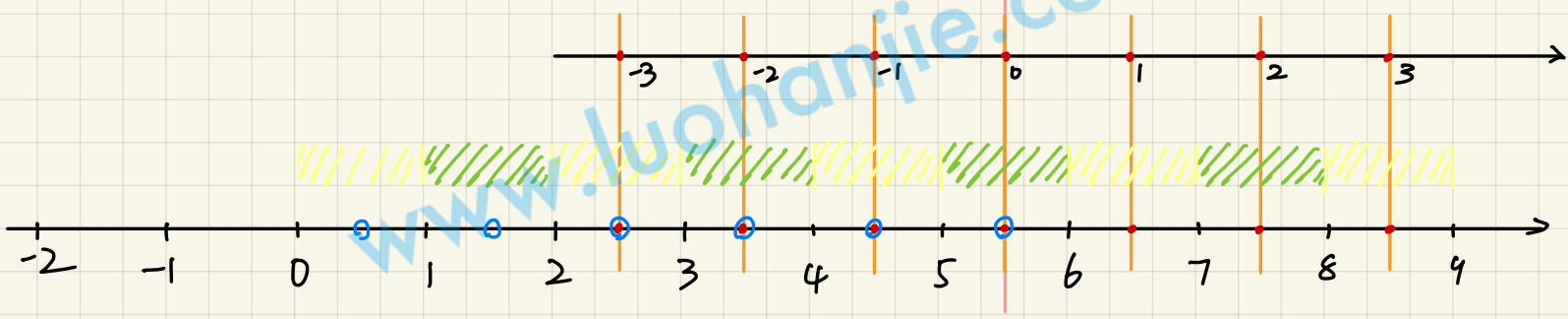


$\text{distribution\_num} - 1 = 11$ (12)

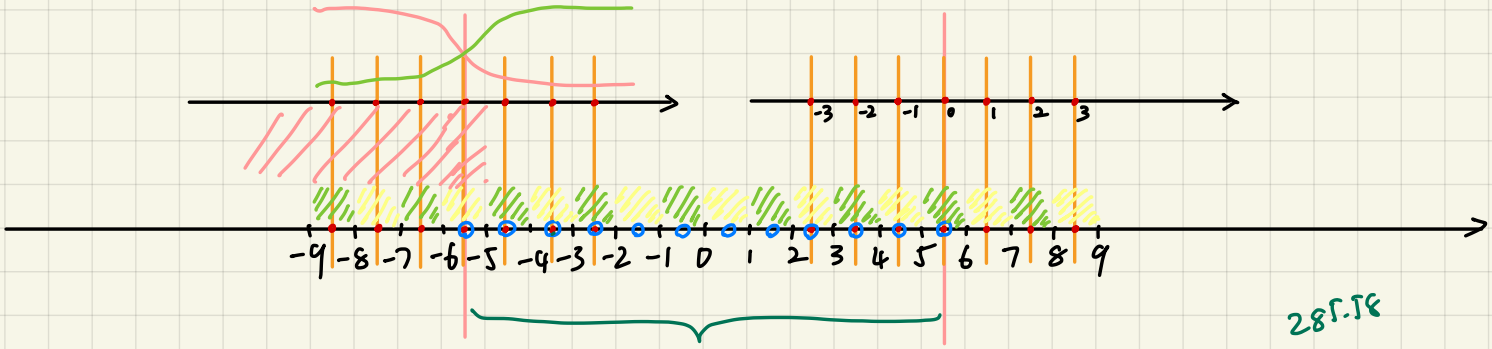
285.18

21

$$\text{line\_length\_segment} = 17 = \underset{7}{\text{function\_num}} + \underset{12}{\text{distribution\_num}} - 2$$



$$\frac{\text{distribution\_num}}{2} \times \text{scale}$$



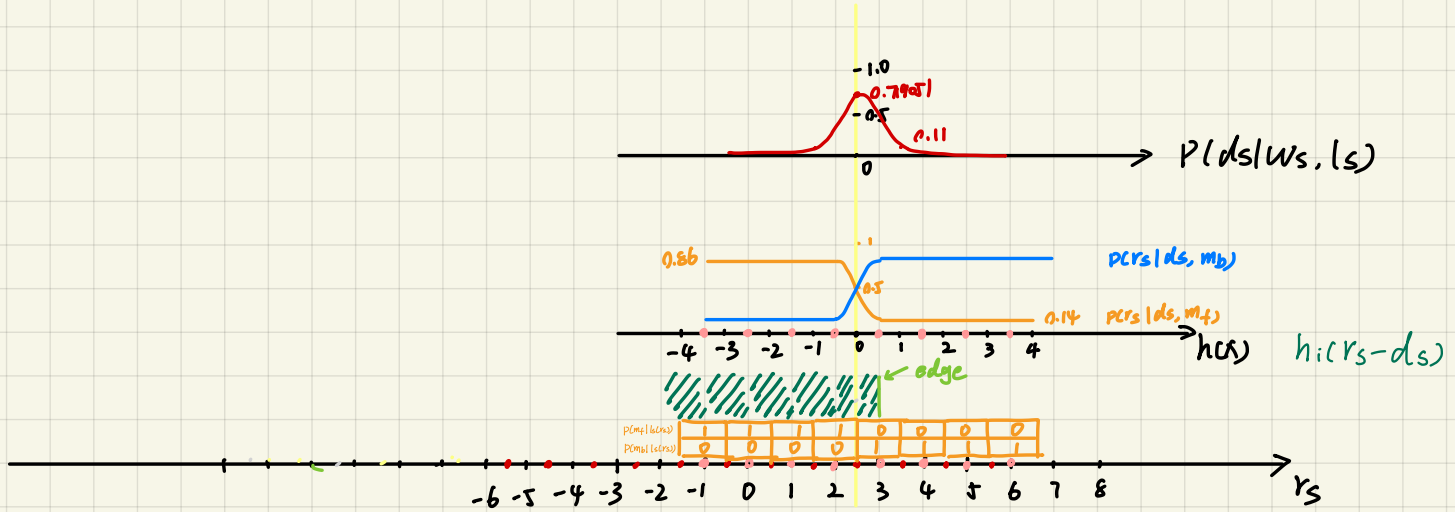
285.16

21

distribution_num − 1 = 11
12

line_length_segment = 17 = function_num + distribution_num − 2
7 12

☆ my version !

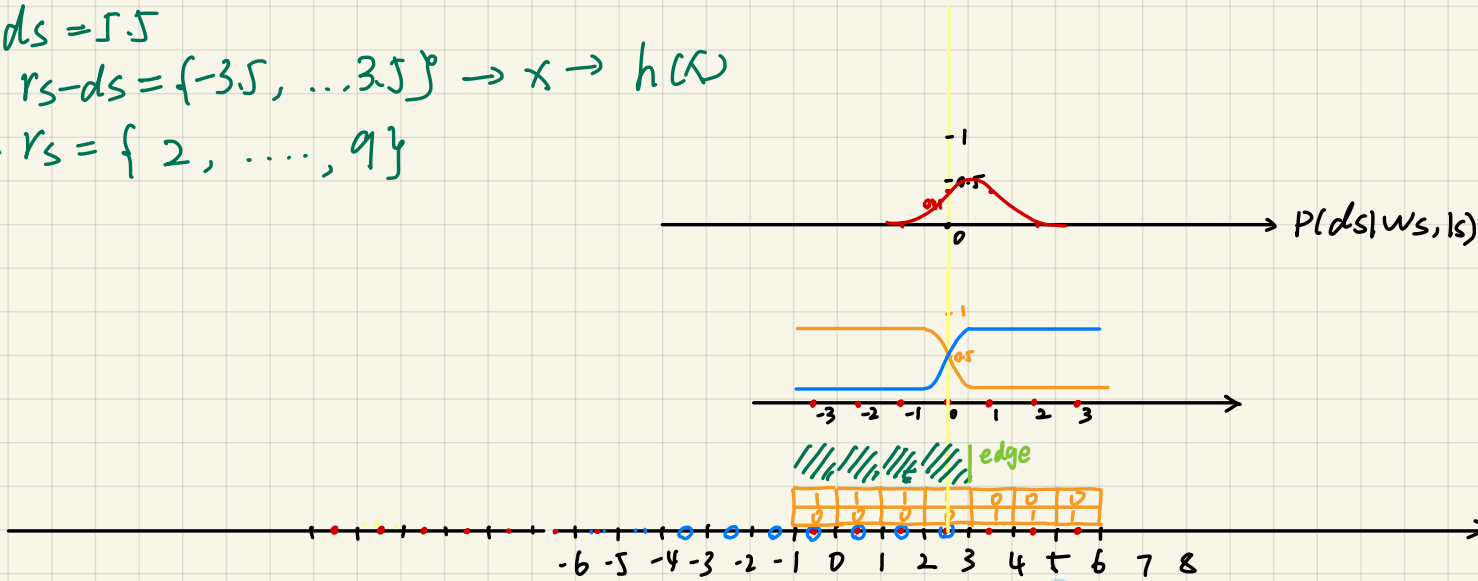


$ds = \{-5.5, -4.5, -3.5, -2.5, -1.5, -0.5, 0.5, 1.5, 2.5, 3.5, 4.5, 5.5\}$

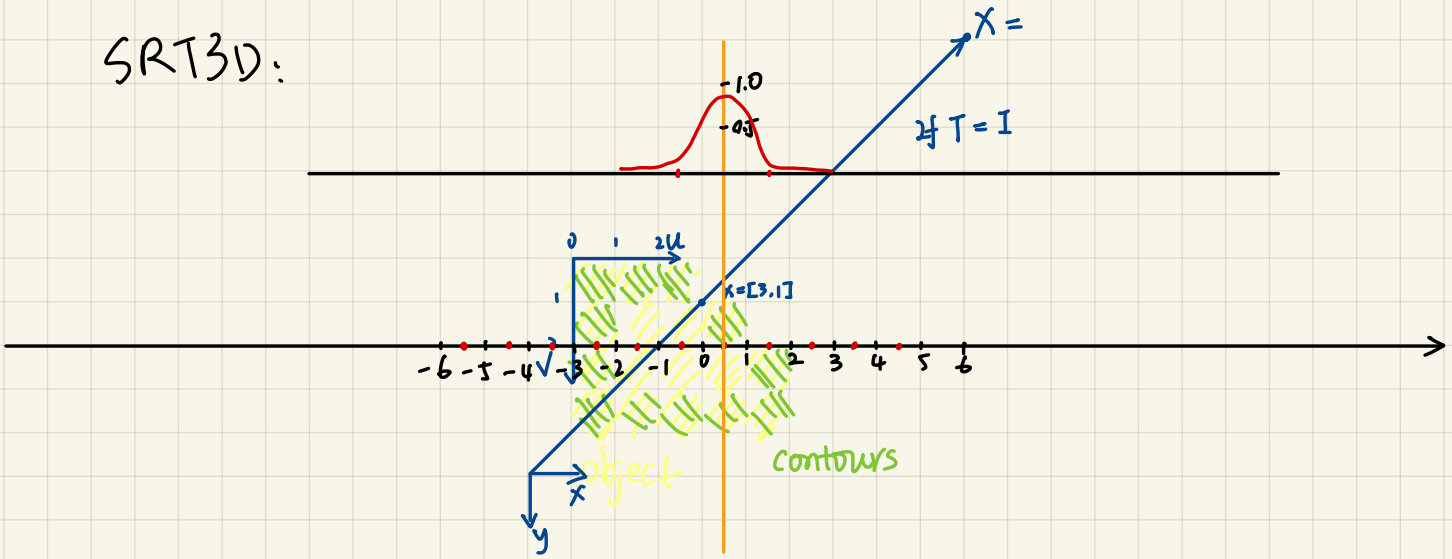
If $ds = 5.5$

$\because rs - ds = \{-3.5, \ldots 3.5\} \rightarrow x \rightarrow h(x)$

$\therefore rs = \{2, \ldots, 9\}$



SRT3D:

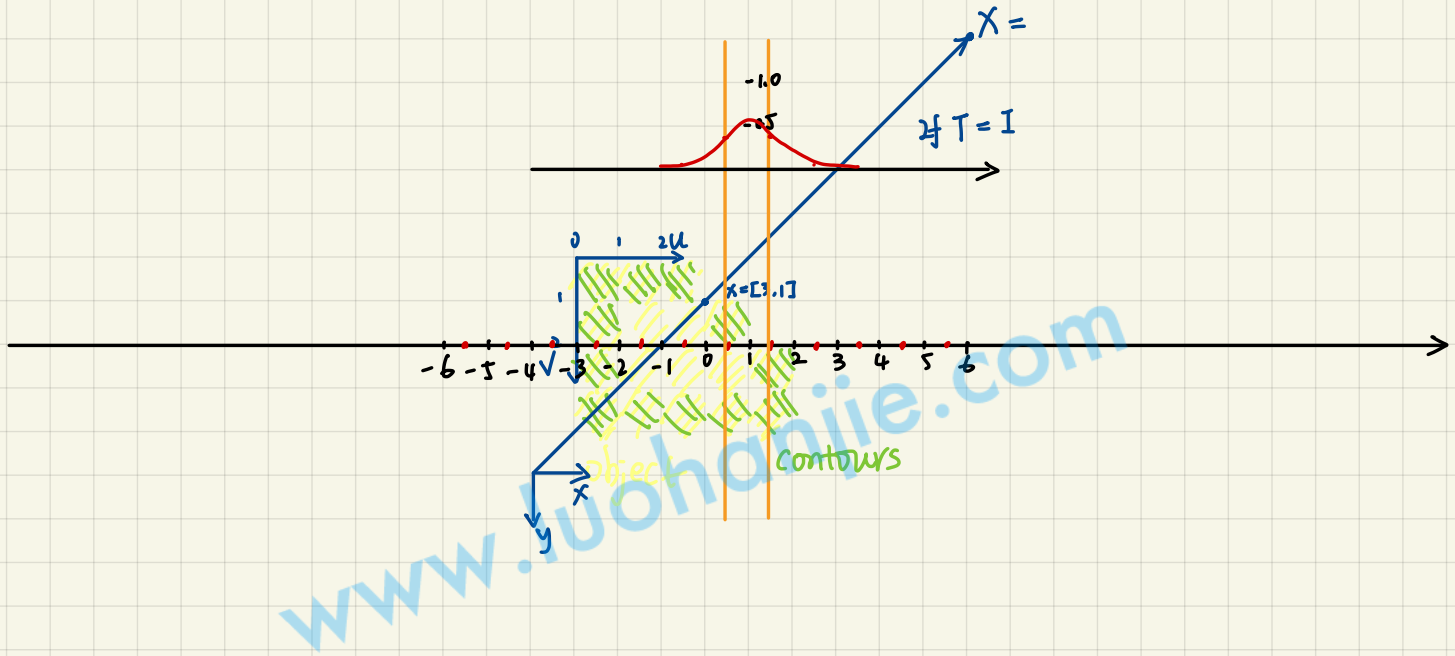


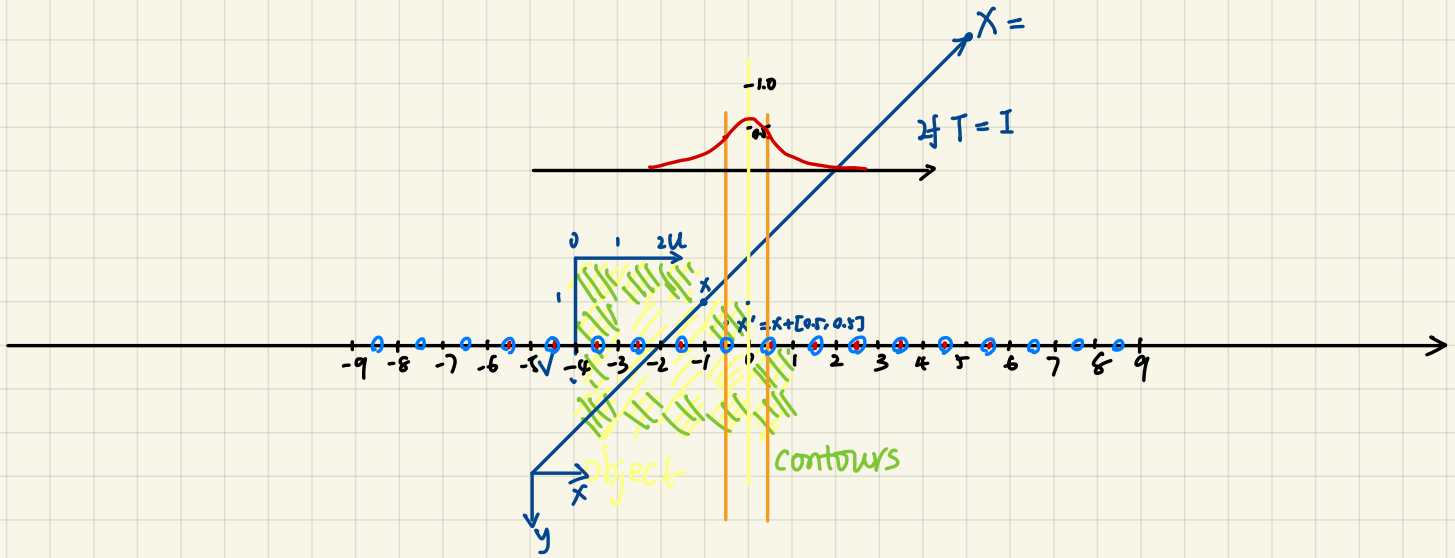
$$d(\theta) = n^T[\pi(c \times (\theta)) - c)$$

$$r_s = (r - \Delta r)\frac{\bar{n}}{s}$$

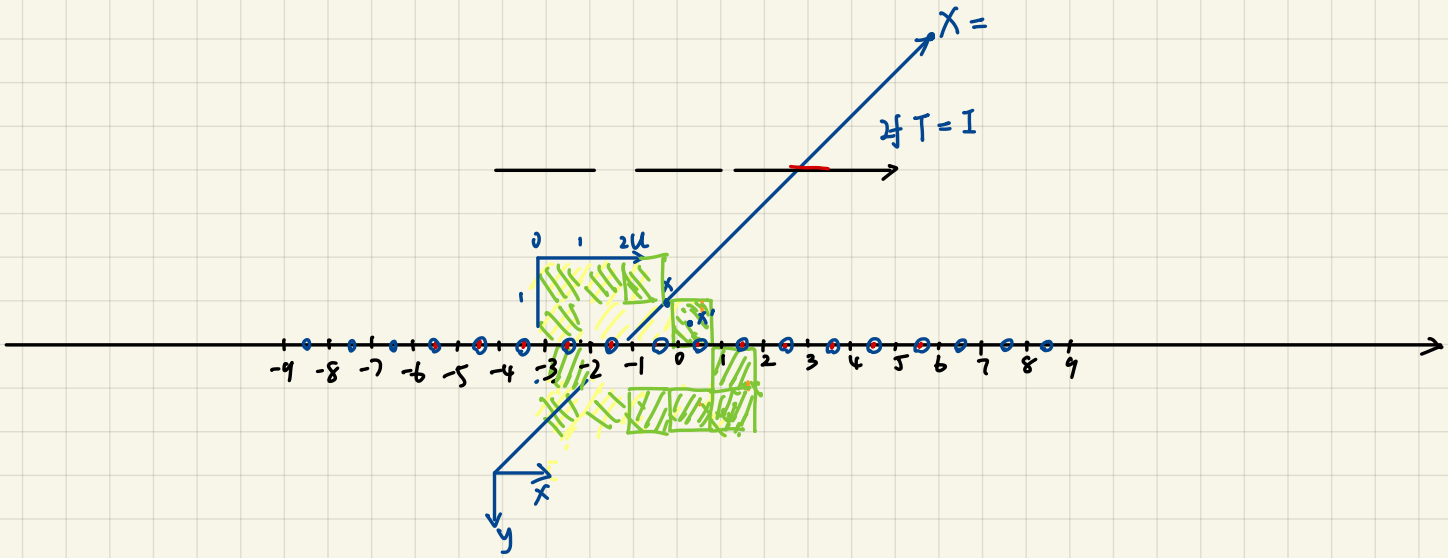
$$d_s = (d - \Delta r)\frac{\bar{n}}{s}$$

My:

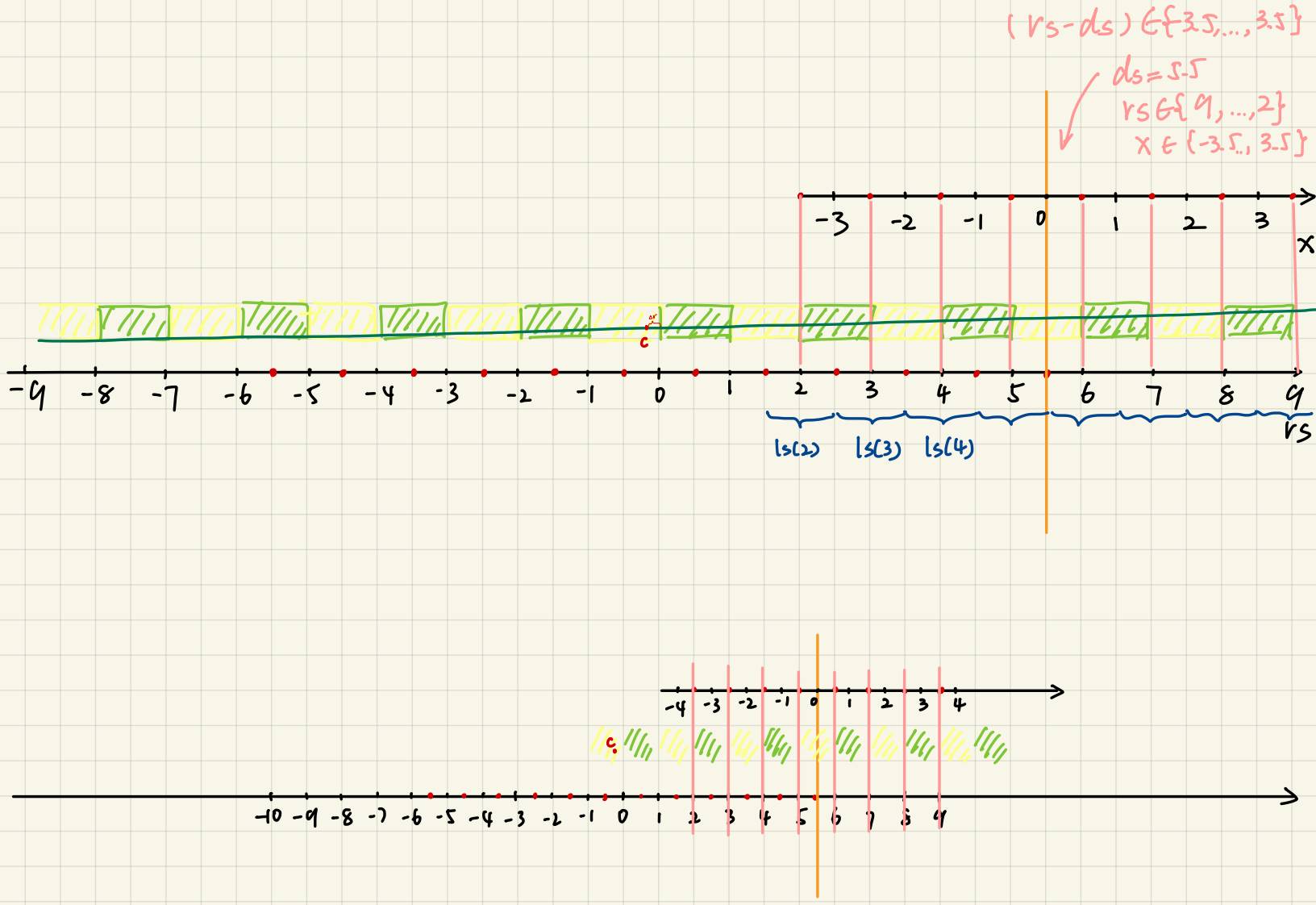




$$d(\theta) = n^T(\pi_L(c\,X(\theta)) - c) \qquad d_S = (d - \Delta r)\frac{\bar{n}}{S}$$



Draft



• , $ds \in \{-5.5 \ldots 5.5\}$, $P(ds \mid ws, ls)$

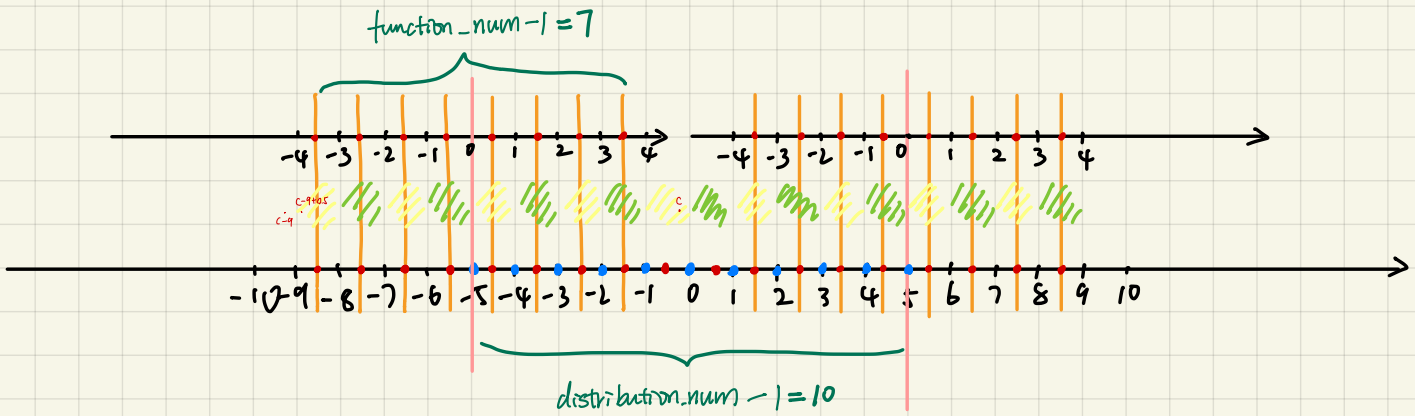
$$P(rs \mid ds, m_i) = h_i(rs - ds) = \begin{cases} \frac{1}{2} - a_h \tanh[(rs-ds)/2s_h] \\ \frac{1}{2} + a_h \tanh[(rs-ds)/2s_h] \end{cases}$$

$$P(ds \mid ws, ls) \propto \prod_{rs \in ws} \sum_{i \in \{f,b\}} P(rs \mid ds, m_i) P(m_i \mid ls(rs))$$

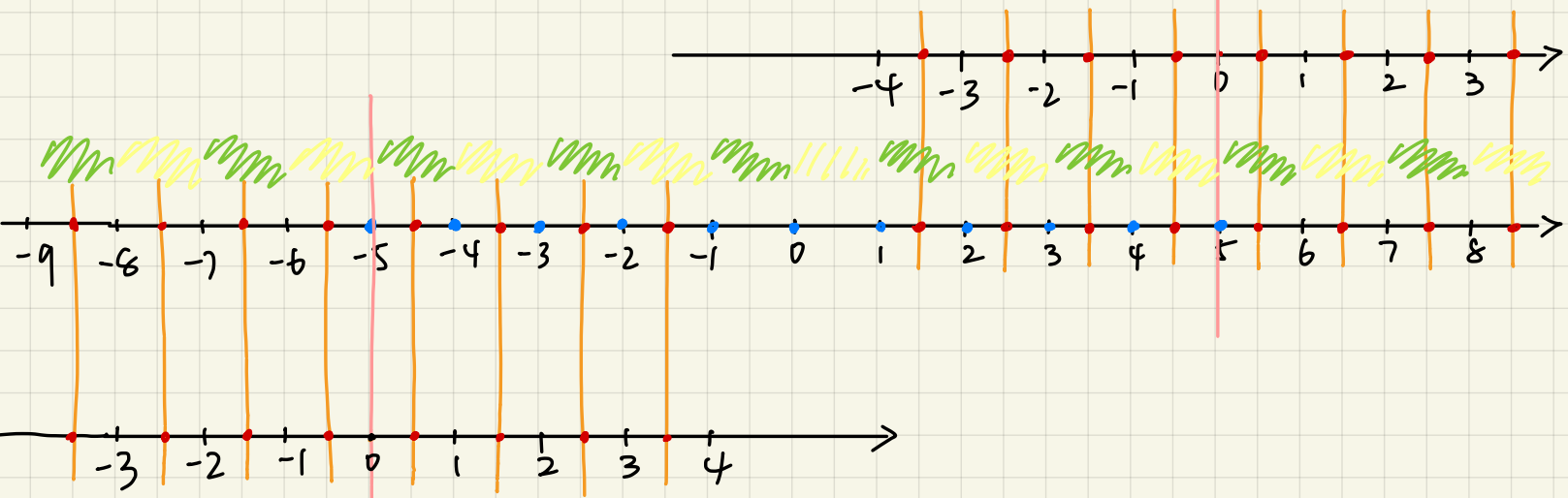
($ds \in \{-5.5, \ldots, 5.5\}$; $= ls(rs)$)

$$rs = (r - \Delta r)\frac{\bar{n}}{s} \qquad ds = (\underbrace{d}_{\sim} - \Delta r)\frac{\bar{n}}{s}$$

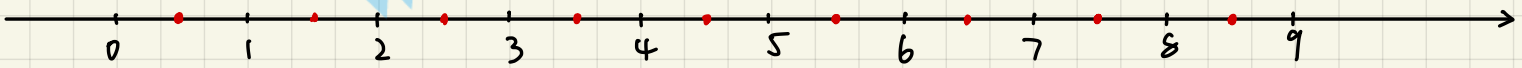
Draft

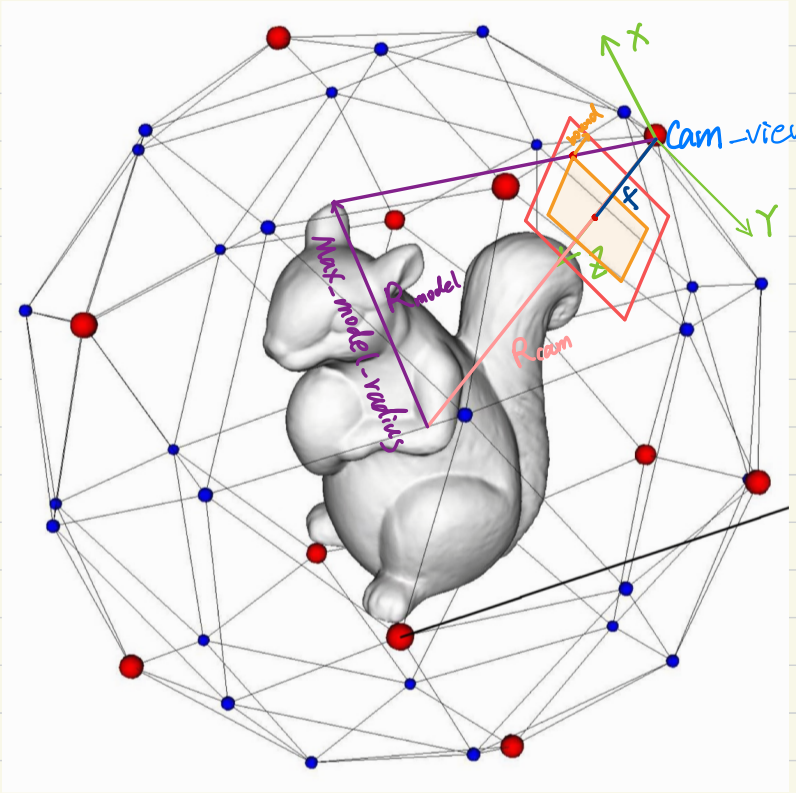


line_length_segment =17= function_num + distribution_num -2



line_length_pixel_ = (line_length_segment+1)×scale
= 2×18 =36





$$\because f_x \frac{X}{Z} + c_x = u$$

One Point in the max model sphere

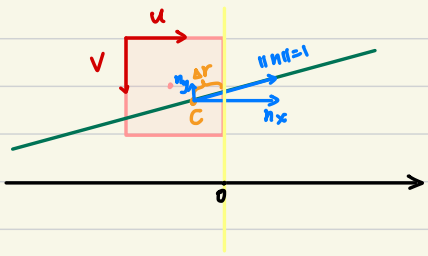
$X = [R_{model}, 0, R_{cam}]$

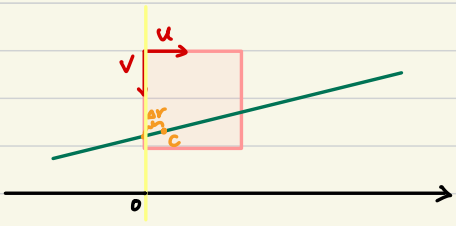
$$f \frac{R_{model}}{R_{cam}} + c_x = image\_size - bound$$

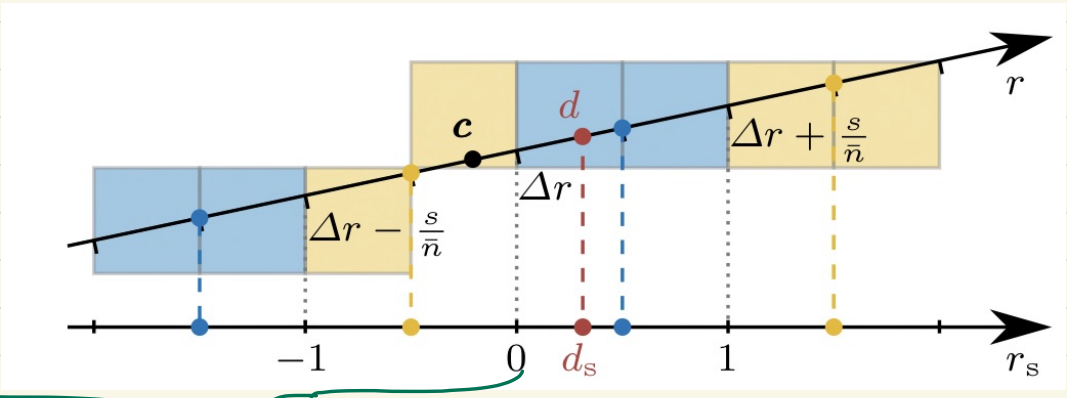
make $c_x = \frac{image\_size}{2}$

$$\therefore f \frac{R_{model}}{R_{cam}} = \frac{image\_size}{2} - bound$$

$$f = \left(\frac{image\_size}{2} - bound\right)\frac{R_{cam}}{R_{model}}$$







$$\frac{(\text{line\_length} - 1)}{2}$$

if Scale = 2.
line_length =

$$P(m_i|y_1,y_2,y_3) = \frac{P(y_1|m_i)}{P(y_1|m_f)+P(y_1|m_b)} \cdot \frac{P(y_2|m_i)}{P(y_2|m_f)+P(y_2|m_b)} \cdot \frac{P(y_3|m_i)}{P(y_3|m_f)+P(y_3|m_b)}$$

$$= P(m_i|y_1)\ P(m_i|y_2)\ P(m_i|y_3).$$

$\because P(m_b)=P(m_f)$

$$P(m_i|y_1,y_2,y_3) = \frac{P(y_1,y_2y_3|m_i)P(m_i)}{P(y_1,y_2,y_3)} = \frac{P(y_1|m_i)P(y_2|m_i)P(y_3|m_i)P(m_i)}{\sum_{j\in\{f,b\}} P(y_1y_2y_3|m_j)P(m_j)}$$

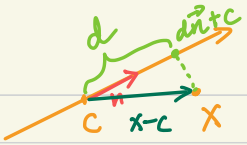
$$= \frac{P(y_1|m_i)P(y_2|m_i)P(y_3|m_i)}{P(y_1y_2y_3|m_f)+P(y_1y_2y_3|m_b)}$$

$$= \frac{P(y_1|m_i)P(y_2|m_i)P(y_3|m_i)}{P(y_1y_2y_3|m_f)+P(y_1y_2y_3|m_b)}$$

$$= \frac{P(y_1|m_i)P(y_2|m_i)P(y_3|m_i)}{P(y_1|m_f)P(y_2|m_f)P(y_3|m_f)+P(y_1|m_b)P(y_2|m_b)P(y_3|m_b)}$$

$$P(m_f|S) = \frac{1}{1+\prod_{y\in S}\frac{P(y|m_b)}{P(y|m_f)}}$$

① $d = n^T(\underbrace{X}_{an} - C)$



② $d_s = (d - \Delta r)\frac{\bar{n}}{s}$

we know d_s, and want to know the pixel $dn + c$

$d = d_s \frac{s}{\bar{n}} + \Delta r$