Supplementary Material - Robustness Certification of Visual Perception Models via Camera Motion Smoothing

Hanjiang Hu[1] **Zuxin Liu**[1] **Linyi Li**[2] **Jiacheng Zhu**[1] **Ding Zhao**[1]
[1]Carnegie Mellon University [2]University of Illinois at Urbana-Champaign
{hanjianghu,dingzhao}@cmu.edu, {zuxinl,jzhu4}@andrew.cmu.edu, linyi2@illinois.edu

A Method Details and Proofs

We first present the preliminary definitions in Section A.1and provide details for Definition 1 regarding translations and rotations on all 6 degrees of freedom in Section A.2. Then we show the proof for main text Lemma 1 and Theorem 1.

A.1 Preliminary Definitions

Definition 1 ((**restated**) Position projective function)**.** *For any 3D point $P = (X, Y, Z) \in \mathbb{P} \subset \mathbb{R}^3$ under the camera coordinate frame with the camera intrinsic matrix K, based on the camera motion $\alpha = (\boldsymbol{\theta}, t) \in \mathcal{Z} \subset \mathbb{R}^6$ with rotation matrix $R = \exp(\boldsymbol{\theta}^\wedge) \in SO(3)$ and translation vector $t \in \mathbb{R}^3$, we define the position projective function $\rho : \mathbb{P} \times \mathcal{Z} \to \mathbb{R}^2$ and the depth function $D : \mathbb{P} \times \mathcal{Z} \to \mathbb{R}$ for point P as*

$$[\rho(P, \alpha), 1]^\top = \frac{1}{D(P, \alpha)} K R^{-1}(P - t), \quad D(P, \alpha) = [0, 0, 1] R^{-1}(P - t) \tag{1}$$

Definition 2 ((**restated**) Channel-wise projective transformation)**.** *Given the position projection function $\rho : \mathbb{P} \times \mathcal{Z} \to \mathbb{R}^2$ and the depth function $D : \mathbb{P} \times \mathcal{Z} \to \mathbb{R}$ over dense 3D point cloud $\mathbb{P}$, define the 3D-2D global channel-wise projective transformation from C-channel colored point cloud $\mathbb{V} = (\mathbb{R}^C, \mathbb{P}) \subset \mathbb{R}^{C+3}$ to $H \times W$ image gird $\mathcal{X} \subset \mathbb{R}^{C \times H \times W}$ as $O : \mathbb{V} \times \mathcal{Z} \to \mathcal{X}$ parameterized by camera motion $\alpha \in \mathcal{Z}$ using Floor function $\lfloor \cdot \rfloor$,*

$$x_{c,r,s} = O(V, \alpha)_{c,r,s} = V_{c,P^*_\alpha}, \text{where } P^*_\alpha = \underset{\{P \in \mathbb{P} | \lfloor \rho(P,\alpha) \rfloor = (r,s)\}}{\operatorname{argmin}} D(P, \alpha) \tag{2}$$

Specifically, if $x = O(V, 0)$, we define the relative projective transformation $\phi : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ as,

$$\phi(x, \alpha) = O(V, \alpha). \tag{3}$$

Definition 3 ((**restated**) Camera motion ε-smoothed classifier)**.** *Let $\phi : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ be a relative projective transformation given the projected image x at the origin of camera motion in the motion space $\mathcal{Z}$, and let $\varepsilon \sim \mathcal{P}_\varepsilon$ be a random camera motion taking values in $\mathcal{Z}$. Let $h : \mathcal{X} \to \mathcal{Y}$ be a base classifier $h(x) = \operatorname{argmax}_{y \in \mathcal{Y}} p(y \mid x)$, the expectation of projected image predictions $\phi(x, \varepsilon)$ over camera motion distribution $\mathcal{P}_\varepsilon$ is $q(y \mid x; \varepsilon) := \mathbb{E}_{\varepsilon \sim \mathcal{P}_\varepsilon} p(y \mid \phi(x, \varepsilon))$. We define the ε-smoothed classifier $g : \mathcal{X} \to \mathcal{Y}$ as*

$$g(x; \varepsilon) := \underset{y \in \mathcal{Y}}{\operatorname{argmax}}\, q(y \mid x; \varepsilon) = \underset{y \in \mathcal{Y}}{\operatorname{argmax}}\, \mathbb{E}_{\varepsilon \sim \mathcal{P}_\varepsilon} p(y \mid \phi(x, \varepsilon)). \tag{4}$$

A.2 Projective Function Details

Following (1), given 3D point $P_0 = (X_0, Y_0, Z_0)^T$ under the camera coordinate frame, with axis-angle or rotation vector $\boldsymbol{\theta} = (\theta n_1, \theta n_2, \theta n_3)^T \in \mathbb{R}^3, \|\boldsymbol{\theta}\|_2 = \theta \in \mathbb{R}$ and translation

6th Conference on Robot Learning (CoRL 2022), Auckland, New Zealand.

$t = (t_x, t_y, t_z)^T \in \mathbb{R}^3$, the camera intrinsic matrix K of the camera is shown below.

$$K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}$$

$$R^{-1} = \begin{pmatrix} \cos\theta + (1-\cos\theta)n_1^2 & (1-\cos\theta)n_1n_2 + n_3\sin\theta & (1-\cos\theta)n_1n_3 - n_2\sin\theta \\ (1-\cos\theta)n_1n_2 - n_3\sin\theta & \cos\theta + (1-\cos\theta)n_2^2 & (1-\cos\theta)n_2n_3 + n_1\sin\theta \\ (1-\cos\theta)n_1n_3 + n_2\sin\theta & (1-\cos\theta)n_2n_3 - n_1\sin\theta & \cos\theta + (1-\cos\theta)n_3^2 \end{pmatrix}$$

First we find the depth of P_0 given camera pose $\alpha = \{\theta, t\}$,

$$\begin{aligned} D(P_0, \alpha) = &[(1-\cos\theta)n_1n_3 + n_2\sin\theta](X_0 - t_x) \\ &+ [(1-\cos\theta)n_2n_3 - n_1\sin\theta](Y_0 - t_y) \\ &+ [\cos\theta + (1-\cos\theta)n_3^2](Z_0 - t_z) \end{aligned}$$

Then we find the pixel coordinates on the image.

$$\begin{aligned} \rho_1(P_0, \alpha) = \frac{1}{D(P_0,\alpha)}\{&[f_x[\cos\theta + (1-\cos\theta)n_1^2] + c_x[(1-\cos\theta)n_1n_3 + n_2\sin\theta]](X_0 - t_x) \\ &+ [f_x[(1-\cos\theta)n_1n_2 + n_3\sin\theta] + c_x[(1-\cos\theta)n_2n_3 - n_1\sin\theta]](Y_0 - t_y) \\ &+ [f_x[(1-\cos\theta)n_1n_3 - n_2\sin\theta] + c_x[\cos\theta + (1-\cos\theta)n_3^2]](Z_0 - t_z)\} \\ \rho_2(P_0, \alpha) = \frac{1}{D(P_0,\alpha)}\{&[f_y[\cos\theta + (1-\cos\theta)n_2^2] + c_y[(1-\cos\theta)n_2n_3 - n_1\sin\theta]](Y_0 - t_y) \\ &+ [f_y[(1-\cos\theta)n_1n_2 - n_3\sin\theta] + c_y[(1-\cos\theta)n_1n_3 + n_2\sin\theta]](X_0 - t_x) \\ &+ [f_y[(1-\cos\theta)n_2n_3 + n_1\sin\theta] + c_y[\cos\theta + (1-\cos\theta)n_3^2]](Z_0 - t_z)\} \end{aligned}$$

Specifically, the camera motion on each axis is shown as follows.

A.2.1 T_z: translation along depth axis

In this case, we have $\theta = 0, t_x = t_y = 0$

$$D_{T_z}(P_0, \alpha) = Z_0 - t_z, \quad \rho_{T_z}(P_0, \alpha) = (\frac{f_xX_0 + c_x(Z_0 - t_z)}{Z_0 - t_z}, \frac{f_yY_0 + c_y(Z_0 - t_z)}{Z_0 - t_z})$$

A.2.2 T_x: translation along depth-orthogonal horizontal axis

In this case, we have $\theta = 0, t_z = t_y = 0$

$$D_{T_x}(P_0, \alpha) = Z_0, \quad \rho_{T_x}(P_0, \alpha) = (\frac{f_x(X_0 - t_x) + c_xZ_0}{Z_0}, \frac{f_yY_0 + c_yZ_0}{Z_0})$$

A.2.3 T_y: translation along depth-orthogonal vertical axis

In this case, we have $\theta = 0, t_z = t_x = 0$

$$D_{T_y}(P_0, \alpha) = Z_0, \quad \rho_{T_y}(P_0, \alpha) = (\frac{f_xX_0 + c_xZ_0}{Z_0}, \frac{f_y(Y_0 - t_y) + c_yZ_0}{Z_0})$$

A.2.4 R_z: rotation around depth roll axis

In this case, we have $n_1 = n_2 = 0, n_3 = 1, t_x = t_y = t_z = 0$

$$D_{R_z}(P_0, \alpha) = Z_0, \quad \rho_{R_z}(P_0, \alpha) = (\frac{f_x\cos\theta X_0 + f_x\sin\theta Y_0}{Z_0} + c_x, \frac{f_y\cos\theta Y_0 + f_y\sin\theta X_0}{Z_0} + c_y)$$

A.2.5 R_x: rotation around depth-orthogonal pitch axis

In this case, we have $n_2 = n_3 = 0, n_1 = 1, t_x = t_y = t_z = 0$

$$\begin{aligned} D_{R_x}(P_0, \alpha) &= -\sin\theta Y_0 + \cos\theta Z_0 \\ \rho_{R_x}(P_0, \alpha) &= (\frac{f_xX_0}{-Y_0\sin\theta + Z_0\cos\theta} + c_x, \frac{Y_0\cos\theta + Z_0\sin\theta}{-Y_0\sin\theta + Z_0\cos\theta}f_y + c_y) \end{aligned}$$

A.2.6 R_y: rotation around depth-orthogonal yaw axis

In this case, we have $n_1 = n_3 = 0, n_2 = 1, t_x = t_y = t_z = 0$

$$D_{R_y}(P_0, \alpha) = \sin\theta X_0 + \cos\theta Z_0$$
$$\rho_{R_y}(P_0, \alpha) = (\frac{X_0\cos\theta - Z_0\sin\theta}{X_0\sin\theta + Z_0\cos\theta} f_x + c_x, \frac{f_y Y_0}{X_0\sin\theta + Z_0\cos\theta} + c_y)$$

A.3 Proof of main text Lemma 1

Lemma 1 (**restated** of main text Lemma 1, Compatible Relative Projection with Global Projection)**.** *With a global projective transformation $O : \mathbb{V} \times \mathcal{Z} \to \mathcal{X}$ from 3D point cloud and a relative projective transformation $\phi : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ given some original camera motions, for any $\alpha_1 \in \mathcal{Z}$ there exists an injective, continuously differentiable and non-vanishing-Jacobian function $\gamma_{\alpha_1} : \mathcal{Z} \to \mathcal{Z}$ such that*

$$\phi(O(V, \alpha_1), \alpha_2) = O(V, \gamma_{\alpha_1}(\alpha_2)), V \in \mathbb{V}, \alpha_2 \in \mathcal{Z}. \tag{5}$$

Proof. Given the fixed colored point cloud map $V \in \mathbb{V} = (\mathbb{R}^C, \mathbb{P})$, decompose the sequential relative camera motions α_1, α_2 into R_1, t_1 and R_2, t_2, where $\alpha_1 = (\theta_1, t_1), R_1 = \exp((\theta_1)^\wedge) \in SO(3)$ and $\alpha_2 = (\theta_2, t_2), R_2 = \exp((\theta_2)^\wedge) \in SO(3)$. Following Definition 1, for any fixed 3D point $P_0 \in \mathbb{P}$ under the initial camera pose, denote the coordinate after each relative camera motion as P_1, P_2, we have,

$$P_0 = R_1 P_1 + t_1, P_1 = R_2 P_2 + t_2$$

Therefore, the composed relative camera motion is derived as,

$$P_0 = R_{1,2} P_2 + t_{1,2} = R_1 R_2 P_2 + (R_1 t_2 + t_1)$$

So the composition of camera motion is $\alpha_1 \circ \alpha_2 = (\theta_{1,2}, t_{1,2}) = (\theta_{1,2}, R_1 t_2 + t_1)$, where $R_{1,2} = \exp((\theta_{1,2})^\wedge) = R_1 R_2$. Let the γ function in (5) be the composition of camera motion, i.e., $\gamma_{\alpha_1}(\alpha_2) = \alpha_1 \circ \alpha_2$, where the projection function with min-pooling in Definition 2 is also satisfied. Specifically, if the rotation is around a fixed axis, $\gamma_{\alpha_1}(\alpha_2) = \alpha_1 \circ \alpha_2 = \alpha_1 + \alpha_2$ holds due to the special case of multiplication in $SO(3)$.

Based on Definition 2, denote the point cloud coordinates after camera motion α_1 to be V^{α_1}, where the projected image is

$$O(V, \alpha_1) = O(V^{\alpha_1}, 0) = x^{\alpha_1}$$

Then based on the composition of camera motion, it holds that

$$O(V, \gamma_{\alpha_1}(\alpha_2)) = O(V, \alpha_1 \circ \alpha_2) = O(V^{\alpha_1}, \alpha_2)$$

Following Equation (3) in Definition 2, we have $\phi(x^{\alpha_1}, \alpha_2) = O(V^{\alpha_1}, \alpha_2)$ So combining the derivations above, we have

$$\begin{aligned}
\phi(O(V, \alpha_1), \alpha_2) &= \phi(O(V^{\alpha_1}, 0), \alpha_2) \\
&= \phi(x^{\alpha_1}, \alpha_2) \\
&= O(V^{\alpha_1}, \alpha_2) \\
&= O(V, \alpha_1 \circ \alpha_2) \\
&= O(V, \gamma_{\alpha_1}(\alpha_2))
\end{aligned}$$

which concludes the proof. □

A.4 Proof of main text Theorem 1

Lemma 2 (Corollary 7 in [1], Corollary 3 in [2])**.** *Suppose $\mathcal{Z} = \mathbb{R}^m$, $\Sigma := \mathrm{diag}(\sigma_1^2, \ldots, \sigma_m^2)$, $\varepsilon_0 \sim \mathcal{N}(0, \Sigma)$ and $\varepsilon_1 := \alpha + \varepsilon_0$ for some $\alpha \in \mathbb{R}^m$. Suppose that $y_A = g(x; \varepsilon_0)$ at $x \in \mathcal{X}$ for some $y_A \in \mathcal{Y}$ and let $p_A, p_B \in [0, 1]$ be bounds to the class probabilities, i.e.,*

$$q(y_A \mid x, \varepsilon_0) \geq p_A > p_B \geq \max_{y \neq y_A} q(y \mid x, \varepsilon_0). \tag{6}$$

Then, it holds that $q(y_A|\, x;\, \varepsilon_1) > \max_{y \neq y_A} q(y|\, x;\, \varepsilon_1)$ *if* α *satisfies*

$$\sqrt{\sum_{i=1}^{m} \left(\frac{\alpha_i}{\sigma_i}\right)^2} < \frac{1}{2}\left(\Phi^{-1}(p_A) - \Phi^{-1}(p_B)\right). \tag{7}$$

The rigorous proof on Lemma2 can be found in [1].

Theorem 1 (**restated** of main text Theorem 1, Robustness certification under camera motion with fixed-axis rotation)**.** *Let* $\alpha \in \mathcal{Z} \subset \mathbb{R}^6$ *be the parameters of projective transformation* ϕ *with translation* $(t_x, t_y, t_z)^T \in \mathbb{R}^3$ *and fixed-axis rotation* $(\theta n_1, \theta n_2, \theta n_3)^T \in \mathbb{R}^3, \sum_{i=1}^{3} n_i^2 = 1$*, suppose the composed camera motion* $\varepsilon_1 \in \mathcal{Z}$ *satisfies* $\phi(x, \varepsilon_1) = \phi(\phi(x, \varepsilon_0), \alpha)$ *given some* $\alpha \in \mathcal{Z}$ *and zero-mean Gaussian motion* ε_0 *with variance* $\sigma_x^2, \sigma_y^2, \sigma_z^2, \sigma_\theta^2$ *for* t_x, t_y, t_z, θ *respectively, let* $p_A, p_B \in [0, 1]$ *be bounds of the top-2 class probabilities for the motion smoothed model, i.e.,*

$$q(y_A \mid x, \varepsilon_0) \geq p_A > p_B \geq \max_{y \neq y_A} q(y \mid x, \varepsilon_0). \tag{8}$$

Then, it holds that $g(\phi(x, \alpha); \varepsilon_0) = g(x; \varepsilon_0)$ *if* $\alpha = (t_x, t_y, t_z, \theta n_1, \theta n_2, \theta n_3)^T$ *satisfies*

$$\sqrt{\left(\frac{\theta}{\sigma_\theta}\right)^2 + \left(\frac{t_x}{\sigma_x}\right)^2 + \left(\frac{t_y}{\sigma_y}\right)^2 + \left(\frac{t_z}{\sigma_z}\right)^2} < \frac{1}{2}\left(\Phi^{-1}(p_A) - \Phi^{-1}(p_B)\right). \tag{9}$$

Proof. For the original transformation ϕ parameterized with $\alpha = (t_x, t_y, t_z, \theta n_1, \theta n_2, \theta n_3)^T \in \mathcal{Z}_\phi \subset \mathbb{R}^6$ with the fixed normalized rotation axis (n_1, n_2, n_3), we have the Gaussian noise for each entry $t_x \sim \mathcal{N}(0, \sigma_x), t_y \sim \mathcal{N}(0, \sigma_y), t_z \sim \mathcal{N}(0, \sigma_z), \theta n_1 \sim \mathcal{N}(0, \sigma_\theta^2 n_1^2), \theta n_2 \sim \mathcal{N}(0, \sigma_\theta^2 n_2^2), \theta n_3 \sim \mathcal{N}(0, \sigma_\theta^2 n_3^2)$. We can find the covariance matrix

$$\Sigma = \mathbb{E}[(\alpha - \mu_\alpha)(\alpha - \mu_\alpha)^\top] = \begin{pmatrix} \sigma_x^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_x^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_x^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & n_1^2\sigma_\theta^2 & n_1 n_2 \sigma_\theta^2 & n_1 n_3 \sigma_\theta^2 \\ 0 & 0 & 0 & n_1 n_2 \sigma_\theta^2 & n_2^2 \sigma_\theta^2 & n_2 n_3 \sigma_\theta^2 \\ 0 & 0 & 0 & n_1 n_3 \sigma_\theta^2 & n_2 n_3 \sigma_\theta^2 & n_3^2 \sigma_\theta^2 \end{pmatrix}$$

Note that since the the last three entries regarding rotation angle θ is correlated, the covariance matrix is not full rank and not positive definite. Therefore, we can find the non-singular linear transformation A to make all entries independent in $\tilde{\alpha} = A\alpha \in \tilde{\mathcal{Z}}_\phi \subset \mathbb{R}^6$. Specifically,

$$\tilde{\alpha} = A\alpha = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & n_1 & n_2 & n_3 \\ 0 & 0 & 0 & n_1 & \frac{1}{n_2} - n_2 & n_3 \\ 0 & 0 & 0 & n_1 & n_2 & \frac{1}{n_3} - n_3 \end{pmatrix} \begin{pmatrix} t_x \\ t_y \\ t_z \\ \theta n_1 \\ \theta n_2 \\ \theta n_3 \end{pmatrix} = \begin{pmatrix} t_x \\ t_y \\ t_z \\ \theta \\ 0 \\ 0 \end{pmatrix}$$

Then for $\tilde{\alpha}, \tilde{\beta}$ from the transformed parameter space where the transformation is additive $\gamma_{\tilde{\alpha}}(\tilde{\beta}) = \tilde{\alpha} + \tilde{\beta}$, we find the covariance matrix as

$$\tilde{\Sigma} = \mathbb{E}[(\tilde{\alpha} - \mu_{\tilde{\alpha}})(\tilde{\alpha} - \mu_{\tilde{\alpha}})^\top] = \mathrm{diag}(\sigma_x^2, \sigma_y^2, \sigma_z^2, \sigma_\theta^2, 0, 0)$$

For the projective transformation $\tilde{\phi}$ parameterized in space $\tilde{\mathcal{Z}}_\phi$, we have

$$\tilde{\phi}(x, \tilde{\varepsilon}) = \tilde{\phi}(x, A\varepsilon) = \tilde{O}(V, A\varepsilon) = O(V, \varepsilon) = \phi(x, \varepsilon)$$

. Therefore, Lemma 1 holds for projective transformation $\tilde{\phi}$ over parameter space $\tilde{\mathcal{Z}}_\phi$. Therefore, for the composed transformation parameterized in space $\tilde{\mathcal{Z}}_\phi$ we have

$$\tilde{\phi}(x, \tilde{\varepsilon}_1) = \tilde{\phi}(\tilde{\phi}(x, \tilde{\varepsilon}_0), \tilde{\alpha}) = \tilde{\phi}(\tilde{O}(V, \tilde{\varepsilon}_0), \tilde{\alpha}) = \tilde{O}(V, \gamma_{\tilde{\varepsilon}_0}(\tilde{\alpha})) = \tilde{O}(V, \tilde{\varepsilon}_0 + \tilde{\alpha}) = \tilde{\phi}(x, \tilde{\varepsilon}_0 + \tilde{\alpha})$$
$$\implies \tilde{\varepsilon}_1 = \tilde{\varepsilon}_0 + \tilde{\alpha}$$

Together with $\tilde{\mathcal{Z}}_\phi = \mathbb{R}^6$, $\tilde{\Sigma} = \mathrm{diag}(\sigma_x^2, \sigma_y^2, \sigma_z^2, \sigma_\theta^2, 0, 0)$, we have $\tilde{\varepsilon}_0 \sim \mathcal{N}(0, \tilde{\Sigma})$ and $\tilde{\varepsilon}_1 := \tilde{\alpha} + \tilde{\varepsilon}_0$ for $\tilde{\alpha} = (t_x, t_y, t_z, \theta, 0, 0)^T \in \mathbb{R}^6$,

$$\sqrt{\sum_{i=1}^{6} \left(\frac{\alpha_i}{\sigma_i}\right)^2} = \sqrt{\left(\frac{\theta}{\sigma_\theta}\right)^2 + \left(\frac{t_x}{\sigma_x}\right)^2 + \left(\frac{t_y}{\sigma_y}\right)^2 + \left(\frac{t_z}{\sigma_z}\right)^2}$$

the smoothed classifier for $p_A - p_B$-confidence condition of (6) in Lemma 2 is satisfied based on ,

$$q(y \mid x; \tilde{\varepsilon}) = \mathbb{E}_{\tilde{\varepsilon} \sim \mathcal{P}_{\tilde{\varepsilon}}} p(y \mid \tilde{\phi}(x, \tilde{\varepsilon})) = \mathbb{E}_{\varepsilon \sim \mathcal{P}_\varepsilon} p(y \mid \tilde{\phi}(x, A\varepsilon)) = \mathbb{E}_{\varepsilon \sim \mathcal{P}_\varepsilon} p(y \mid \phi(x, \varepsilon)) = q(y \mid x; \varepsilon)$$

so by Lemma 2, it holds that

$$q(y_A | \, x; \, \varepsilon_1) = q(y_A | \, x; \, \tilde{\varepsilon}_1) > \max_{y \neq y_A} q(y | \, x; \, \tilde{\varepsilon}_1) = \max_{y \neq y_A} q(y | \, x; \, \varepsilon_1)$$

Then according to Definition 3, we have

$$g(x; \varepsilon_1) = \operatorname*{argmax}_{y \in \mathcal{Y}} q(y \mid x; \varepsilon) = y_A = g(x; \tilde{\varepsilon}_0) = g(x; \varepsilon_0) \tag{10}$$

Furthermore, combining Definition 2, Definition 3 and Lemma 1, it holds that

$$\begin{aligned}
g(\phi(x, \alpha); \varepsilon_0) &= g(O(V, \alpha); \varepsilon_0) && \text{(By Definition 2)} \\
&= \operatorname*{argmax}_{y \in \mathcal{Y}} \mathbb{E}_{\varepsilon_0 \sim \mathcal{P}_\varepsilon} p(y \mid \phi(O(V, \alpha), \varepsilon_0)) && \text{(By Definition 3)} \\
&= \operatorname*{argmax}_{y \in \mathcal{Y}} \mathbb{E}_{\varepsilon_0 \sim \mathcal{P}_\varepsilon} p(y \mid O(V, \gamma_\alpha(\varepsilon_0))) && \text{(By Lemma 1)} \\
&= \operatorname*{argmax}_{y \in \mathcal{Y}} \mathbb{E}_{\varepsilon_0 \sim \mathcal{P}_\varepsilon} p(y \mid O(V, \varepsilon_1)) \\
&= \operatorname*{argmax}_{y \in \mathcal{Y}} \mathbb{E}_{\varepsilon_0 \sim \mathcal{P}_\varepsilon} p(y \mid \phi(x; \varepsilon_1)) && \text{(By Definition 2)} \\
&= g(x; \varepsilon_1) && \text{(By Definition 3)} \\
&= g(x; \varepsilon_0) && \text{(By Lemma 2 and Equation (10))}
\end{aligned}$$

which concludes the proof. □

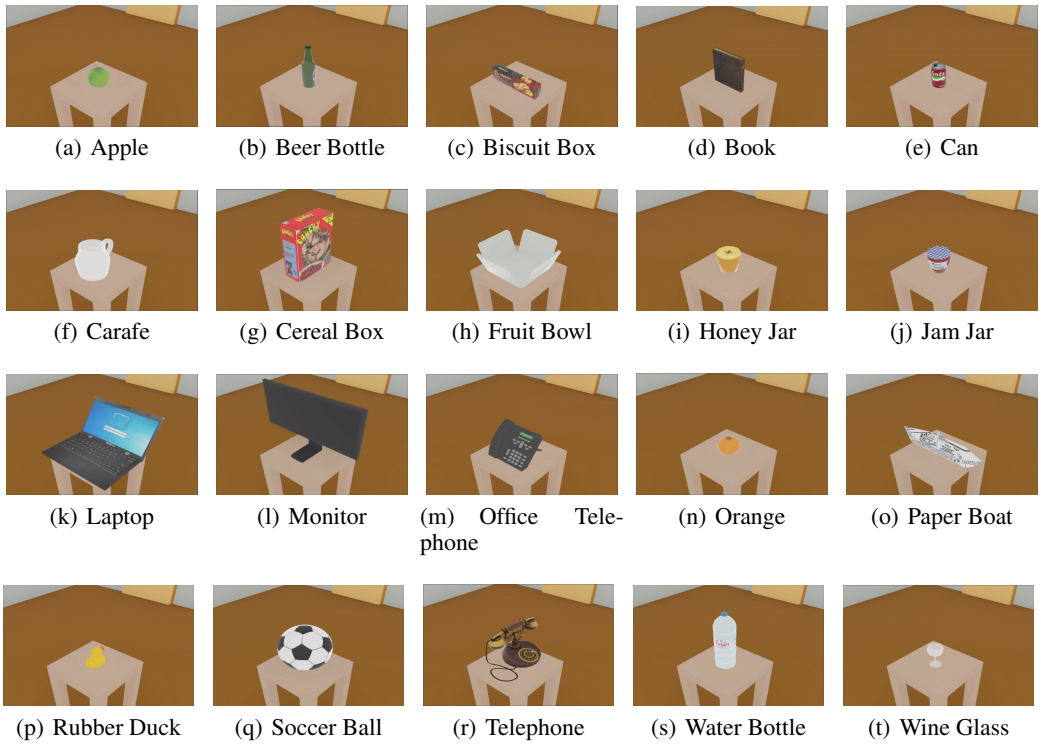
B More Experiment Details

B.1 MetaRoom Dataset

The entire room contains four surrounding walls with a length of 6 m and a height of 3.7 m, a ceiling and floor with size of 6 m × 6 m. On the walls, there is a door with the default size, a window with the default size, two paintings, two photographs, two closets, a blackboard, and a clock. In the center of the room, there is a small four-leg table with the size of 0.5 m × 0.5 m and the height of 1 m. In order to make the reconstructed point cloud have a consistent appearance under different camera perspectives, the texture of all the walls, ceiling, floor, door, window, and the table is *Roughcast* to avoid reflections. All the objects are listed in Figure 1.

To collect the whole point cloud, we first collect point cloud maps for table objects, background walls, floor, and ceiling, respectively, and then merge them together with downsampling to the density of 0.0025 m. To collect each point cloud map, we rotate the camera around the object where the focal length of the camera is 3090.194 and resolution is 1280×720. We collect the camera poses and images with the traverses under roll angle of $-5°, 0°, 5°$, yaw angle of $0° \sim 360°$ with the interval of $10°$, pitch angle between $30° \sim 80°$ with the interval of $5°$ and radius of 2.1 m, 2.5 m, 3.0 m, 3.4 m for table object, background walls, floor and ceiling with corresponding camera orientations. See the attached code for more details.

For the collection of the training set and test set, we use the collection strategy stated in Section 4.1 of the main text. To speed up the image projection for testing and evaluation, we first collect images with poses using focal length of 3090.194 and resolution of 1280×720 to construct the local point



(a) Apple (b) Beer Bottle (c) Biscuit Box (d) Book (e) Can

(f) Carafe (g) Cereal Box (h) Fruit Bowl (i) Honey Jar (j) Jam Jar

(k) Laptop (l) Monitor (m) Office Telephone (n) Orange (o) Paper Boat

(p) Rubber Duck (q) Soccer Ball (r) Telephone (s) Water Bottle (t) Wine Glass

Figure 1: All the objects in the MetaRoom dataset

Camera motion type	T_z	T_x	T_y	R_z	R_x	R_y
Motion Agumentation (Gaussian variance)	0.1 m	0.05 m	0.05 m	0.122 rad (7°)	0.0436 rad (2.5°)	0.0436 rad (2.5°)

Table 1: Motion augmentation details

cloud within each perturbation radius. Then collect the test camera poses with the intrinsic matrix of the focal length of 386.274 and resolution of 160×90, where the low-resolution images are directly used for model training and evaluation without resizing to rigorously follow Definition 2. For model training with data augmentation, we collect all images under perturbations offline for computational efficiency, which is consistent with training epochs of undefended vanilla models.

B.2 Model Training

To train the base classifiers, we train the ResNet-18 and ResNet-50 models from random initialization for both motions augmented and undefended vanilla models. The motion augmentation for each training sample is implemented with perturbations under Gaussian distribution and σ for each axis is shown in Figure 1. The inputs are without resizing or other image-based augmented for a clean and fair comparison, followed by channel normalization to 0.5 mean and variance. The models are trained with a batch size of 32 and a learning rate of 0.001 for 100 epochs. We remark that the goal of the model training is to ensure that the base classifiers can perform normally on the test set without overfitting or underfitting and we focus more on the evaluation for robustness analysis, so we did not fully explore the training potential. The performance of course can be further improved through tuning the hyper parameters and model architectures in more effective ways. All the experiments are conducted on NVIDIA A6000 with 48G GPU and 128G RAM.

Camera motion type	Uniform Down Sample (every k points)	Uniform Down Sample (density: m)
Translation Z	7	0.0133
Translation X	6	0.01365
Translation Y	7	0.0137
Rotation Z	7	0.0135
Rotation X	6	0.01355
Rotation Y	7	0.0134

Table 2: Hyperparameters for two-stage down sampling to speed up smoothing

Camera Motion Types	Smoothed ResNet18	Smoothed ResNet50
T_z, radius [-0.1m, 0.1m]	Vanilla / Motion Aug.	Vanilla / Motion Aug.
5-perturbed Emp. Robust Acc.	0.817 / 0.833	0.617 / 0.850
100-perturbed Emp. Robust Acc.	0.783 / 0.817	0.567 / 0.825
T_x, radius [-0.05m, 0.05m]	Vanilla / Motion Aug.	Vanilla / Motion Aug.
5-perturbed Emp. Robust Acc.	0.783 / 0.875	0.675 / 0.825
100-perturbed Emp. Robust Acc.	0.758 / 0.867	0.617 / 0.800
T_y, radius [-0.05m, 0.05m]	Vanilla / Motion Aug.	Vanilla / Motion Aug.
5-perturbed Emp. Robust Acc.	0.825 / 0.875	0.767 / 0.925
100-perturbed Emp. Robust Acc.	0.792 / 0.842	0.758 / 0.908
R_z, radius [-7°, 7°]	Vanilla / Motion Aug.	Vanilla / Motion Aug.
5-perturbed Emp. Robust Acc.	0.742 / 0.933	0.717 / 0.917
100-perturbed Emp. Robust Acc.	0.717 / 0.892	0.675 / 0.917
R_x, radius [-2.5°, 2.5°]	Vanilla / Motion Aug.	Vanilla / Motion Aug.
5-perturbed Emp. Robust Acc.	0.800 / 0.942	0.742 / 0.933
100-perturbed Emp. Robust Acc.	0.750 / 0.892	0.692 / 0.917
R_y, radius [-2.5°, 2.5°]	Vanilla / Motion Aug.	Vanilla / Motion Aug.
5-perturbed Emp. Robust Acc.	0.875 / 0.925	0.783 / 0.992
100-perturbed Emp. Robust Acc.	0.808 / 0.925	0.742 / 0.983

Table 3: Comparison of performance in terms of 5 and 100 perturbed empirical robust accuracy.

B.3 Evaluation Details

Benign and empirical robust accuracy for base models. To evaluate base models, we only use the metrics of benign accuracy and empirical robust accuracy because certified accuracy cannot be obtained without smoothing. The base classifiers are the trained models and we test them directly on the test set for benign accuracy while we use offline motion perturbed images under uniform distribution around each test sample to obtain the worst-case empirical robust accuracy.

Benign, empirical robust accuracy for smoothed models. For the smoothing model, since the smoothing is with Gaussian distribution through online Monte Carlo sampling for each test image, we adopt online perturbation within a certain radius given the point cloud and camera pose in the test set to obtain the benign and empirical accuracy for a fair comparison. For the empirical accuracy through Monte Carlo, we adopt 100 samples to obtain top-2 classes with the confidence of 99% and batch size of 100 for the smoothed classifier following [3].

Comparison between 5 and 100 perturbed empirical robust accuracy. From Table 3, it can be seen that grid search based attack 100-perturbed attacks are just a little bit stronger than 5-perturbed ones, so either can be used to approximate the worst-case adversarial samples although the gradient-based attack cannot be implemented directly in the camera motion transformation space [4, 5]. Note that the gap between 5-perturbed and 100-perturbed attacks is less for motion augmented models compared to undefended vanilla models, showing that motion augmentation can improve empirical robustness against uniform perturbation.

Camera Motion Types	Vanilla ResNet18	Vanilla ResNet50
T_z, radius [-0.1m, 0.1m]	Base / Smoothed	Base / Smoothed
Benign Accuracy	0.800 / **0.858**	**0.708** / 0.675
100-perturbed Emp. Robust Acc.	0.708 / **0.783**	0.517 / **0.567**
T_x, radius [-0.05m, 0.05m]	Base / Smoothed	Base / Smoothed
Benign Accuracy	0.817 / **0.825**	0.717 / **0.767**
100-perturbed Emp. Robust Acc.	0.608 / **0.758**	0.467 / **0.617**
T_y, radius [-0.05m, 0.05m]	Base / Smoothed	Base / Smoothed
Benign Accuracy	0.825 / **0.850**	**0.817** / 0.792
100-perturbed Emp. Robust Acc.	0.675 / **0.792**	0.674 / **0.758**
R_z, radius [-7°, 7°]	Base / Smoothed	Base / Smoothed
Benign Accuracy	0.800 / **0.817**	**0.783** / 0.758
100-perturbed Emp. Robust Acc.	0.667 / **0.717**	0.558 / **0.675**
R_x, radius [-2.5°, 2.5°]	Base / Smoothed	Base / Smoothed
Benign Accuracy	**0.867** / 0.842	**0.767** / **0.767**
100-perturbed Emp. Robust Acc.	0.667 / **0.750**	0.467 / **0.692**
R_y, radius [-2.5°, 2.5°]	Base / Smoothed	Base / Smoothed
Benign Accuracy	**0.917** / 0.892	0.800 / **0.808**
100-perturbed Emp. Robust Acc.	0.692 / **0.808**	0.600 / **0.742**

Table 4: The comparison between base vanilla models and smoothed vanilla models in benign and 100-perturbed empirical robust accuracy for all camera motions. The higher one between each base and smoothed model is in **bold**.

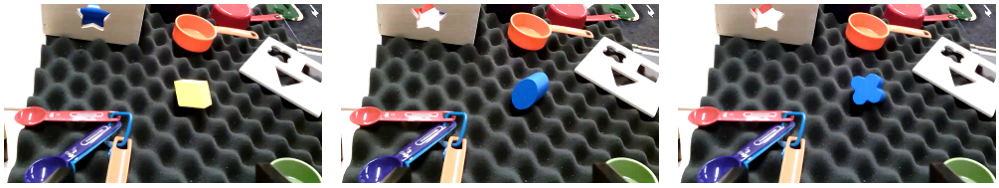
Certified accuracy for smoothed models. For the certification of the smoothed model, we also use Monte Carlo sampling for smoothing over confidence of 99% using 1000 samples and 100 samples to obtain the top-1 classes with a batch size of 200. To make the projection more efficient during smoothing, we use the local dense point cloud map reconstructed from maximum perturbations on each axis. Specifically, we adopt two-stage down-sampling strategy of `uniform_down_sample` and `voxel_down_sample`. The down-sampling hyperparameters are listed in Table 2.

Smoothed v.s. base vanilla model. For the undefended vanilla models, Table 4 presents that smoothed models have higher empirical robust accuracy and the gap between benign and empirical robust accuracy becomes less after motion smoothing compared to the base models, showing that smoothing strategy works not only for well-defended motion augmented models, but also for undefended vanilla models. Compared to main text Table 1, there is more robustness/accuracy trade-off in rotation for the vanilla models, which implies that motion augmentation as a defense in model training helps to improve the robustness better against rotational perturbations than translational perturbations.

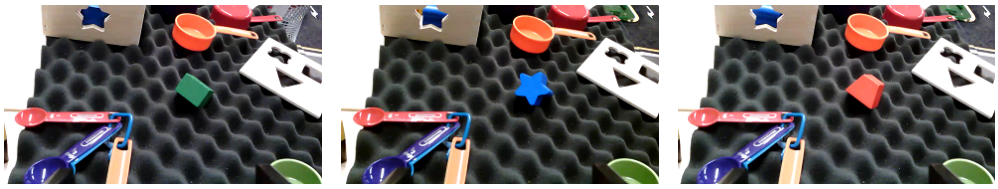
B.4 Real-world Robot Experiment Details

The working zone of the pick-place environment is on the table with the size of $1m \times 1m$. For the data collection, we first place each object $0.7m$ away from the robot base and randomly choose roll angles in $[-60°, 60°]$, pitch angles in $[35°, 65°]$, yaw angles in $[-30°, 30°]$ and radius in $[0.35m, 0.45m]$, capturing 2500 images along all the random waypoints using the default planning trajectories for each object. The non-overlapped gap is set between the training set and test set to choose 19 random poses, which are fixed for 6 objects as 114 test poses in total. The base model has well-trained over 5 epochs. At each perturbation, the smoothed model is with the 10 samples from zero-mean Gaussian distribution with variance $0.625cm$ for all translations and $1.25°$ for all rotations. Following the metrics in main text Section 4.1, for both the base model and smoothed model, we adopt 10 uniform samples over $[-1.25cm, 1.25cm]$ and $[-2.5°, 2.5°]$ as empirical robust accuracy. The objects used for the perception model can be seen in Figure 2. Since we do not do any defense or augmentation in model training, the results in the main text Table 3 are from the vanilla model. The gap to avoid the overlapping between the test set and training set is $20°$ for roll angle, $5°$ for pitch angle, $10°$ for yaw angle, and $1.6cm$ for radius. To make the robot application more practical, we remark that the smoothing method is used to improve empirical robustness so

we omit the benign accuracy using the smoothing method, which can be found in main text Table 1 and 4. The qualitative results can be found in Figure 3 to illustrate the smoothing process to improve the perception robustness. The video demo of real-world robot experiment can be found on https://www.youtube.com/watch?v=iCfRBk3O3CA.



(a) Diamond (b) Oval (c) Quatrefoil



(d) Rectangle (e) Star (f) Trapezoid

Figure 2: All the objects in the real-world robot experiment

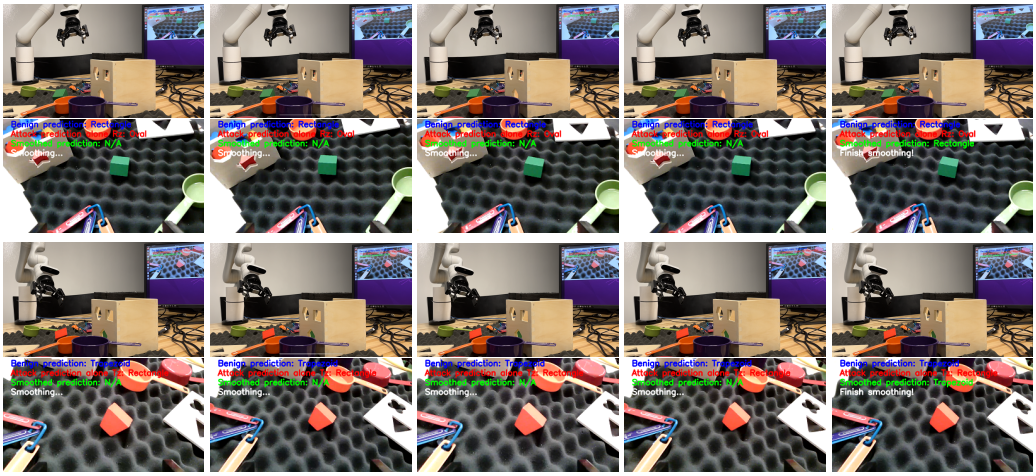


Figure 3: Smoothing process to improve robustness against camera motion of R_z (top) and T_z (bottom). The left four columns are randomized smoothing samples, and the right column is the classification result after smoothing.

B.5 Limitation and Discussion

One limitation of this work is that our current certification framework is only evaluated on image classification tasks, although it is fundamental for other robot applications. It can be extended to regression tasks through discretization and applied to object detection, keypoint detection, depth estimation, etc. In addition, the robustness certification procedure requires the prior 3D dense point cloud of the entire environment, which may be hard to obtain sometimes. It also costs many computational resources to obtain the guarantee and can be addressed through differential certifications [6, 1] as future work. Finally, the current certification framework is built upon the camera sensor, and it would be interesting to extend the current work to other perception sensors in robotics and autonomous driving, e.g. 3D LiDAR.

References

[1] L. Li, M. Weber, X. Xu, L. Rimanic, B. Kailkhura, T. Xie, C. Zhang, and B. Li. Tss: Transformation-specific smoothing for robustness certification. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 535–557, 2021.

[2] W. Chu, L. Li, and B. Li. Tpc: Transformation-specific smoothing for point cloud models. In *International Conference on Machine Learning*. PMLR, 2022.

[3] J. Cohen, E. Rosenfeld, and Z. Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320. PMLR, 2019.

[4] L. Engstrom, B. Tran, D. Tsipras, L. Schmidt, and A. Madry. Exploring the landscape of spatial robustness. In *International conference on machine learning*, pages 1802–1811. PMLR, 2019.

[5] C. Sitawarin, Z. J. Golan-Strieb, and D. Wagner. Demystifying the adversarial robustness of random transformation defenses. In *International Conference on Machine Learning*, pages 20232–20252. PMLR, 2022.

[6] L. Li, T. Xie, and B. Li. Sok: Certified robustness for deep neural networks. *arXiv preprint arXiv:2009.04131*, 2020.