MISSCLICK: EXECUTION-AWARE ADVERSARIAL ATTACKS ON COORDINATE GENERATION IN GUI GROUNDING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent GUI visual grounding models generate screen coordinates as digit-token sequences that are parsed into numerical values and mapped to executable clicks. This generation-to-execution interface creates an attack surface that existing objectives over visual representations or coordinate-token sequences do not explicitly model. Although each coordinate digit is predicted as a token, its spatial effect after parsing depends on decimal position: changing a hundreds-place digit by one shifts the coordinate by 100 units, whereas the same change at the ones place shifts it by one. This mismatch motivates attack objectives that account for both numerical coordinate structure and click execution. Moreover, untargeted and targeted attacks require different objectives because they aim to move the click outside the correct region and into an attacker-specified region, respectively. We propose **MissClick**, an execution-aware white-box attack that aligns optimization with click-level success conditions. MissClick-U maximizes soft-coordinate displacement for untargeted disruption, while MissClick-T minimizes a place-weighted target-digit loss for targeted redirection. On OS-Atlas and UGround across desktop, web, and mobile platforms, MissClick-U achieves untargeted success rates of 75.07% and 72.93% ($+16.62$ and $+30.72$ pp), while MissClick-T achieves targeted success rates of 44.86% and 62.67% ($+31.73$ and $+47.06$ pp). Among the evaluated objectives, soft-coordinate displacement performs best for untargeted attacks, whereas place-weighted target-digit optimization performs best for targeted attacks, supporting goal-specific execution-aware objective design.

1 INTRODUCTION

Recent advances in vision-language models have accelerated the development of GUI agents that autonomously operate graphical user interfaces (Hong et al., 2024; Zhang et al., 2025a; Qin et al., 2025; Wang et al., 2026; Lin et al., 2025). A core capability of these agents is visual grounding: given a screenshot and a natural-language instruction, the model predicts where to click. A dominant approach formulates this as coordinate generation (Gou et al., 2025; Wu et al., 2025c): the model autoregressively produces screen coordinates as sequences of digit tokens (one token per decimal digit), and the resulting string is parsed into coordinates that determine the executed click (Fig. 1).

The interface between coordinate generation and click execution creates an execution-level attack surface. Because the predicted click can trigger actions such as form submission or navigation, an adversarially shifted click can cause the agent to interact with an unintended UI element on behalf of the user. The security of GUI grounding models is therefore an important concern, yet prior work on adversarial perturbations against GUI grounding remains limited. Zhao et al. (2025) study both untargeted and targeted attacks against GUI grounders. Their objectives optimize either the distance between clean and perturbed image embeddings or the probability of generating a target coordinate sequence. These objectives are defined over intermediate representations or generated token sequences, whereas attack success is ultimately determined by the click executed after parsing the generated coordinates. Consequently, token-level objectives do not explicitly characterize the resulting coordinate changes or click displacement. These execution-level effects depend on which digit values change, their decimal positions, and how the parsed coordinates are mapped to an on-screen click location.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

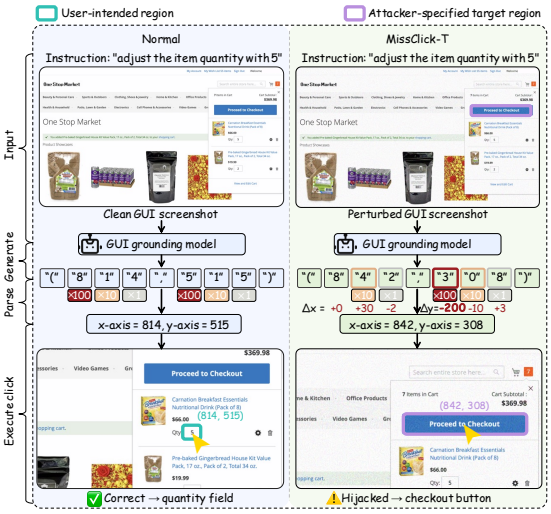


Figure 1: The coordinate generation pipeline of GUI grounding models and its security implications. Digit tokens are parsed into coordinates, with each position carrying a decimal place value ($\times 100$, $\times 10$, $\times 1$); changes to higher-order digits therefore produce larger coordinate shifts. Here, Δx and Δy denote the per-position changes relative to the clean output. Under an ℓ_∞-bounded perturbation ($\varepsilon = 16/255$), MissClick-T redirects the click from the intended quantity field to the checkout button. The example is from UGround on a web screenshot.

This mismatch has direct consequences for attack design because decimal notation assigns different numerical effects to different digit positions. Each coordinate component is assembled from its constituent digits according to their decimal positions: a hundreds-place digit is weighted by 100, a tens-place digit by 10, and a ones-place digit by 1 (Sec. 2.2, Equation 5). A single-digit change at a high-order position therefore shifts the executed click by a much larger distance than the same change at a low-order position. An attack objective should therefore account for this numerical structure. The appropriate objective also depends on the attack goal: an untargeted attack succeeds when the executed click falls outside the ground-truth region, whereas a targeted attack succeeds only when it produces a valid click inside an attacker-specified target region.

Based on these observations, we propose **MissClick**, an execution-aware white-box attack on coordinate-generating GUI grounding models. MissClick derives separate objectives for untargeted and targeted attacks from their click-level success conditions. For untargeted attacks, MissClick-U constructs differentiable soft coordinates from digit probabilities and decimal place values, maps them to a soft click, and maximizes its distance from the ground-truth click. For targeted attacks, MissClick-T minimizes a place-weighted target-digit loss that assigns larger weights to higher-order coordinate digits. This directly promotes the target digits during autoregressive generation, since matching an expected soft coordinate does not guarantee that they become the most probable tokens. We evaluate MissClick on OS-Atlas and UGround across desktop, web, and mobile platforms. MissClick-U achieves untargeted attack success rates of 75.07% and 72.93%, improving over the strongest baseline by 16.62 and 30.72 percentage points, while MissClick-T achieves targeted success rates of 44.86% and 62.67%, with gains of 31.73 and 47.06 percentage points. Among the evaluated objectives, soft-coordinate displacement performs best for untargeted attacks, whereas place-weighted target-digit optimization performs best for targeted attacks.

Our main contributions are as follows:

- We identify an execution-level attack surface at the interface between coordinate generation and click execution. Because digit values and decimal positions determine the parsed coordinates and resulting click displacement, coordinate outputs should be modeled according to their execution semantics rather than treated merely as ordinary text.
- We propose MissClick, an execution-aware attack with separate objectives for untargeted and targeted attacks: soft-coordinate displacement for untargeted attacks and place-weighted target-digit optimization for targeted attacks. Each objective is derived from the corresponding click-level success condition.
- Experiments on two representative GUI grounding models across desktop, web, and mobile platforms demonstrate improvements over existing methods in both attack settings. An objective comparison further shows that soft-coordinate displacement achieves the highest untargeted attack success rates, whereas place-weighted target-digit optimization achieves the highest targeted attack success rates.

2 METHOD

MissClick formulates its attack objectives from the click action produced after coordinate generation and numerical parsing. We first formalize this generation-to-execution pipeline and then derive separate objectives from the click-level success conditions of untargeted and targeted attacks.

2.1 PROBLEM FORMULATION

Let $\mathbf{I} \in [0,1]^d$ denote an input screenshot and q a task instruction, where $d = C \times H \times W$, and C, H, and W denote the number of channels, height, and width of the screenshot, respectively. Given $(\mathbf{I}, q)$, a coordinate-generating GUI grounding model $\mathcal{M}$ autoregressively generates a textual answer sequence $s = (s_1, \ldots, s_T)$. The coordinate string in s is parsed into numerical coordinates $c = \mathrm{Parse}_{\mathcal{M}}(s)$.

Different GUI grounding models may use different coordinate formats. For a bounding-box output $c = (x_1, y_1, x_2, y_2)$, the executed click is taken as the box center, $r(c) = \big((x_1{+}x_2)/2,\ (y_1{+}y_2)/2\big)$. For a point-coordinate output $c = (x, y)$, the executed click is $r(c) = c$. Together, $\mathcal{M}$, $\mathrm{Parse}_{\mathcal{M}}$, and r define the generation-to-execution mapping from the screenshot and instruction to the click evaluated by the environment. We express all coordinates in the normalized $[0, 1000]$ coordinate system. Let $B_{\mathrm{gt}} \subseteq [0, 1000]^2$ denote the ground-truth bounding box of the intended UI element for q. The grounding result is correct if the executed click falls within the ground-truth bounding box, i.e., $r(c) \in B_{\mathrm{gt}}$.

We consider a white-box threat model in which the adversary has full access to the victim model's architecture and weights and can compute gradients with respect to the input screenshot, while modifying neither the model parameters nor the task instruction. The goal of an untargeted attack is to craft an adversarial example $\widetilde{\mathbf{I}} = \mathbf{I} + \boldsymbol{\delta}$ such that the executed click falls outside the ground-truth bounding box. Formally, $\widetilde{\mathbf{I}}$ is an untargeted adversarial example under an ℓ_∞-norm bound of ε if

$$r(\mathrm{Parse}_{\mathcal{M}}\left(\mathcal{M}(\mathbf{I} + \boldsymbol{\delta}, q)\right)) \notin B_{\mathrm{gt}}, \tag{1}$$

subject to

$$\|\boldsymbol{\delta}\|_\infty \le \varepsilon, \qquad \mathbf{I} + \boldsymbol{\delta} \in [0,1]^d. \tag{2}$$

The goal of a targeted attack is to craft an adversarial example $\widetilde{\mathbf{I}} = \mathbf{I} + \boldsymbol{\delta}$ such that the executed click falls within an attacker-specified target region $B_{\mathrm{tgt}} \subseteq [0, 1000]^2$. Formally, $\widetilde{\mathbf{I}}$ is a targeted adversarial example if

$$r(\mathrm{Parse}_{\mathcal{M}}\left(\mathcal{M}(\mathbf{I} + \boldsymbol{\delta}, q)\right)) \in B_{\mathrm{tgt}}, \tag{3}$$

subject to the same perturbation constraints. During evaluation, an answer that cannot be parsed into valid coordinates yields no executable click and is therefore counted as a successful untargeted attack but an unsuccessful targeted attack.

2.2 COORDINATE-DIGIT STRUCTURE

Numerical interpretation. Let $c = (c_1, \ldots, c_K)$ denote the parsed coordinate output. For point outputs, $K = 2$ and $(c_1, c_2) = (x, y)$; for bounding-box outputs, $K = 4$ and $(c_1, c_2, c_3, c_4) = (x_1, y_1, x_2, y_2)$. For each $k \in \{1, \ldots, K\}$, suppose that the coordinate component c_k is represented by m_k decimal digits $\hat{d}_{k,1}, \ldots, \hat{d}_{k,m_k}$, where $\hat{d}_{k,j} \in \{0, \ldots, 9\}$ denotes the numerical value of the j-th digit in the greedily generated coordinate string. The parsed coordinate component is

$$c_k = \mathrm{int}\left(\hat{d}_{k,1} \cdots \hat{d}_{k,m_k}\right) = \sum_{j=1}^{m_k} a_{k,j} \hat{d}_{k,j}, \qquad a_{k,j} = 10^{m_k - j}, \tag{4}$$

where $a_{k,j}$ is the decimal place value of the j-th digit. For example, the coordinate 950 has digit values $(9, 5, 0)$ and place values $(100, 10, 1)$. Suppose that the j-th digit of coordinate component c_k changes from $\hat{d}_{k,j}$ to $\hat{d}'_{k,j}$ while the other digits remain unchanged. The corresponding coordinate change is

$$\Delta c_k = a_{k,j} \Delta d_{k,j}, \qquad \Delta d_{k,j} = \hat{d}'_{k,j} - \hat{d}_{k,j}. \tag{5}$$

The coordinate change depends on both the digit change and its decimal place value. For the same digit change, a higher-order position produces a larger coordinate displacement. For example, changing a digit by one at the hundreds place changes the coordinate value by 100, whereas the same change at the ones place changes it by 1. An adversarial perturbation that changes a high-order digit can therefore induce a large shift in the parsed coordinate and, through $r(\cdot)$, the executed click. This non-uniform mapping is the numerical structure that both MissClick objectives incorporate when optimizing the resulting click action.

Digit probability distributions. During autoregressive generation, the logits at later positions depend on previously generated tokens, so gradients cannot propagate through the discrete token-selection operations. We therefore perform a teacher-forced forward pass with the input screenshot, the task instruction, and a reference answer sequence containing the reference coordinate output c^{ref}. The reference coordinates follow the victim model's native output format. For untargeted attacks, c^{gt} is the coordinate tuple of B_{gt} for bounding-box outputs and $\mathrm{center}(B_{\mathrm{gt}})$ for point-coordinate outputs. For targeted attacks, c^{tgt} is the coordinate tuple of B_{tgt} for bounding-box outputs and $\mathrm{center}(B_{\mathrm{tgt}})$ for point-coordinate outputs. We set $c^{\mathrm{ref}} = c^{\mathrm{gt}}$ for untargeted attacks and $c^{\mathrm{ref}} = c^{\mathrm{tgt}}$ for targeted attacks. This yields next-token logits at each position of the reference answer sequence that are differentiable with respect to the input screenshot. From these logits, for the j-th digit of c_k^{ref}, we extract the full-vocabulary logit vector $\mathbf{z}_{k,j} \in \mathbb{R}^{|\mathcal{V}|}$. For the victim models considered in this work, each decimal digit is represented by an individual token. We then restrict $\mathbf{z}_{k,j}$ to the ten digit tokens (`0`–`9`) and apply softmax to obtain the digit probability distribution:

$$p_{k,j}(v) = \frac{\exp\left(z_{k,j}^{(v)}\right)}{\sum_{w=0}^{9} \exp\left(z_{k,j}^{(w)}\right)}, \qquad v \in \{0, \ldots, 9\}, \tag{6}$$

where $z_{k,j}^{(v)}$ is the logit assigned to digit token v.

Both attack objectives are built from $p_{k,j}(v)$ and the place values $a_{k,j}$, but use them differently because untargeted and targeted attacks impose distinct click-level success conditions. MissClick-U uses them to compute soft coordinates, whereas MissClick-T uses them to construct a place-weighted target-digit loss. Note that teacher forcing is used only during attack optimization. During evaluation, the model performs greedy autoregressive generation over the full vocabulary without the reference answer sequence.

2.3 Untargeted Attack

The success condition for an untargeted attack is that the executed click falls outside the ground-truth bounding box, i.e., $r(c) \notin B_{\mathrm{gt}}$. This click-level condition motivates a geometry-based objective that maximizes the displacement between the click and the ground-truth click $\mathbf{u}^{\mathrm{gt}} = \mathrm{center}(B_{\mathrm{gt}})$. Directly optimizing this objective with respect to the input perturbation requires a differentiable path from the perturbed screenshot to the click coordinates. The digit probability distributions $p_{k,j}$ obtained via teacher forcing (Equation 6) are differentiable with respect to the perturbed screenshot, but the executed click is obtained through argmax-based token selection, which is non-differentiable. Therefore, the click displacement cannot be directly differentiated with respect to the perturbed screenshot.

A standard differentiable alternative is a token-based cross-entropy objective over the ground-truth coordinate answer sequence. Maximizing this objective reduces the probability assigned to the ground-truth token at each position in the sequence. However, this token-based objective does not reflect the numerical effects of alternative digit values. It also does not account for the different displacement scales induced by digit changes at different decimal places, as characterized in equation 5.

MissClick-U therefore constructs a soft coordinate that is differentiable with respect to the perturbed screenshot and accounts for both digit values and their place-value scales:

$$\tilde{d}_{k,j} = \sum_{v=0}^{9} v\, p_{k,j}(v). \tag{7}$$

The soft coordinate component is then obtained by combining the expected digit values with their place values:

$$\tilde{c}_k = \sum_{j=1}^{m_k} a_{k,j}\, \tilde{d}_{k,j}. \tag{8}$$

Collecting all coordinate components gives $\tilde{c} = (\tilde{c}_1, \ldots, \tilde{c}_K)$. Equation 7 and equation 8 define a continuous relaxation of the coordinates obtained through discrete digit selection, using expected digit values under $p_{k,j}$ instead of argmax-selected digit values. This construction yields the differentiable path $\widetilde{\mathbf{I}} \to \mathbf{z}_{k,j} \to p_{k,j} \to \tilde{d}_{k,j} \to \tilde{c} \to r(\tilde{c})$. Moreover, this relaxation accounts for both the numerical effect of each digit value and the scale of each decimal position.

Applying the click mapping $r(\cdot)$ defined in Sec. 2.1, we define the MissClick-U objective as the squared distance between the soft click and the ground-truth click:

$$\mathcal{L}_{\mathrm{U}} = \left\| r(\tilde{c}) - \mathbf{u}^{\mathrm{gt}} \right\|_2^2. \tag{9}$$

MissClick-U maximizes $\mathcal{L}_{\mathrm{U}}$ with respect to the input perturbation, thereby increasing the displacement of the soft click from $\mathbf{u}^{\mathrm{gt}}$. Attack success is evaluated by checking whether the click obtained through greedy autoregressive generation falls outside B_{gt}.

2.4 Targeted Attack

The success condition for a targeted attack is that the executed click falls inside the attacker-specified target region, i.e., $r(c) \in B_{\mathrm{tgt}}$. In principle, any coordinate whose executed click falls inside B_{tgt} constitutes a successful attack. We instantiate this region-level goal with a representative target coordinate c^{tgt} in the victim model's native output format. For bounding-box outputs, c^{tgt} is the coordinate tuple of B_{tgt}; for point-coordinate outputs, $c^{\mathrm{tgt}} = \mathrm{center}(B_{\mathrm{tgt}})$. Given the soft-coordinate objective used for untargeted attacks, a natural approach for the targeted case would be to minimize the squared distance between the soft click and the target click $\mathbf{u}^{\mathrm{tgt}} = \mathrm{center}(B_{\mathrm{tgt}})$:

$$\mathcal{L}_{\text{soft-tgt}} = \left\| r(\tilde{c}) - \mathbf{u}^{\mathrm{tgt}} \right\|_2^2. \tag{10}$$

However, the soft coordinate is constructed from expected digit values under the teacher-forced distributions. During greedy autoregressive generation, a discrete token is selected by argmax at each step. The resulting coordinate string is then parsed into numerical coordinates according to equation 4. Targeted success is therefore determined by whether minimizing $\mathcal{L}_{\text{soft-tgt}}$ drives the argmax-selected digits toward the target. Yet matching an expected digit to the corresponding target value does not guarantee that the target digit is the argmax of $p_{k,j}$:

$$\tilde{d}_{k,j} = d_{k,j}^{\mathrm{tgt}} \;\not\Rightarrow\; \operatorname*{argmax}_{v \in \{0,\ldots,9\}} p_{k,j}(v) = d_{k,j}^{\mathrm{tgt}}, \tag{11}$$

where $d_{k,j}^{\mathrm{tgt}} \in \{0, \ldots, 9\}$ denotes the j-th digit of the target reference coordinate component c_k^{tgt}. For example, consider a hundreds-place target digit $d_{k,1}^{\mathrm{tgt}} = 5$ and the distribution $p_{k,1}(1) = 0.48$, $p_{k,1}(8) = 0.16$, and $p_{k,1}(9) = 0.36$. Its expected digit value is $\tilde{d}_{k,1} = 1(0.48)+8(0.16)+9(0.36) = 5$, although the most probable digit under $p_{k,1}$ is 1 rather than the target digit 5. Therefore, matching expected digit values does not guarantee that the target digit is the argmax of $p_{k,j}$. Consequently, the soft-coordinate objective does not directly optimize the discrete digit selections that determine the executed click and is therefore less directly aligned with targeted success than target-digit optimization.

Place-weighted target-digit loss. MissClick-T therefore directly increases the probability assigned to the target digit at each position, encouraging it to become the argmax of $p_{k,j}$ rather than merely matching a soft coordinate value. A standard approach is to minimize the cross-entropy $-\log p_{k,j}(d_{k,j}^{\mathrm{tgt}})$ at each digit position with uniform weights. However, uniform weighting does not reflect the place-value structure: a one-unit digit error at the hundreds place changes the coordinate value by 100, whereas the same error at the ones place changes it by only 1 (Equation 5). MissClick-T therefore weights each per-digit cross-entropy by its place value $a_{k,j}$:

$$\mathcal{L}_{\mathrm{T}} = -\frac{\sum_{k=1}^{K} \sum_{j=1}^{m_k} a_{k,j} \log p_{k,j}\left(d_{k,j}^{\mathrm{tgt}}\right)}{\sum_{k=1}^{K} \sum_{j=1}^{m_k} a_{k,j}}. \tag{12}$$

The place-value weight $a_{k,j}$ assigns a larger loss weight to high-order digit positions, whose errors produce larger coordinate displacements. The denominator converts the weighted sum into a normalized weighted average.

Minimizing $\mathcal{L}_{\mathrm{T}}$ increases the probability assigned to each target digit under $p_{k,j}$, thereby encouraging it to become the argmax. Following the same evaluation protocol as MissClick-U, attack success is evaluated by checking whether the executed click obtained through greedy autoregressive generation falls inside B_{tgt}.

2.5 Attack Optimization

MissClick keeps the victim model and its decoding procedure fixed and optimizes only the screenshot perturbation. Both MissClick-U and MissClick-T instantiate their execution-aware objectives using projected sign-gradient updates following PGD (Madry et al., 2018). At each iteration, the perturbation is updated as

$$\boldsymbol{\delta} \leftarrow \Pi_{[-\varepsilon,\,\varepsilon]^d}\left(\boldsymbol{\delta} \pm \alpha\ \mathrm{sign}\left(\nabla_{\boldsymbol{\delta}}\mathcal{L}\right)\right), \tag{13}$$

where α is the step size and $\Pi_{[-\varepsilon,\varepsilon]^d}$ denotes projection onto the ℓ_∞ perturbation set. MissClick-U uses the $+$ sign to maximize $\mathcal{L}_{\mathrm{U}}$ (Equation 9), whereas MissClick-T uses the $-$ sign to minimize $\mathcal{L}_{\mathrm{T}}$ (Equation 12).

At each iteration, the attack objective is computed from the digit distributions obtained via teacher forcing (Sec. 2.2), and its gradient is backpropagated to $\boldsymbol{\delta}$. The complete optimization procedure is provided in Algorithm 1 in Appendix A.1. The two attack modes therefore share the same optimization procedure but encode different execution-level goals: click displacement for untargeted disruption and discrete target-digit generation for targeted redirection.

3 Experiments

3.1 Experimental Setup

Victim models. We evaluate two coordinate-generating GUI grounding models: OS-Atlas-Base-7B (Wu et al., 2025c), which outputs bounding-box coordinates, and UGround-V1-7B (Gou et al., 2025), which outputs point coordinates. Both models autoregressively generate coordinates as digit-token sequences.

Dataset. ScreenSpot-v2 (Cheng et al., 2024; Wu et al., 2025c) contains 1,272 GUI grounding tasks across desktop, web, and mobile platforms. For targeted attacks, we use the 896 tasks containing a valid target element distinct from the ground-truth element.

Baselines. We compare against three baselines. *Random noise* samples a uniform perturbation within the same ε budget. *Token CE* maximizes the ground-truth coordinate sequence loss for untargeted attacks and minimizes the target coordinate sequence loss for targeted attacks. Following Zhao et al. (2025), *Representation attack* maximizes the MSE between clean and adversarial visual embeddings and is used only for untargeted attacks.

Attack setting and metrics. All attacks modify only the screenshot, leaving the model parameters and task instruction unchanged. Unless otherwise specified, all methods use $\varepsilon = 16/255$; iterative gradient-based attacks use random initialization, a step size of $\alpha = 1/255$, and $N = 100$ iterations. Teacher forcing is used only during optimization, whereas ASR is measured from greedy autoregressive outputs after coordinate parsing and click mapping. We evaluate only tasks correctly grounded on the clean screenshot. A valid output succeeds if its click falls outside B_{gt} for an untargeted attack or inside B_{tgt} for a targeted attack. Following the evaluator, an unparseable output counts as a successful untargeted attack and a failed targeted attack. The resulting evaluation sets contain 1,071 and 1,123 untargeted tasks, and 769 and 801 targeted tasks, for OS-Atlas and UGround, respectively.

Target region selection. For each targeted task, we uniformly sample B_{tgt} from annotated elements with zero IoU with B_{gt}. Appendices A and B provide implementation and evaluation details.

Table 1: Comparison of MissClick with existing baselines on ScreenSpot-v2. Attack success rates (ASR) are reported. ΔASR is MissClick's improvement over each baseline in percentage points (pp). *All* is the micro-averaged ASR over the three platforms.

Method	Untargeted Attacks									
	OS-Atlas					UGround				
	Desktop	Web	Mobile	All	ΔASR	Desktop	Web	Mobile	All	ΔASR
Random noise	18.32%	8.84%	6.88%	10.46%	+64.61 pp	5.42%	1.34%	3.29%	3.21%	+69.72 pp
Token CE	51.28%	45.30%	23.39%	37.91%	+37.16 pp	32.88%	27.15%	23.25%	27.07%	+45.86 pp
Representation	68.50%	61.88%	49.31%	58.45%	+16.62 pp	61.69%	40.32%	31.14%	42.21%	+30.72 pp
MissClick-U	**85.35%**	**79.83%**	**64.68%**	**75.07%**	–	**78.31%**	**79.57%**	**64.04%**	**72.93%**	–
Method	Targeted Attacks									
	OS-Atlas					UGround				
	Desktop	Web	Mobile	All	ΔASR	Desktop	Web	Mobile	All	ΔASR
Token CE	22.37%	9.96%	11.56%	13.13%	+31.73 pp	27.33%	8.24%	16.07%	15.61%	+47.06 pp
MissClick-T	**54.61%**	**49.08%**	**37.28%**	**44.86%**	–	**59.01%**	**59.50%**	**66.76%**	**62.67%**	–

Table 2: Comparison of attack objectives under untargeted and targeted settings on ScreenSpot-v2. Attack success rates (ASR) are reported. *All* is the micro-averaged ASR over the three platforms.

Attack Objective	Untargeted Attacks							
	OS-Atlas				UGround			
	Desktop	Web	Mobile	All	Desktop	Web	Mobile	All
Token CE	51.28%	45.30%	23.39%	37.91%	32.88%	27.15%	23.25%	27.07%
Soft-coordinate distance	**85.35%**	**79.83%**	**64.68%**	**75.07%**	**78.31%**	**79.57%**	**64.04%**	**72.93%**
Digit CE	58.97%	49.72%	29.59%	43.88%	32.88%	27.42%	22.81%	26.98%
Place-weighted digit CE	72.16%	69.06%	51.38%	62.65%	60.00%	62.37%	42.98%	53.87%
Attack Objective	Targeted Attacks							
	OS-Atlas				UGround			
	Desktop	Web	Mobile	All	Desktop	Web	Mobile	All
Token CE	22.37%	9.96%	11.56%	13.13%	27.33%	8.24%	16.07%	15.61%
Soft-coordinate distance	17.11%	14.76%	15.90%	15.73%	21.74%	23.30%	33.52%	27.59%
Digit CE	47.37%	36.90%	28.61%	35.24%	52.80%	44.80%	54.29%	50.69%
Place-weighted digit CE	**54.61%**	**49.08%**	**37.28%**	**44.86%**	**59.01%**	**59.50%**	**66.76%**	**62.67%**

3.2 Main Results

To evaluate the effectiveness of MissClick, we compare MissClick-U and MissClick-T with their respective baselines under the same attack settings on ScreenSpot-v2. Table 1 shows that MissClick-U outperforms all baselines on every model-platform combination. It achieves overall ASRs of 75.07% and 72.93% on OS-Atlas and UGround, exceeding Representation attack by 16.62 and 30.72 percentage points, respectively. MissClick-T also leads every model-platform combination, achieving overall ASRs of 44.86% and 62.67% and outperforming Token CE by 31.73 and 47.06 percentage points on OS-Atlas and UGround, respectively. It further achieves the highest targeted ASR across both element types and all target-area and distance groups. Even in the farthest-distance group, it outperforms Digit CE by 6.7 and 8.5 percentage points on the two models. Detailed results are reported in Appendix C.1 and Tables 9 and 10. All MissClick-U outputs are parseable (1,071 for OS-Atlas and 1,123 for UGround), as are all MissClick-T outputs (769 and 801, respectively). The reported ASRs therefore reflect executable misclicks rather than malformed generations. The breakdown is reported in Appendix Table 6.

3.3 Why Do Untargeted and Targeted Attacks Favor Different Objectives?

We compare Token CE, Soft-coordinate distance, Digit CE, and Place-weighted digit CE under identical attack settings to isolate the effect of objective design. Soft-coordinate distance optimizes click displacement, whereas the digit objectives optimize digit probabilities, with place-value weighting in the latter. Results are shown in Table 2. For the untargeted setting, Soft-coordinate distance achieves the highest overall ASR on both models, reaching 75.07% on OS-Atlas and 72.93% on UGround. The strong untargeted result of Soft-coordinate distance is consistent with its direct optimization of soft-click displacement. For the targeted setting, Place-weighted digit CE achieves the highest overall ASR, with 44.86% and 62.67% on OS-Atlas and UGround, respectively. Soft-

coordinate distance is less effective than the digit-based objectives in the targeted setting, achieving 15.73% and 27.59%, whereas Digit CE achieves 35.24% and 50.69%, respectively. As discussed in Sec. 2.4, moving the expected digit values toward the target does not guarantee that the target digits become the argmax. This indicates that directly optimizing target-digit probabilities better matches the targeted attack goal than Soft-coordinate distance. Under a common optimization procedure, the two attack goals favor different objectives: click displacement for untargeted disruption and discrete target-digit generation for targeted redirection. This result supports the goal-specific execution-aware design of MissClick-U and MissClick-T. Appendix D further relates the teacher-forced optimization signals to the final greedy-generated clicks. Appendix D.2 reports how place-value weighting changes target-digit probabilities and generated-digit match rates at the hundreds, tens, and ones positions.

3.4 Effect of Place-Value Weighting

We compare Digit CE, which assigns equal weight to all digit positions, with Place-weighted digit CE, which assigns weights according to decimal place values. All other attack settings remain identical, and the place-weighted loss is normalized by the total place weight. According to the targeted results reported in Table 2, Place-weighted digit CE outperforms Digit CE on both models and all three platforms. The overall targeted ASR improves from 35.24% to 44.86% on OS-Atlas, corresponding to an improvement of 9.62 percentage points, and from 50.69% to 62.67% on UGround, corresponding to an improvement of 11.98 percentage points. These results show that, within the targeted digit objective, accounting for the non-uniform geometric effects of digit positions through place-value weighting improves attack effectiveness compared with uniform weighting.

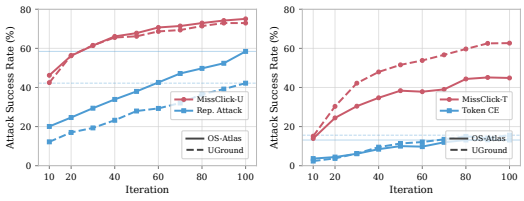
3.5 Further Analysis

Perturbation-budget sensitivity. We evaluate MissClick and the corresponding baselines under $\varepsilon \in \{4, 8, 16\}/255$ on fixed stratified subsets of 300 untargeted and targeted tasks, with all other settings unchanged. Table 3 reports the micro-averaged ASR over the three platforms. MissClick outperforms the baselines at every tested budget. Notably, MissClick-U at $\varepsilon = 4/255$ exceeds Representation attack at $\varepsilon = 16/255$ on UGround (54.34% vs. 39.25%), while MissClick-T shows the same cross-budget advantage on both models (24.91% vs. 13.75% and 34.67% vs. 17.15%).

Table 3: Attack success rates under different perturbation budgets ε on separate fixed stratified subsets of 300 tasks for the untargeted and targeted settings.

Method	Untargeted Attacks					
	OS-Atlas			UGround		
	ε=4	ε=8	ε=16	ε=4	ε=8	ε=16
Representation	33.73%	47.39%	55.42%	12.45%	29.06%	39.25%
MissClick-U	**54.62%**	**69.88%**	**75.90%**	**54.34%**	**64.53%**	**74.72%**
Method	Targeted Attacks					
	OS-Atlas			UGround		
	ε=4	ε=8	ε=16	ε=4	ε=8	ε=16
Token CE	3.72%	11.52%	13.75%	5.47%	11.68%	17.15%
MissClick-T	**24.91%**	**40.15%**	**46.47%**	**34.67%**	**55.84%**	**66.06%**

Iteration-budget analysis. Under the same attack settings, we plot the micro-averaged ASR against the number of optimization iterations. The results are shown in Fig. 2, where the horizontal lines indicate the corresponding baseline ASR after 100 iterations. MissClick reaches or exceeds these reference results within 10–30 iterations in all four settings. On UGround, MissClick-U achieves 42.48% ASR after 10 iterations, compared with 42.21% for Representation attack after 100 iterations. This indicates that MissClick requires fewer optimization iterations to reach the corresponding baseline performance and achieves stronger attack performance under limited iteration budgets.



(a) Untargeted attacks. (b) Targeted attacks.

Figure 2: Attack success rate under different optimization iteration budgets. The horizontal lines indicate the 100-iteration ASR of the corresponding baselines. Solid and dashed curves denote OS-Atlas and UGround, respectively.

Qualitative examples. Figure 1 illustrates an executable targeted click redirection. Additional untargeted and targeted examples across both models and all three platforms are provided in Appendix F, together with perturbation visualizations under different budgets.

3.6 Cross-Model Transfer and Input Preprocessing

Cross-model transferability. We evaluate transfer between OS-Atlas and UGround without re-optimization. As reported in Appendix Table 13, MissClick-U obtains untargeted transfer ASRs of 17.01% from OS-Atlas to UGround and 38.56% in the reverse direction, compared with 14.16% and 54.72% for Representation attack. MissClick-T achieves targeted transfer ASRs of 3.37% and 8.71%, compared with 1.75% and 4.16% for Token CE. Overall, cross-model transfer remains substantially weaker than white-box performance and varies with transfer direction and attack objective.

Non-adaptive input preprocessing. We evaluate fixed JPEG compression and Gaussian blur without incorporating either transformation into attack optimization. Across both models, these transformations reduce MissClick-U ASR to 13.09%–20.92% and MissClick-T ASR to 2.25%–4.55%. They also reduce clean grounding accuracy by up to 4.95 percentage points. These results show that fixed preprocessing substantially suppresses MissClick but also affects clean grounding accuracy. Complete results are reported in Appendix Table 14.

4 Related Work

GUI visual grounding. GUI visual grounding maps natural-language instructions to screen regions (Cheng et al., 2024). Many methods generate point or bounding-box coordinates autoregressively (Cheng et al., 2024; Gou et al., 2025; Wu et al., 2025c). UGround (Gou et al., 2025) targets cross-platform generalization, while OS-Atlas (Wu et al., 2025c) uses a unified multi-platform action space. GUI-Actor (Wu et al., 2025b) instead uses a coordinate-free action head. MissClick targets coordinate-generating models, where digit tokens are parsed into executable clicks.

Adversarial perturbation attacks. Adversarial examples are well studied in image classification (Goodfellow et al., 2015; Carlini & Wagner, 2017; Croce & Hein, 2020). For GUI grounding, Zhao et al. (2025) optimize visual-embedding distance or target-coordinate probability. Related autoregressive attacks optimize token-sequence objectives (Carlini et al., 2023; Bailey et al., 2024). These objectives operate before numerical parsing and click mapping, whereas MissClick directly targets click-level outcomes. Our setting is complementary to prompt injection, backdoors, and broader agent attacks (Wang et al., 2025; Ye et al., 2025; Wu et al., 2025a; Zhang et al., 2025c; Debenedetti et al., 2024).

Numerical-aware objectives. Prior supervised objectives incorporate numerical structure during training. Number Token Loss models numerical distance (Zausinger et al., 2025), while Fei et al. (2025) combine Earth Mover's Distance with position weighting. DIST^{2}Loss uses token-distance-based soft targets (Chung et al., 2026), and Phi-Ground studies coordinate-digit smoothing and position weighting (Zhang et al., 2025b). Unlike these training objectives, MissClick keeps the victim model fixed and optimizes a bounded screenshot perturbation: MissClick-U maximizes soft-click displacement, whereas MissClick-T optimizes place-weighted target digits.

5 Conclusion, Limitations, and Future Work

This work identifies an execution-level attack surface at the interface between coordinate generation and click execution in GUI grounding models. MissClick aligns attack optimization with this interface through soft-coordinate displacement for untargeted disruption and place-weighted target-digit optimization for targeted redirection. Across two models and three platforms, MissClick-U improves untargeted ASR by 16.62 and 30.72 percentage points, while MissClick-T improves targeted ASR by 31.73 and 47.06 percentage points over the corresponding baselines. The objective comparison further shows that untargeted and targeted attacks favor different objectives. Our evaluation covers two models under a white-box threat model, both of which generate per-digit decimal coordinates on ScreenSpot-v2. MissClick-T also treats all incorrect values within each digit position uniformly. Cross-model transfer remains weaker than white-box performance, while fixed input preprocessing reduces both attack success and clean grounding accuracy. Future work will extend execution-aware attacks to broader output interfaces, ordinal-aware objectives, query-based settings, and adaptive defenses.

AI USE STATEMENT

Generative AI tools were used to assist with manuscript preparation, primarily for improving English expression, clarity, organization, and readability. The authors reviewed and edited all AI-assisted content and independently verified the technical claims, equations, citations, experimental results, and conclusions presented in the paper. Generative AI tools were not used to generate experimental data or replace the authors' technical judgment. The authors take full responsibility for the final content of the manuscript.

ETHICS STATEMENT

This work studies adversarial perturbations against GUI grounding models and therefore has potential dual-use implications. The proposed techniques are intended to identify vulnerabilities in coordinate-generating GUI grounding systems and to support the development of more robust models and execution safeguards. All experiments were conducted in an offline research setting using publicly available models and benchmark data. The study does not involve human participants, private user data, or interactions with deployed third-party systems. We acknowledge that adversarial techniques may potentially be misused, and we present the results for the purpose of security analysis and robustness research.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the threat model, generation-to-execution pipeline, attack objectives, and optimization procedure in Sec. 2. The experimental setup, including victim models, dataset, baselines, attack settings, evaluation metrics, and decoding procedure, is described in Sec. 3. Additional implementation details, model-specific coordinate formats, coordinate parsing procedures, evaluation protocols, and the complete attack algorithm are provided in the appendix. The appendix also reports additional experimental results and analyses to facilitate verification and reproduction of the reported findings.

REFERENCES

Luke Bailey, Euan Ong, Stuart Russell, and Scott Emmons. Image hijacks: adversarial images can control generative models at runtime. In *Proceedings of International Conference on Machine Learning (ICML)*, pp. 2443–2455, 2024.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy (IEEE S&P)*, pp. 39–57. IEEE, 2017.

Nicholas Carlini, Milad Nasr, Christopher A Choquette-Choo, Matthew Jagielski, Irena Gao, Pang Wei W Koh, Daphne Ippolito, Florian Tramer, and Ludwig Schmidt. Are aligned neural networks adversarially aligned? *Advances in Neural Information Processing Systems (NeurIPS)*, 36:61478–61500, 2023.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Li YanTao, Jianbing Zhang, and Zhiyong Wu. Seeclick: Harnessing gui grounding for advanced visual gui agents. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 9313–9332, 2024.

Jiwan Chung, Saejin Kim, Yongrae Jo, Jaewoo Park, Dongjun Min, and Youngjae Yu. Teaching metric distance to discrete autoregressive language models. In *International Conference on Learning Representations (ICLR)*, 2026.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 2206–2216. PMLR, 2020.

Edoardo Debenedetti, Jie Zhang, Mislav Balunovic, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for llm agents. *Advances in Neural Information Processing Systems (NeurIPS)*, 37:82895–82920, 2024.

Xiang Fei, Jinghui Lu, Qi Sun, Hao Feng, Yanjie Wang, Wei Shi, An-Lan Wang, Jingqun Tang, and Can Huang. Advancing sequential numerical prediction in autoregressive models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 562–574, 2025.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015.

Boyu Gou, Demi Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. Navigating the digital world as humans do: Universal visual grounding for gui agents. In *International Conference on Learning Representations (ICLR)*, volume 2025, pp. 30851–30883, 2025.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, et al. Cogagent: A visual language model for gui agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14281–14290, 2024.

Kevin Qinghong Lin, Linjie Li, Difei Gao, Zhengyuan Yang, Shiwei Wu, Zechen Bai, Stan Weixian Lei, Lijuan Wang, and Mike Zheng Shou. Showui: One vision-language-action model for gui visual agent. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19498–19508, 2025.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, et al. Ui-tars: Pioneering automated gui interaction with native agents. *arXiv preprint arXiv:2501.12326*, 2025.

Xilong Wang, John Bloch, Zedian Shao, Yuepeng Hu, Shuyan Zhou, and Neil Zhenqiang Gong. Webinject: Prompt injection attack to web agents. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 2010–2030, 2025.

Yuan Wang, Mingyu Li, and Haibo Chen. From imperative to declarative: Towards llm-friendly os interfaces for boosted computer-use agents. In *Proceedings of the European Conference on Computer Systems (EuroSys)*, pp. 296–310, 2026.

Chen Wu, Rishi Shah, Jing Yu Koh, Russ Salakhutdinov, Daniel Fried, and Aditi Raghunathan. Dissecting adversarial robustness of multimodal lm agents. In *International Conference on Learning Representations (ICLR)*, volume 2025, pp. 28362–28383, 2025a.

Qianhui Wu, Kanzhi Cheng, Rui Yang, Chaoyun Zhang, Jianwei Yang, Huiqiang Jiang, Jian Mu, Baolin Peng, Bo Qiao, Reuben Tan, et al. Gui-actor: Coordinate-free visual grounding for gui agents. *Advances in Neural Information Processing Systems (NeurIPS)*, 38:15101–15128, 2025b.

Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, et al. Os-atlas: Foundation action model for generalist gui agents. In *International Conference on Learning Representations (ICLR)*, volume 2025, pp. 5090–5108, 2025c.

Ziang Ye, Yang Zhang, Wentao Shi, Xiaoyu You, Fuli Feng, and Tat-Seng Chua. Visualtrap: A stealthy backdoor attack on gui agents via visual grounding manipulation. In *Conference on Language Modeling (COLM)*, 2025.

Jonas Zausinger, Lars Pennig, Anamarija Kozina, Sean Sdahl, Julian Sikora, Adrian Dendorfer, Timofey Kuznetsov, Mohamad Hagog, Nina Wiedemann, Kacper Chlodny, et al. Regress, don't guess: A regression-like loss on number tokens for language models. In *Proceedings of International Conference on Machine Learning (ICML)*, pp. 73995–74017. PMLR, 2025.

Chaoyun Zhang, He Huang, Chiming Ni, Jian Mu, Si Qin, Shilin He, Lu Wang, Fangkai Yang, Pu Zhao, Chao Du, et al. Ufo2: The desktop agentos. *Transactions on Machine Learning Research (TMLR)*, 2025a.

Miaosen Zhang, Ziqiang Xu, Jialiang Zhu, Qi Dai, Kai Qiu, Yifan Yang, Chong Luo, Tianyi Chen, Justin Wagle, Tim Franklin, et al. Phi-ground tech report: Advancing perception in gui grounding. *arXiv preprint arXiv:2507.23779*, 2025b.

Yitong Zhang, Ximo Li, Liyi Cai, and Jia Li. Environmental injection attacks against gui agents in realistic dynamic environments. *arXiv preprint arXiv:2509.11250*, 2025c.

Haoren Zhao, Tianyi Chen, and Zhen Wang. On the robustness of gui grounding models against image attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 1618–1623, 2025.

A Complete Attack Procedure and Implementation Details

This section provides the complete MissClick optimization procedure and the model-specific details used to instantiate its execution-aware objectives. Algorithm 1 shows how MissClick-U and MissClick-T share the same projected-gradient optimization procedure while using different reference coordinates and attack objectives. The remainder of this section describes the native input and output formats of the victim models, coordinate-digit alignment, objective implementations, and attack settings.

Algorithm 1 MissClick optimization with goal-specific execution-aware objectives

Require: GUI grounding model $\mathcal{M}$, screenshot $\mathbf{I}$, instruction q, reference coordinates c^{ref}, perturbation budget ε, step size α, number of iterations N, and attack mode $m \in \{\mathrm{U}, \mathrm{T}\}$

Ensure: Adversarial screenshot $\widetilde{\mathbf{I}}$

Set $c^{\mathrm{ref}} \leftarrow c^{\mathrm{gt}}$ if $m = \mathrm{U}$; otherwise set $c^{\mathrm{ref}} \leftarrow c^{\mathrm{tgt}}$
Construct a model-native reference answer sequence containing c^{ref} for teacher forcing
Extract coordinate-digit positions (k, j) and place values $a_{k,j}$ from the reference answer sequence
Initialize $\boldsymbol{\delta}$ randomly within $[-\varepsilon, \varepsilon]^d$
for $i = 1, \ldots, N$ **do**
    Run a teacher-forced forward pass on $(\mathbf{I} + \boldsymbol{\delta}, q)$ using the reference answer sequence and the frozen model $\mathcal{M}$
    Extract digit distributions $p_{k,j}(v)$ using equation 6
    if $m = \mathrm{U}$ **then**
        Compute soft coordinates $\widetilde{c}$ using equation 7 and equation 8
        $\mathcal{L} \leftarrow \mathcal{L}_{\mathrm{U}}$ using equation 9
        $\boldsymbol{\delta} \leftarrow \Pi_{[-\varepsilon,\varepsilon]^d}\left(\boldsymbol{\delta} + \alpha \operatorname{sign}\left(\nabla_{\boldsymbol{\delta}} \mathcal{L}\right)\right)$ {maximize click displacement}
    else if $m = \mathrm{T}$ **then**
        $\mathcal{L} \leftarrow \mathcal{L}_{\mathrm{T}}$ using equation 12
        $\boldsymbol{\delta} \leftarrow \Pi_{[-\varepsilon,\varepsilon]^d}\left(\boldsymbol{\delta} - \alpha \operatorname{sign}\left(\nabla_{\boldsymbol{\delta}} \mathcal{L}\right)\right)$ {minimize target-digit loss}
    end if
end for
return $\widetilde{\mathbf{I}} = \operatorname{clamp}\left(\mathbf{I} + \boldsymbol{\delta}, 0, 1\right)$

A.1 Complete Attack Procedure

MissClick keeps the victim model fixed and updates only the screenshot perturbation. At each iteration, a teacher-forced forward pass provides the coordinate-digit distributions used by the selected execution-aware objective. MissClick-U maximizes soft-click displacement from the ground-truth click, whereas MissClick-T minimizes the place-weighted loss of the target digits.

Teacher forcing is used only to obtain differentiable optimization signals. Attack success is determined from the coordinates produced by standard greedy autoregressive generation without the reference answer sequence.

A.2 Model-Specific Inputs and Coordinate Outputs

Prompts and Native Coordinate Formats. We evaluate OS-Atlas-Base-7B and UGround-V1-7B using their respective native grounding prompts and coordinate formats. For OS-Atlas, the user prompt is

> In this UI screenshot, what is the position of the element corresponding to the command ``*q*'' (with bbox)?

where q denotes the task instruction. OS-Atlas generates a bounding-box coordinate $c = (x_1, y_1, x_2, y_2)$ in the normalized $[0, 1000]$ coordinate system. Its native supervised answer contains an object-reference segment followed by a bounding-box segment of the form

> <|object_ref_start|>q<|object_ref_end|>
> <|box_start|>$(x_1, y_1), (x_2, y_2)$ <|box_end|>.

For UGround, the user prompt is

> Your task is to help the user identify the precise coordinates (x, y) of a specific area/element/object on the screen based on a description.
> Description: *q*
> Answer:

UGround directly generates a point coordinate $c = (x, y)$ in the same normalized coordinate system. Its native reference answer is represented as (x, y). The two native formats are converted to executed clicks using the mapping $r(\cdot)$ defined in the main paper. For an OS-Atlas bounding box, we use its center,

$$r(c) = \left(\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2} \right).$$

For a UGround point output, we directly use $r(c) = (x, y)$.

Reference Answers and Coordinate-Digit Alignment. During attack optimization, each input is paired with a model-native reference answer. For untargeted attacks, the reference answer contains the ground-truth coordinate output c^{gt}. For targeted attacks, the task instruction and screenshot remain unchanged, while the coordinate part of the reference answer is replaced with the attacker-specified target coordinate output c^{tgt}. For OS-Atlas, c^{gt} or c^{tgt} is represented as a bounding box enclosed by the native <|box_start|> and <|box_end|> markers. For UGround, the corresponding reference is represented as a point coordinate. The targeted UGround reference uses the model-native point associated with the center of the selected target region. The final targeted success criterion nevertheless uses the complete target region B_{tgt} rather than requiring exact equality with this point coordinate. The supervised answer positions are identified from the teacher-forced labels, where prompt and padding positions are masked with -100. We then locate the individual digit tokens within the coordinate span of the reference answer. For each coordinate component c_k, the j-th digit is associated with its token position $t_{k,j}$ and decimal place value $a_{k,j} = 10^{m_k - j}$, where m_k is the number of digits in c_k. We therefore record each coordinate digit as $(t_{k,j}, k, a_{k,j})$. Here, $t_{k,j}$ denotes its absolute position in the complete teacher-forced token sequence, k identifies the coordinate component, and $a_{k,j}$ specifies its numerical place value. Since the logits at position $t_{k,j} - 1$ predict the token at $t_{k,j}$, these indices are used to extract the corresponding next-token digit logits. For OS-Atlas, we identify the four coordinate components within the native bounding-box markers and assign them to x_1, y_1, x_2, and y_2 in order. For UGround, we identify the two coordinate components in its native point-format answer and assign them to x and y. We then identify the token position of each coordinate digit in the teacher-forced reference answer. Each position is associated with its coordinate component and decimal place value.

A.3 Attack Optimization Details

Coordinate-Digit Identification and Logit Alignment. A causal language model uses the logits at position $t_{k,j} - 1$ to predict the token at position $t_{k,j}$. Therefore, for each coordinate-digit position

Table 4: Core attack settings.

Item	Configuration
Victim models	OS-Atlas-Base-7B; UGround-V1-7B
Max pixels	1024
Perturbation budget	$16/255$
Step size	$1/255$
Iterations (optimized methods)	100

identified above, the implementation extracts the corresponding next-token logit vector. It then selects the logits associated with the decimal digits `0`, ..., `9` using the victim model's tokenizer. These digit-level logits are used to construct the probability distributions and attack objectives described below.

Digit Distributions and Soft-Coordinate Reconstruction. Let $z_{k,j,v}$ denote the next-token logit for digit v at position $t_{k,j} - 1$. A softmax over the ten digit logits produces the digit-restricted distribution $p_{k,j}(v)$:

$$p_{k,j}(v) = \frac{\exp(z_{k,j,v})}{\sum_{u=0}^{9} \exp(z_{k,j,u})}, \qquad v \in \{0, \ldots, 9\}. \tag{14}$$

The expected digit value is computed as

$$\widetilde{d}_{k,j} = \sum_{v=0}^{9} v \, p_{k,j}(v),$$

and each soft coordinate component is reconstructed by

$$\widetilde{c}_k = \sum_{j=1}^{m_k} a_{k,j} \widetilde{d}_{k,j}.$$

For OS-Atlas, the four reconstructed components form $\widetilde{c} = (\widetilde{x}_1, \widetilde{y}_1, \widetilde{x}_2, \widetilde{y}_2)$, and the soft click is the center of this soft box. For UGround, the two reconstructed components directly form the soft click $(\widetilde{x}, \widetilde{y})$. The reconstructed soft click is used by the soft-coordinate objective and is later compared with the final generated click. This computation uses only the digit-restricted distributions during optimization. During final evaluation, the model generates its response from the full vocabulary without digit-only or format-constrained decoding.

Baselines and Objective Variants. Except for Random noise, the compared methods use the same perturbation budget and optimization settings but differ in their loss functions.

Random noise uses a random perturbation clipped to the same pixel-space ℓ_∞ budget, without gradient updates.

Token CE applies standard cross-entropy to all supervised tokens in the model-native reference answer. The loss is maximized on the ground-truth answer for untargeted attacks and minimized on the target answer for targeted attacks.

Representation attack maximizes the mean-squared difference between the clean and adversarial visual embeddings. Because this objective does not specify a target coordinate or region, it is used only for untargeted attacks.

Digit CE applies cross-entropy only at the coordinate-digit positions and assigns equal weight to every digit. The digit loss is maximized for untargeted attacks and minimized for targeted attacks.

Soft-coordinate distance optimizes the squared distance between the reconstructed soft click and a reference click. For untargeted attacks, MissClick-U maximizes this distance from the ground-truth click. For targeted attacks, Soft-coordinate distance minimizes the distance to the center of the target region.

Place-weighted digit CE weights the cross-entropy at each coordinate-digit position by its decimal place value and normalizes the result by the total place weight. For untargeted attacks, it maximizes the weighted loss on the ground-truth digits. For targeted attacks, MissClick-T minimizes the weighted loss on the target digits.

Table 5: Numbers of evaluated samples after model-specific clean-correct filtering and, for targeted attacks, valid-target filtering.

Setting	Model	Desktop	Web	Mobile	All
Untargeted	OS-Atlas	273	362	436	1,071
Untargeted	UGround	295	372	456	1,123
Targeted	OS-Atlas	152	271	346	769
Targeted	UGround	161	279	361	801

Table 6: Parse validity of adversarial outputs on ScreenSpot-v2. Untargeted outputs are evaluated on model-specific clean-correct sets (1071 for OS-Atlas, 1123 for UGround). Targeted outputs are evaluated on model-specific clean-correct targeted sets (769 for OS-Atlas, 801 for UGround). Invalid outputs count as untargeted successes and targeted failures.

Setting	Model	Method	Outputs	Valid	Invalid	Valid rate
Untargeted	OS-Atlas	Random noise	1071	1071	0	100.00%
		Token CE	1071	1068	3	99.72%
		Representation attack	1071	1071	0	100.00%
		Digit CE	1071	1070	1	99.91%
		Place-weighted digit CE	1071	1070	1	99.91%
		MissClick-U	1071	1071	0	100.00%
	UGround	Random noise	1123	1123	0	100.00%
		Token CE	1123	1123	0	100.00%
		Representation attack	1123	1123	0	100.00%
		Digit CE	1123	1123	0	100.00%
		Place-weighted digit CE	1123	1123	0	100.00%
		MissClick-U	1123	1123	0	100.00%
Targeted	OS-Atlas	Token CE	769	768	1	99.87%
		Soft-coordinate distance	769	769	0	100.00%
		Digit CE	769	769	0	100.00%
		MissClick-T	769	769	0	100.00%
	UGround	Token CE	801	801	0	100.00%
		Soft-coordinate distance	801	801	0	100.00%
		Digit CE	801	801	0	100.00%
		MissClick-T	801	801	0	100.00%

Attack Settings. The core attack settings are summarized in Table 4. Within each model and attack setting, the compared objectives use the same screenshots, instructions, model-specific prompts, reference-answer construction, preprocessing, perturbation constraint, random seed, and evaluator. During optimization, the victim-model parameters are frozen and only the image perturbation is updated. Except for Random noise, all methods update the image perturbation using gradients and constrain it to the same ℓ_∞ budget after every step. The final saved images are evaluated using the same autoregressive generation, coordinate parser, click mapping, and success criteria. Reported ASR values are computed on the same model-specific clean-correct task sets.

A.4 Teacher-Forced Optimization and Greedy Autoregressive Evaluation

Teacher-Forced Optimization Path. Teacher forcing is used only to construct a differentiable optimization path. The input sequence contains the screenshot, task instruction, and model-native reference answer. The reference answer contains c^{gt} for an untargeted attack and c^{tgt} for a targeted attack. The victim model remains frozen. The forward pass yields differentiable next-token logits at the fixed reference-answer positions. MissClick extracts the logits at coordinate-digit positions, forms digit-restricted probability distributions, and computes either the soft-coordinate objective or the target-digit objective. Gradients are propagated from this objective to the screenshot perturbation. The reference answer, teacher-forced logits, soft coordinates, and monitored digit predictions are used only during attack optimization. They are not used to determine whether the final attack succeeds.

Table 7: Target area, source-to-target center distance, and source–target IoU in the common 896-task target-construction set. Areas and distances are computed from normalized $[0, 1]$ coordinates. Here, n is the number of tasks; Min and Max are the minimum and maximum; Q1, Median, and Q3 are the 25th, 50th, and 75th percentiles; and IQR equals Q3 minus Q1. Target-area values are reported as percentages of the screen area.

Platform	Quantity	n	Min	Q1	Median	Q3	IQR	Max
Desktop	Target area	178	0.068%	0.354%	0.729%	1.579%	1.225%	52.417%
	Center distance	178	0.022663	0.199514	0.391831	0.630893	0.431379	1.202555
	Source–target IoU	178	0	0	0	0	0	0
Web	Target area	323	0.018%	0.170%	0.385%	1.024%	0.854%	22.973%
	Center distance	323	0.031281	0.248606	0.404980	0.602149	0.353544	1.211590
	Source–target IoU	323	0	0	0	0	0	0
Mobile	Target area	395	0.084%	0.598%	1.399%	3.932%	3.334%	15.990%
	Center distance	395	0.069027	0.282592	0.438890	0.665074	0.382481	1.066478
	Source–target IoU	395	0	0	0	0	0	0
All	Target area	896	0.018%	0.353%	0.798%	2.173%	1.820%	52.417%
	Center distance	896	0.022663	0.254889	0.422449	0.634902	0.380014	1.211590
	Source–target IoU	896	0	0	0	0	0	0

Table 8: Targeted attack success by ScreenSpot-v2 element type. Each cell reports successes/tasks and ASR.

Objective	OS-Atlas text	OS-Atlas icon/widget	UGround text	UGround icon/widget
MissClick-T	183/469; 39.0%	162/300; 54.0%	294/481; 61.1%	208/320; 65.0%
Digit CE	142/469; 30.3%	129/300; 43.0%	235/481; 48.9%	171/320; 53.4%
Soft-coordinate distance	59/469; 12.6%	62/300; 20.7%	116/481; 24.1%	105/320; 32.8%
Token CE	31/469; 6.6%	70/300; 23.3%	53/481; 11.0%	72/320; 22.5%

Final Generation and Evaluation Path. After optimization, the final adversarial screenshot is saved and evaluated in a separate inference pass. The reference answer is not provided in this pass. The model receives only the adversarial screenshot and the original task instruction and performs greedy autoregressive generation over the complete vocabulary. The generated textual response is parsed using the model-specific coordinate parser, mapped to an executed click, and compared with B_{gt} or B_{tgt}. All ASR values in the main paper are computed from this path.

B Evaluation Protocol and Target Construction

This section specifies how evaluation samples are constructed and how freely generated textual outputs are converted into attack success labels. The same protocol is used for all methods under each attack setting.

B.1 Clean-Correct Filtering and ASR Computation

Evaluation Sample Construction. ScreenSpot-v2 contains $1{,}272$ tasks: 334 desktop, 437 web, and 501 mobile tasks. Each task contains a screenshot, a natural-language instruction, and a ground-truth element bounding box. We first run each victim model on the unperturbed screenshots using its native prompt and output format. A task is considered clean-correct if the freely generated response can be parsed and the resulting executed click falls inside the ground-truth region B_{gt}. Clean correctness is determined independently for OS-Atlas and UGround because the models can succeed on different subsets of tasks. The untargeted evaluation set for a victim model contains all of its clean-correct tasks. This gives $1{,}071$ samples for OS-Atlas and $1{,}123$ samples for UGround. For targeted attacks, we first construct a common set of 896 ScreenSpot-v2 tasks that contain at least one valid target element. For each victim model, the targeted evaluation set includes only tasks that have a valid target element and are clean-correct for that model. This gives 769 samples for OS-Atlas and

801 samples for UGround. The same model-specific clean-correct set is used for all attack methods compared under the corresponding setting.

Denominators and Micro-Averaging. For a model $\mathcal{M}$ and attack setting m, ASR is defined as

$$\mathrm{ASR} = \frac{\sum_{i \in \mathcal{E}_{\mathcal{M},m}} \mathbb{I}[\mathrm{Success}(i)]}{|\mathcal{E}_{\mathcal{M},m}|},$$

where $\mathcal{E}_{\mathcal{M},m}$ is the corresponding model- and setting-specific evaluation set defined above. For untargeted attacks, $\mathrm{Success}(i) = 1$ when the executed click falls outside B_{gt}. For targeted attacks, $\mathrm{Success}(i) = 1$ when the executed click falls inside B_{tgt}. The "All" result is a micro-average. We first sum successful attacks over desktop, web, and mobile and then divide by the total number of evaluated samples:

$$\mathrm{ASR}_{\mathrm{All}} = \frac{n_{\mathrm{succ}}^{\mathrm{desktop}} + n_{\mathrm{succ}}^{\mathrm{web}} + n_{\mathrm{succ}}^{\mathrm{mobile}}}{n^{\mathrm{desktop}} + n^{\mathrm{web}} + n^{\mathrm{mobile}}}.$$

It is not obtained by simply averaging the three platform-specific ASR values. Table 5 reports the corresponding sample counts by setting, model, and platform.

B.2 Coordinate Parsing and Executed-Click Evaluation

Parsing and Click Mapping. For OS-Atlas, the evaluator extracts four numerical values from the native bounding-box response and forms

$$c = (x_1, y_1, x_2, y_2).$$

The executed click is the center of the predicted box. For UGround, it extracts the point coordinate $c = (x, y)$ and directly uses this point as the executed click. Before checking whether a click falls inside a bounding box, the generated $[0, 1000]$ click coordinates are normalized to $[0, 1]$ to match the bounding boxes in the evaluation JSON files. A point on the boundary of an annotated region is considered inside the region. Thus, a click $\mathbf{u} = (u_x, u_y)$ lies in $B = (x_1, y_1, x_2, y_2)$ when

$$x_1 \le u_x \le x_2 \quad \text{and} \quad y_1 \le u_y \le y_2.$$

Invalid-Output Handling. An output is invalid when the formal evaluator cannot convert the generated response into the required numerical coordinate format. If the model output on the clean screenshot cannot be parsed, the task is excluded because it cannot be verified as clean-correct. In the untargeted setting, an unparseable response does not produce a valid click inside B_{gt} and is counted as an untargeted attack success. In the targeted setting, an unparseable response does not produce an executed click inside B_{tgt} and is counted as a targeted attack failure.

Table 6 reports parse validity for every evaluated model and attack objective. In the untargeted setting, 13,159 of 13,164 outputs are parseable, corresponding to a validity rate of 99.96%. The five invalid outputs all occur on OS-Atlas: Token CE produces three, Digit CE produces one, and Place-weighted digit CE produces one. Every untargeted output generated by UGround is parseable, and MissClick-U achieves a 100% valid rate on both victim models.

In the targeted setting, 6,279 of 6,280 outputs are parseable, corresponding to a validity rate of 99.98%. The only invalid targeted output is produced by OS-Atlas Token CE. MissClick-T achieves a 100% valid rate on both victim models. Consequently, every success reported for MissClick-U and MissClick-T corresponds to an executable parsed click rather than an output-format failure.

B.3 Target-Region Construction

Candidate Selection and Sampling. Target regions are constructed before applying model-specific clean-correct filtering. For each ScreenSpot-v2 task, we consider the annotated UI elements in the same screenshot as candidate targets. A valid candidate must correspond to an element distinct from the ground-truth element and must have zero intersection-over-union with B_{gt}. If at least one valid candidate exists, one candidate is selected uniformly at random. Tasks without a valid candidate are excluded from targeted evaluation. This procedure produces a common target-construction set of 896 tasks: 178 desktop, 323 web, and 395 mobile tasks. The selected source–target pair is fixed before attack generation and shared across victim models and targeted methods.

Table 9: Targeted ASR by fixed target-area quartile. Each cell reports successes/tasks and ASR. The two model panels use the same frozen quartile boundaries. G1 contains the smallest 25% of targets, while G4 contains the largest 25%.

OS-Atlas

Objective	G1	G2	G3	G4
MissClick-T	87/186; 46.8%	88/185; 47.6%	78/199; 39.2%	92/199; 46.2%
Digit CE	76/186; 40.9%	71/185; 38.4%	58/199; 29.1%	66/199; 33.2%
Soft-coordinate distance	34/186; 18.3%	35/185; 18.9%	29/199; 14.6%	23/199; 11.6%
Token CE	36/186; 19.4%	27/185; 14.6%	20/199; 10.1%	18/199; 9.0%

UGround

Objective	G1	G2	G3	G4
MissClick-T	115/193; 59.6%	123/202; 60.9%	127/201; 63.2%	137/205; 66.8%
Digit CE	95/193; 49.2%	104/202; 51.5%	105/201; 52.2%	102/205; 49.8%
Soft-coordinate distance	49/193; 25.4%	59/202; 29.2%	55/201; 27.4%	58/205; 28.3%
Token CE	34/193; 17.6%	39/202; 19.3%	25/201; 12.4%	27/205; 13.2%

Table 10: Targeted ASR by fixed source-to-target center-distance quartile. Each cell reports successes/tasks and ASR. The two model panels use the same frozen quartile boundaries. G1 contains the 25% of tasks with the smallest source-to-target center distances, while G4 contains the 25% with the largest distances.

OS-Atlas

Objective	G1	G2	G3	G4
MissClick-T	130/189; 68.8%	94/193; 48.7%	59/193; 30.6%	62/194; 32.0%
Digit CE	97/189; 51.3%	78/193; 40.4%	47/193; 24.4%	49/194; 25.3%
Soft-coordinate distance	51/189; 27.0%	28/193; 14.5%	17/193; 8.8%	25/194; 12.9%
Token CE	38/189; 20.1%	21/193; 10.9%	13/193; 6.7%	29/194; 14.9%

UGround

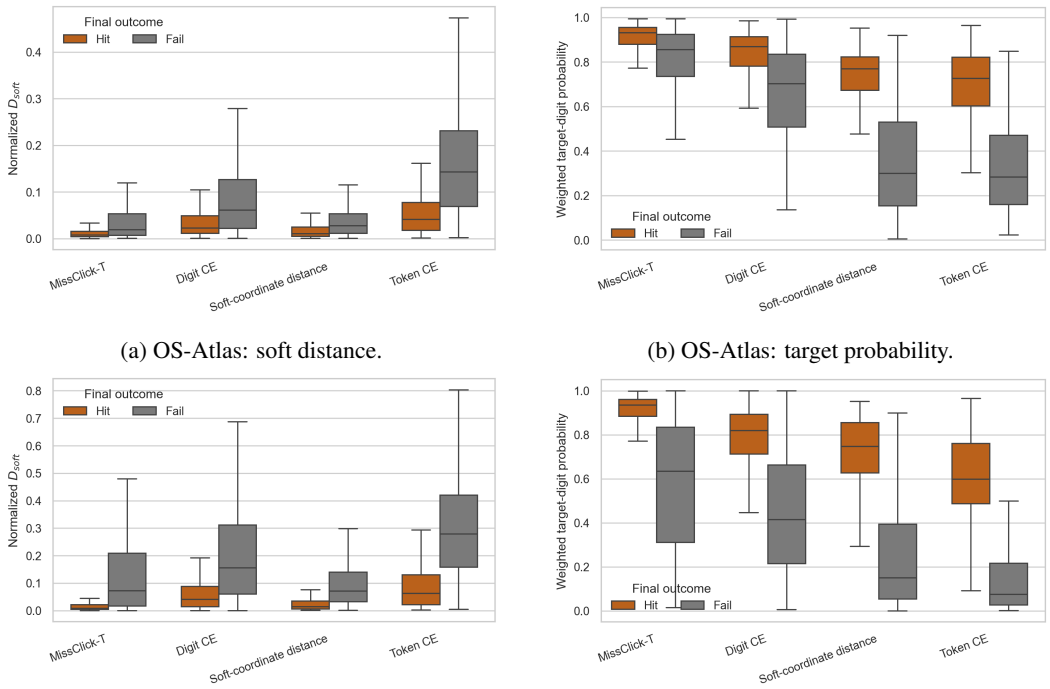
Objective	G1	G2	G3	G4
MissClick-T	155/200; 77.5%	133/200; 66.5%	110/201; 54.7%	104/200; 52.0%
Digit CE	125/200; 62.5%	105/200; 52.5%	89/201; 44.3%	87/200; 43.5%
Soft-coordinate distance	63/200; 31.5%	63/200; 31.5%	48/201; 23.9%	47/200; 23.5%
Token CE	30/200; 15.0%	34/200; 17.0%	24/201; 11.9%	37/200; 18.5%

Model-Specific Target References. The selected target element defines a common target region B_{tgt} in the normalized screen coordinate system. For OS-Atlas, the teacher-forced target reference contains the target bounding-box coordinates in the native OS-Atlas answer format. For UGround, the target reference contains the model-native point corresponding to the target-element center. These reference formats affect only the teacher-forced optimization sequence. For both victim models, targeted attack success is evaluated using the same complete target region: the final freely generated executed click must fall inside B_{tgt}. The attack is not required to exactly reproduce the reference coordinate string.

B.4 Target-Region Statistics

The geometry statistics are computed from the normalized $[0, 1]$ coordinates used in the evaluation JSON files. For each source–target pair, target area is the area of B_{tgt}, and source-to-target center distance is the Euclidean distance between the centers of B_{gt} and B_{tgt}.

Table 7 reports these quantities and source–target IoU for the common 896-task set, both overall and by platform. Target regions differ in size and in their distance from the original element. For all 896 tasks, the source and target bounding boxes have zero IoU.



(a) OS-Atlas: soft distance.

(b) OS-Atlas: target probability.

(c) UGround: soft distance.

(d) UGround: target probability.

Figure 3: Teacher-forced quantities stratified by final targeted outcome. Top row: OS-Atlas. Bottom row: UGround. Left column: normalized soft target distance D_{soft}. Right column: place-weighted target-digit probability $P_{\text{tgt}}^{\text{w}}$.

C Targeted Attack Results

C.1 Results by Element Type

ScreenSpot-v2 categorizes each task element as either text or icon/widget. Table 8 reports targeted ASR separately for these two element types. MissClick-T achieves the highest targeted ASR in all four model–element-type groups. Compared with the strongest alternative, Digit CE, MissClick-T improves ASR by 8.7 and 11.0 percentage points on OS-Atlas text and icon/widget tasks, and by 12.2 and 11.6 percentage points on the corresponding UGround tasks. These results show that the advantage of MissClick-T is observed for both element types and both victim models.

C.2 Target-Region Size and Distance Analysis

We analyze target size and source-to-target distance to determine whether the advantage of MissClick-T is limited to large or nearby targets. We define four target-area groups and four center-distance groups using the common 896-task set. We use quartiles to obtain four similarly sized groups without manually choosing area or distance thresholds. All models and targeted objectives use the same group boundaries. The target-area boundaries are 0.353%, 0.798%, and 2.173%, and the normalized center-distance boundaries are 0.254889, 0.422449, and 0.634902.

Tables 9 and 10 report the number of tasks and targeted ASR in each group. MissClick-T achieves the highest targeted ASR in every target-area and center-distance group for both models. Targeted ASR generally decreases as the source-to-target distance increases, but MissClick-T remains the strongest method even in the farthest group, exceeding Digit CE by 6.7 percentage points on OS-Atlas and 8.5 percentage points on UGround. The target-area patterns differ between the two models and do not show a common monotonic trend. Overall, the advantage of MissClick-T is not limited to large or nearby targets.

D Analysis of Targeted Optimization and Generation

MissClick optimizes a teacher-forced objective but evaluates on freely generated outputs. This section examines whether the teacher-forced optimization signals are consistent with the final targeted outcomes, and how place-value weighting changes digit-level target alignment relative to uniform digit weighting.

D.1 Optimization Signal and Final Outcome

For each targeted sample, we record two quantities from the teacher-forced forward pass after optimization: the normalized soft target distance D_{soft} (how close the soft-decoded coordinate is to the target center) and the place-weighted target-digit probability $P^{\mathrm{w}}_{\mathrm{tgt}}$ (how confidently the model assigns probability to the correct target digits, weighted by decimal place value). We then run greedy autoregressive generation on the same adversarial image and record the normalized hard distance D_{hard} and whether the executed click falls inside the target region (Hit or Fail).

Table 11 reports these quantities stratified by final outcome. Figure 3 shows the corresponding distributions. Across all four objectives and both models, Hit samples have lower D_{soft} and higher $P^{\mathrm{w}}_{\mathrm{tgt}}$ than Fail samples. For MissClick-T, Hit samples have a median D_{soft} of 0.008 on both models, compared to 0.019 and 0.072 for Fail samples on OS-Atlas and UGround, respectively. The corresponding median $P^{\mathrm{w}}_{\mathrm{tgt}}$ is 0.931 for Hit and 0.856 for Fail samples on OS-Atlas, and 0.935 for Hit and 0.634 for Fail samples on UGround.

Among Fail samples, Soft-coordinate distance has a relatively small D_{soft} but substantially lower $P^{\mathrm{w}}_{\mathrm{tgt}}$ and R_{argmax} than MissClick-T and Digit CE. This pattern suggests that a small teacher-forced soft-coordinate distance can coexist with weak target-digit alignment and does not ensure successful final autoregressive generation.

Figure 4 plots D_{soft} against D_{hard} for each sample, colored by final outcome. Hit samples cluster in the lower-left corner (small D_{soft}, small D_{hard}), while Fail samples are spread toward higher values on both axes. This pattern is consistent across models and objectives: samples with smaller soft target distance after optimization tend to have smaller hard distance after generation. The single invalid OS-Atlas Token CE output (Table 6) remains in the Fail count but is excluded from the scatter plot because it does not have a parsed final coordinate.

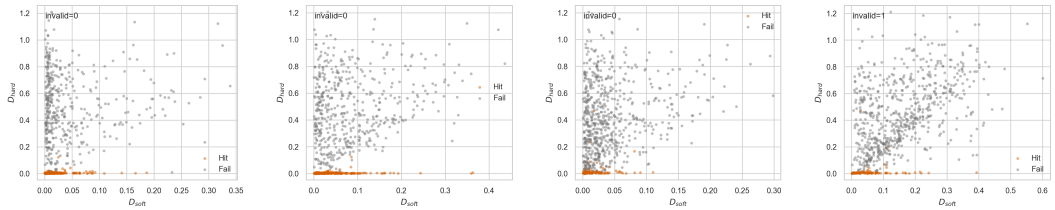
These results indicate that the teacher-forced optimization signal and the final greedy-generated outcome maintain a consistent relationship. They do not establish that a successful optimization signal guarantees a successful final outcome, as the final result is determined solely by the greedy-generated click.

D.2 Effect of Place-Value Weighting on Digit Positions

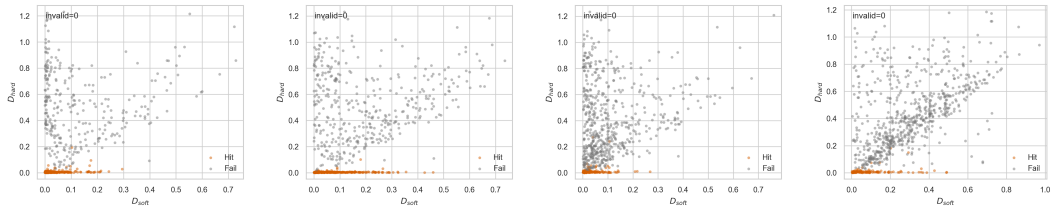
MissClick-T weights the cross-entropy at each digit position by its decimal place value, while Digit CE assigns equal weight to all positions. Table 12 compares the per-position results of the two objectives, grouped by place value (hundreds, tens, ones).

On both models, MissClick-T achieves higher mean target probability and argmax accuracy at the hundreds and tens positions than Digit CE. On OS-Atlas, the hundreds-place target probability is 0.922 for MissClick-T versus 0.818 for Digit CE; the tens-place values are 0.730 versus 0.671. UGround shows the same pattern: 0.851 versus 0.691 at the hundreds place, 0.710 versus 0.612 at the tens place.

At the ones place, the pattern reverses: MissClick-T has lower target probability than Digit CE on both models (0.184 vs. 0.364 on OS-Atlas; 0.302 vs. 0.528 on UGround). This trade-off is consistent with the method design. A change in the hundreds digit shifts the coordinate by approximately 100 coordinate units in the $[0, 1000]$ system, whereas a change in the ones digit shifts it by approximately 1 coordinate unit. Place-value weighting therefore gives greater priority to digits with a larger numerical effect on the generated coordinate. The observed redistribution toward the hundreds and tens positions is consistent with the higher targeted ASR of MissClick-T over Digit CE in the main results. Figure 5 visualizes this comparison. MissClick-T achieves higher target probability and



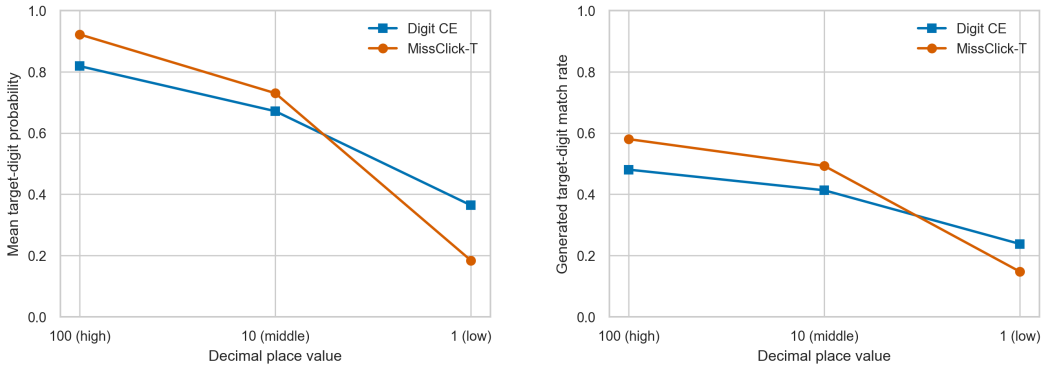
(a) OS-Atlas, MissClick-T. (b) OS-Atlas, Digit CE. (c) OS-Atlas, soft-coordinate distance. (d) OS-Atlas, Token CE.



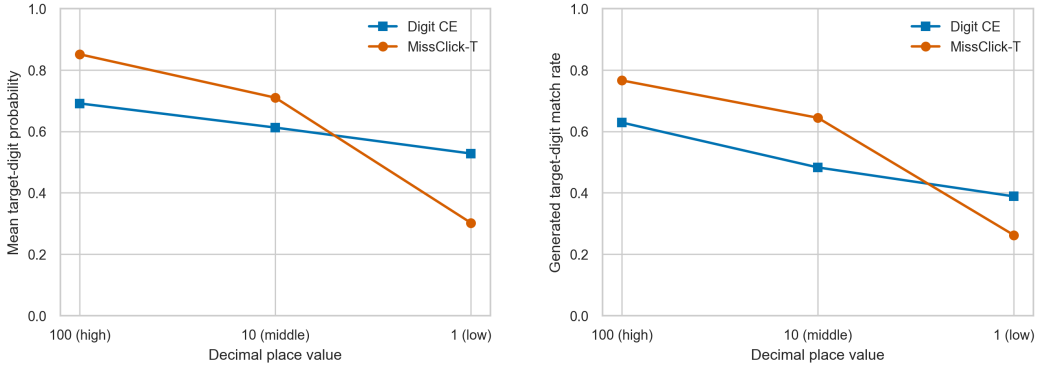
(e) UGround, MissClick-T. (f) UGround, Digit CE. (g) UGround, soft-coordinate distance. (h) UGround, Token CE.

Figure 4: Soft target distance D_{soft} versus final greedy-generated target distance D_{hard}, colored by final outcome (Hit or Fail). Each panel shows one model–objective combination.

match rate at the hundreds and tens positions, while Digit CE is higher at the ones position. This pattern is consistent across both models and both metrics.



(a) OS-Atlas (b) OS-Atlas



(c) UGround (d) UGround

Figure 5: Per-position comparison of uniform Digit CE and place-weighted MissClick-T. Left column: mean target-digit probability. Right column: generated target-digit match rate. Top row: OS-Atlas. Bottom row: UGround.

Table 11: Hit/Fail diagnostic. Values are median (IQR); n is the final-outcome task count, n_d is the number with a valid teacher-forced diagnostic, and n_h is the number with a valid final hard distance.

Model	Objective	Outcome	n	n_d	D_{soft}	$P^{\text{w}}_{\text{tgt}}$	R_{argmax}	D_{hard} / D_{region} (n_h)
OS-Atlas	MissClick-T	Hit	345	345	0.008 (0.012)	0.931 (0.075)	0.700 (0.114)	0.003 (0.003) / 0.000 (0.000) (345)
		Fail	424	424	0.019 (0.046)	0.856 (0.189)	0.636 (0.182)	0.494 (0.362) / 0.417 (0.368) (424)
	Digit CE	Hit	271	271	0.023 (0.038)	0.869 (0.133)	0.818 (0.217)	0.002 (0.003) / 0.000 (0.000) (271)
		Fail	498	498	0.061 (0.105)	0.702 (0.327)	0.727 (0.230)	0.473 (0.383) / 0.385 (0.360) (498)
	Soft-coordinate distance	Hit	121	121	0.011 (0.020)	0.770 (0.151)	0.500 (0.167)	0.007 (0.010) / 0.000 (0.000) (121)
		Fail	648	648	0.028 (0.042)	0.300 (0.376)	0.333 (0.194)	0.396 (0.372) / 0.318 (0.375) (648)
	Token CE	Hit	101	101	0.041 (0.059)	0.726 (0.218)	0.600 (0.167)	0.005 (0.004) / 0.000 (0.000) (101)
		Fail	668	668	0.143 (0.162)	0.283 (0.312)	0.455 (0.220)	0.430 (0.386) / 0.353 (0.378) (667)
UGround	MissClick-T	Hit	502	502	0.008 (0.017)	0.935 (0.076)	0.800 (0.167)	0.004 (0.004) / 0.000 (0.000) (502)
		Fail	299	299	0.072 (0.192)	0.634 (0.523)	0.600 (0.167)	0.487 (0.416) / 0.424 (0.410) (299)
	Digit CE	Hit	406	406	0.041 (0.074)	0.819 (0.179)	0.833 (0.200)	0.004 (0.002) / 0.000 (0.000) (406)
		Fail	395	395	0.155 (0.251)	0.414 (0.448)	0.667 (0.333)	0.502 (0.411) / 0.420 (0.419) (395)
	Soft-coordinate distance	Hit	221	221	0.014 (0.029)	0.747 (0.228)	0.667 (0.167)	0.006 (0.006) / 0.000 (0.000) (221)
		Fail	580	580	0.071 (0.108)	0.150 (0.340)	0.333 (0.300)	0.358 (0.405) / 0.292 (0.405) (580)
	Token CE	Hit	125	125	0.062 (0.109)	0.598 (0.274)	0.667 (0.200)	0.005 (0.007) / 0.000 (0.000) (125)
		Fail	676	676	0.279 (0.262)	0.075 (0.189)	0.500 (0.267)	0.406 (0.366) / 0.322 (0.363) (676)

E Transferability and Input-Preprocessing Robustness

E.1 Cross-Model Transferability

To evaluate cross-model transferability, we use the adversarial images generated against one victim model and evaluate them on the other model using the same evaluation protocol. No re-optimization is performed on the target model.

Table 13 reports untargeted and targeted transfer ASR in both directions. In the untargeted setting, transfer from UGround to OS-Atlas is substantially higher than from OS-Atlas to UGround for both methods. MissClick-U achieves higher untargeted transfer than Representation attack in the OS-Atlas $\rightarrow$ UGround direction (17.01% vs. 14.16%), while the reverse direction favors Representation attack (54.72% vs. 38.56%). The transfer results therefore vary with both the transfer direction and the attack objective.

In the targeted setting, transfer rates are low for both methods and both directions. MissClick-T nevertheless achieves roughly twice the targeted transfer rate of Token CE in both directions (3.37% vs. 1.75% for OS-Atlas $\rightarrow$ UGround; 8.71% vs. 4.16% for UGround $\rightarrow$ OS-Atlas).

Overall, cross-model transfer is substantially weaker than white-box attack performance and varies by transfer direction and attack objective.

E.2 Robustness to Input Preprocessing

To examine whether simple non-adaptive input preprocessing reduces attack effectiveness, we apply JPEG compression (quality$=75$) and Gaussian blur (σ=1.0, kernel size$=3$) to both clean and adversarial images before evaluation. These defenses are non-adaptive: the attack is not aware of the preprocessing at optimization time.

Table 14 reports clean accuracy and the ASR of MissClick-U and MissClick-T under each preprocessing. Both defenses reduce clean accuracy. JPEG-75 lowers OS-Atlas clean accuracy from 84.20% to 79.25% and UGround from 88.29% to 87.50%; Gaussian blur lowers them to 80.82% and 87.66%, respectively.

Both defenses also reduce MissClick ASR substantially. Untargeted ASR drops from 75.07% to below 21% on OS-Atlas and from 72.93% to below 17% on UGround. Targeted ASR is reduced even more sharply, falling below 5% for both models under either defense.

These results apply only to the evaluated non-adaptive setting and do not establish robustness against an adaptive attack optimized through the preprocessing operation.

Table 12: Coordinate-digit results for uniform Digit CE and place-weighted MissClick-T. Each cell reports mean target probability, target-digit argmax accuracy, and generated target-digit match rate, in that order.

Model	Objective	100 (high)	10 (middle)	1 (low)
OS-Atlas	Digit CE	0.818 / 0.897 / 0.480	0.671 / 0.808 / 0.413	0.364 / 0.544 / 0.238
	MissClick-T	0.922 / 0.961 / 0.580	0.730 / 0.850 / 0.493	0.184 / 0.235 / 0.147
UGround	Digit CE	0.691 / 0.789 / 0.629	0.612 / 0.709 / 0.483	0.528 / 0.778 / 0.388
	MissClick-T	0.851 / 0.900 / 0.766	0.710 / 0.816 / 0.644	0.302 / 0.426 / 0.262

Table 13: Cross-model transfer attack success on ScreenSpot-v2 (ε=$16/255$). Each reported rate is a micro-averaged ASR. Untargeted and targeted denominators are the target model's corresponding clean-correct sets pooled over desktop, web, and mobile.

Source → Target	Untargeted source attack	U-ASR	Targeted source attack	T-ASR
OS-Atlas → UGround	Representation attack	14.16%	Token CE	1.75%
	MissClick-U	17.01%	MissClick-T	3.37%
UGround → OS-Atlas	Representation attack	54.72%	Token CE	4.16%
	MissClick-U	38.56%	MissClick-T	8.71%

F Attack Visualization

This section shows successful attack examples evaluated using the same success criteria as the quantitative experiments. The displayed clean and adversarial clicks are obtained from the model outputs on the corresponding images.

F.1 Untargeted Attack Examples

Figures 6 and 7 show successful MissClick-U examples from desktop, web, and mobile tasks. In each case, the clean click lies inside the ground-truth region, whereas the adversarial click lies outside it.

F.2 Targeted Attack Examples

Figures 8 and 9 show successful MissClick-T examples. The final click lies inside the selected target region in every displayed case.

F.3 Perturbation Visibility across Budgets

Figures 10, 11, and 12 compare the same two samples at $\varepsilon \in \{4/255, 8/255, 16/255\}$. Both examples are successful targeted attacks at all three perturbation budgets. The figures visually compare the corresponding adversarial images and perturbations, but they do not quantitatively evaluate whether the perturbations are imperceptible.

Table 14: Effect of non-adaptive input preprocessing on clean accuracy and MissClick attack success (ε=16/255). Each cell reports the corresponding accuracy or attack success rate. Clean accuracy is evaluated on preprocessed original screenshots from the full ScreenSpot-v2 set, whereas U-ASR and T-ASR are evaluated on preprocessed adversarial screenshots generated from the frozen clean-correct task sets. All entries are micro-averages over desktop, web, and mobile.

Defense	OS Clean	OS U-ASR	OS T-ASR	UG Clean	UG U-ASR	UG T-ASR
None	84.20%	75.07%	44.86%	88.29%	72.93%	62.67%
JPEG-75	79.25%	19.61%	4.55%	87.50%	16.83%	3.75%
Gaussian blur	80.82%	20.92%	3.12%	87.66%	13.09%	2.25%

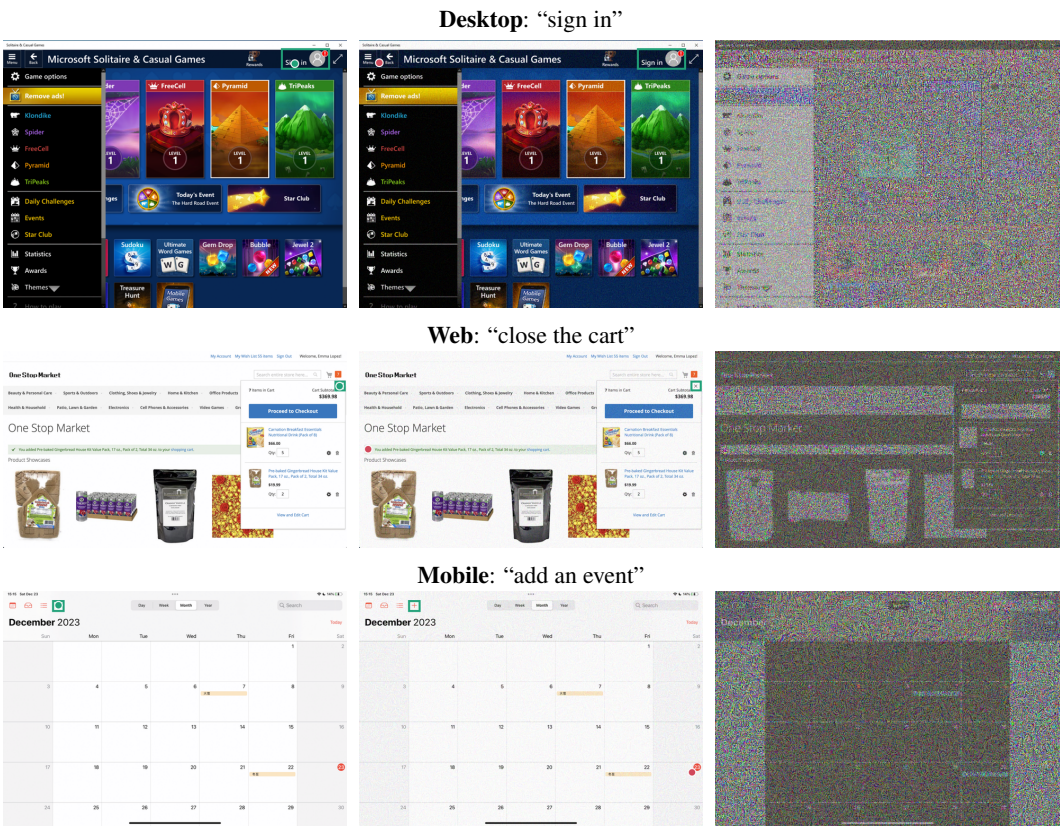
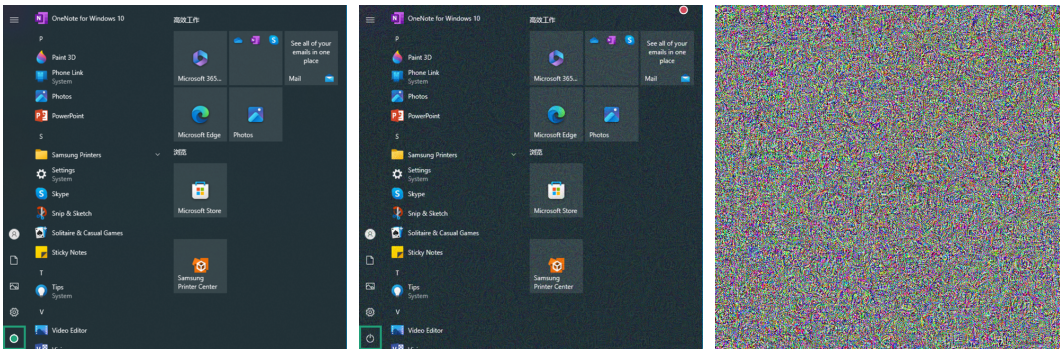
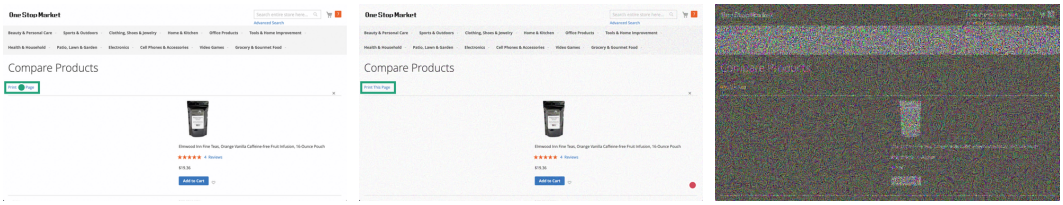


Figure 6: Successful OS-Atlas MissClick-U examples at $\varepsilon = 16/255$. Each row shows the processed clean screenshot, the final saved adversarial screenshot, and the signed perturbation visualized as $\mathrm{clip}(0.5 + 10\delta, 0, 1)$. Green boxes and points denote the ground-truth region and clean click; red points denote the final adversarial click.

Desktop: “turn off the computer”



Web: “print the page”



Mobile: “check settings”

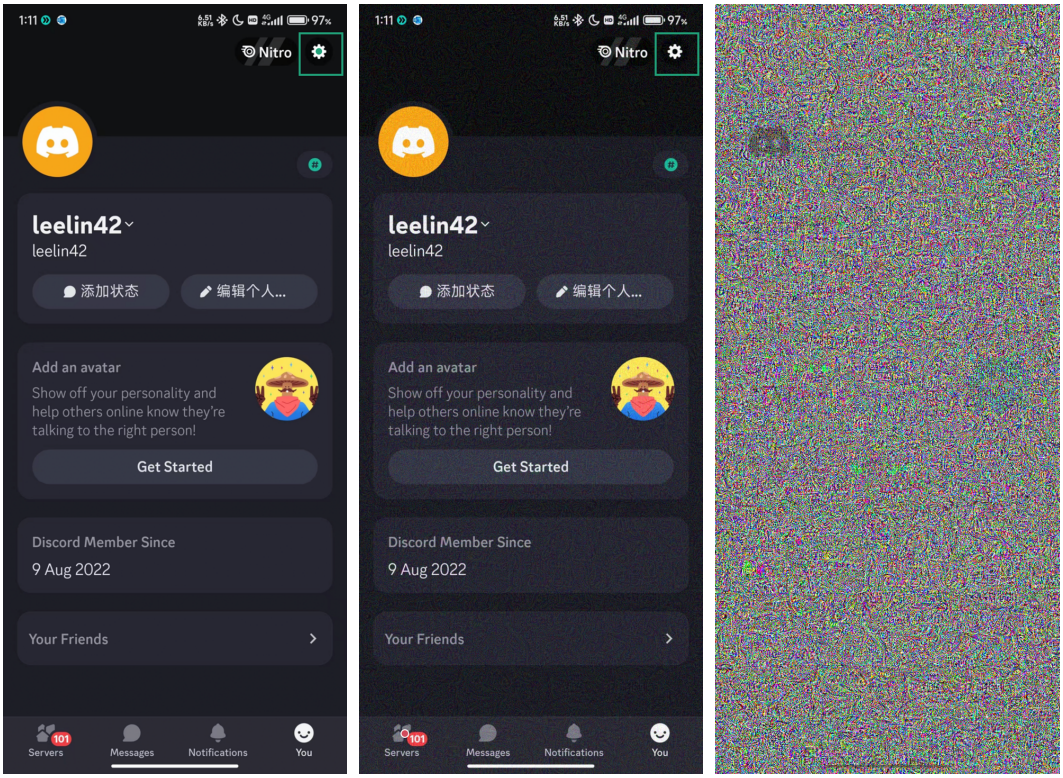
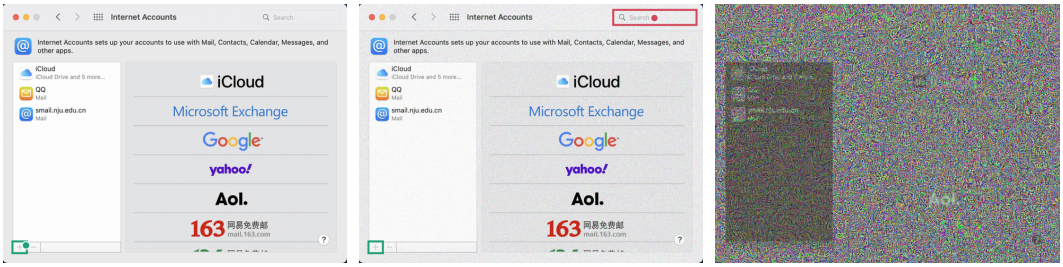
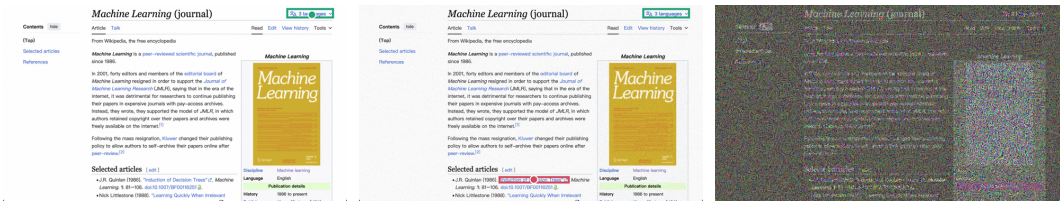


Figure 7: Successful UGround MissClick-U examples at $\varepsilon = 16/255$. The layout and annotation rules match Figure 6.

Desktop: “add a new account”



Web: “change page language”

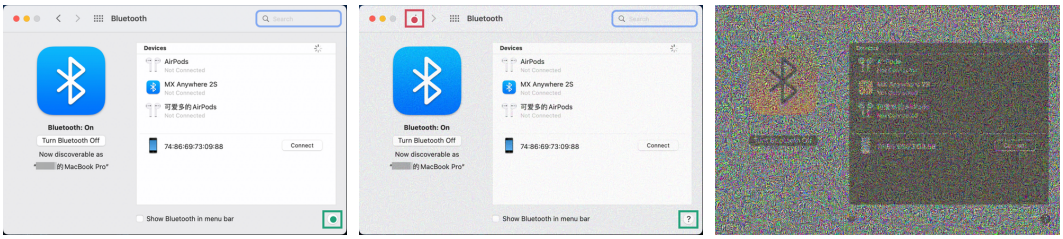


Mobile: “add a new app to the desktop”

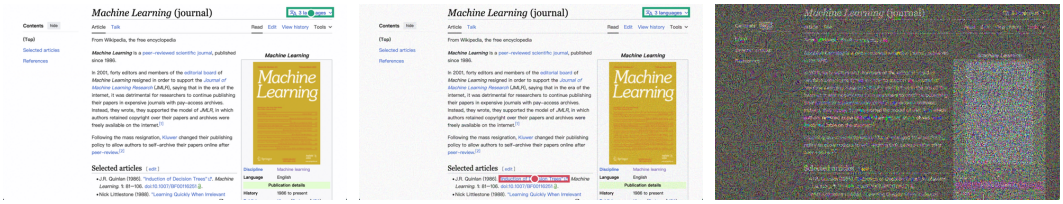


Figure 8: Successful OS-Atlas MissClick-T examples at $\varepsilon = 16/255$. Green boxes and points denote the ground-truth region and clean click. Red boxes denote the attacker-selected target region, and red points denote the final adversarial click. The third column shows the signed perturbation at a fixed $10\times$ scale.

Desktop: “question about bluetooth”



Web: “change page language”



Mobile: “add new city”

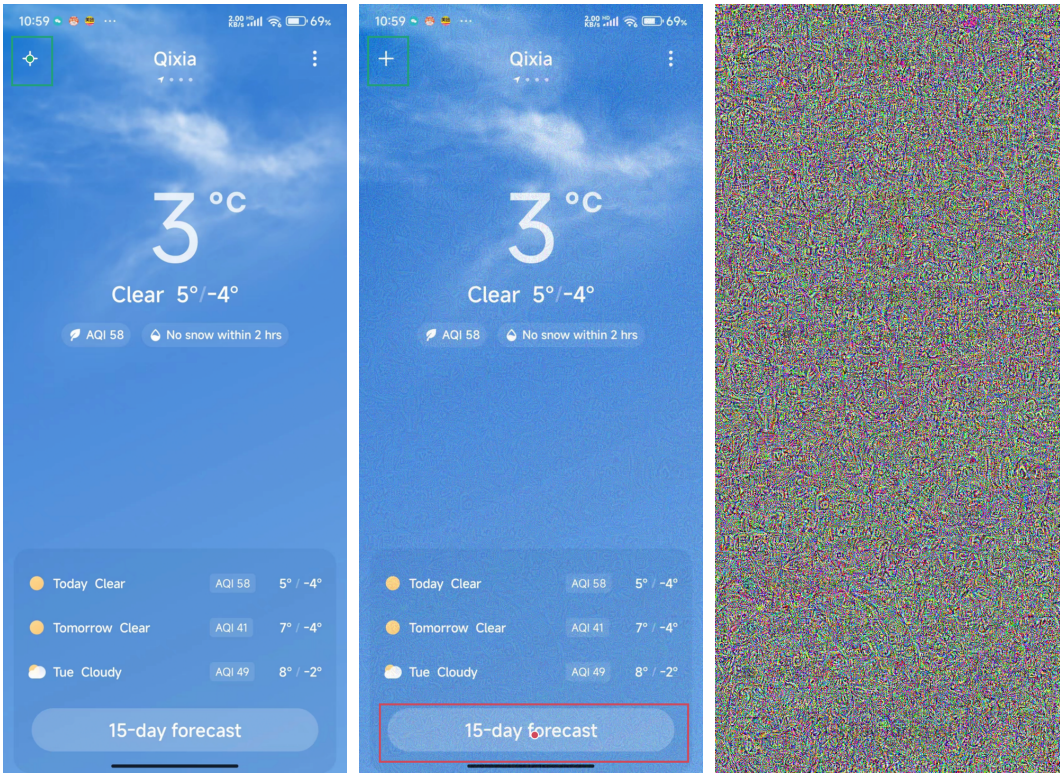
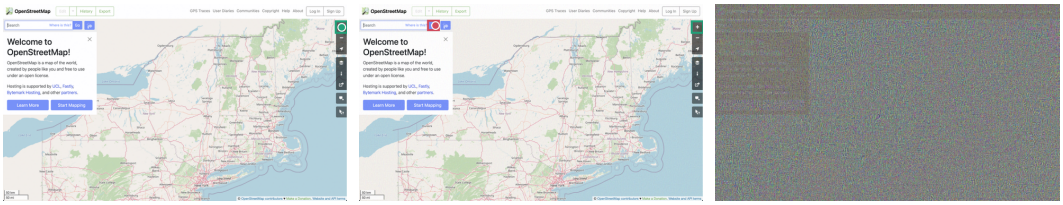


Figure 9: Successful UGround MissClick-T examples at $\varepsilon = 16/255$. The layout and annotation rules match Figure 8.

OS-Atlas, Web: “zoom in on the map”



UGround, Mobile: “add new city”

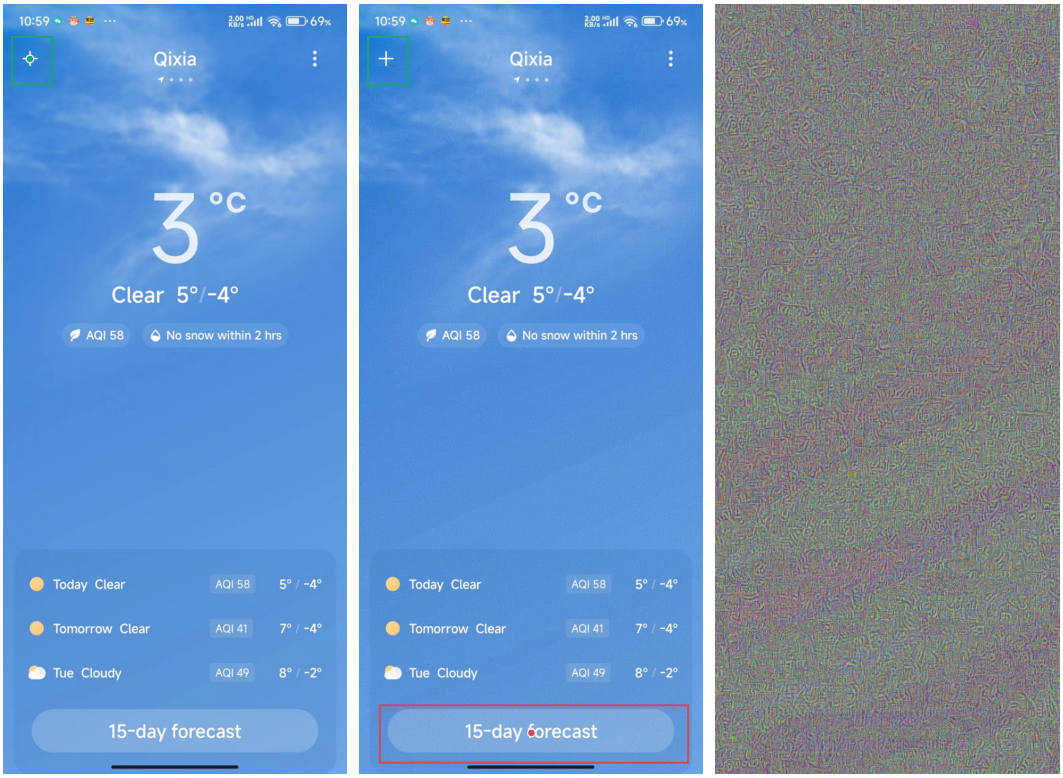
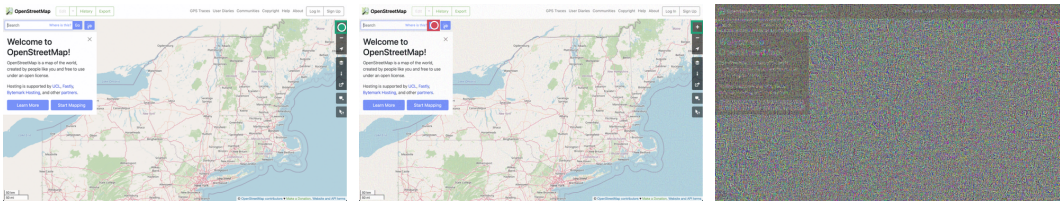


Figure 10: Same-sample comparison at $\varepsilon = 4/255$. The perturbation panels use the signed visualization rule $\mathrm{clip}(0.5 + 10\delta, 0, 1)$, where δ is computed by subtracting the processed clean image from the corresponding final saved adversarial image. No image is resized when computing the perturbation.

OS-Atlas, Web: “zoom in on the map”



UGround, Mobile: “add new city”

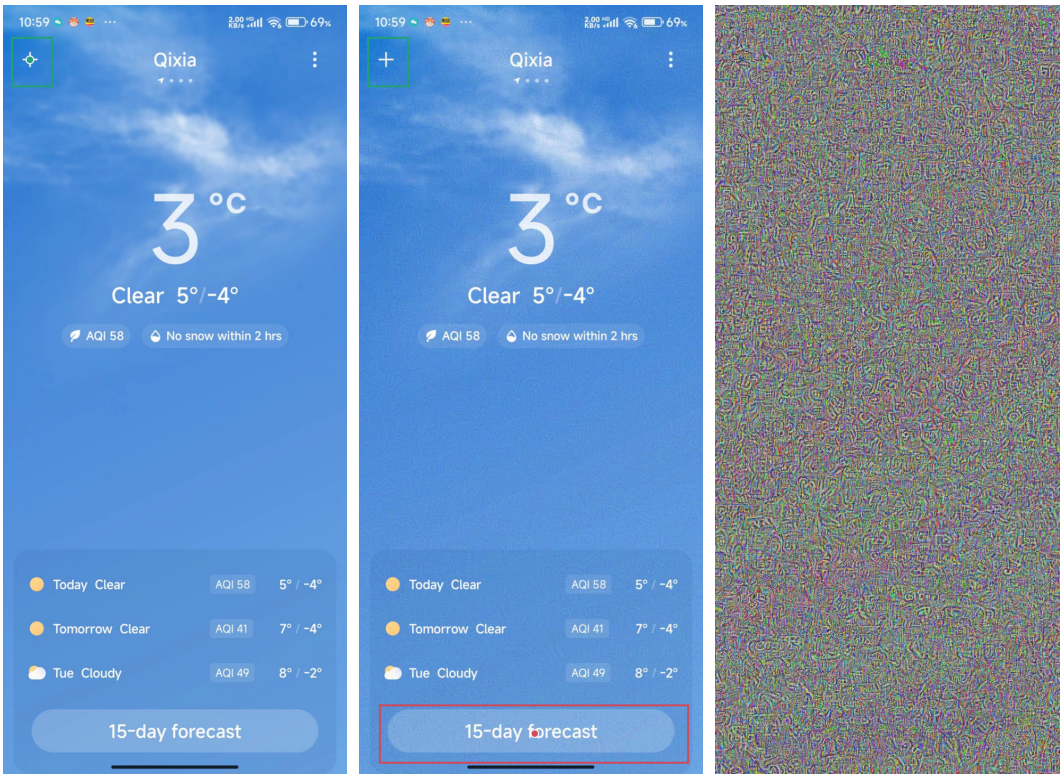
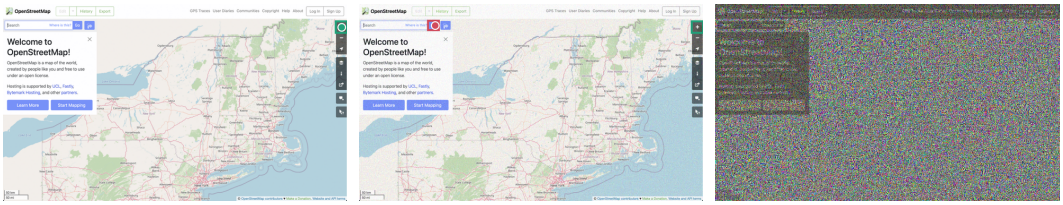


Figure 11: Same-sample comparison at $\varepsilon = 8/255$. The visualization rule is identical to Figure 10.

OS-Atlas, Web: “zoom in on the map”



UGround, Mobile: “add new city”

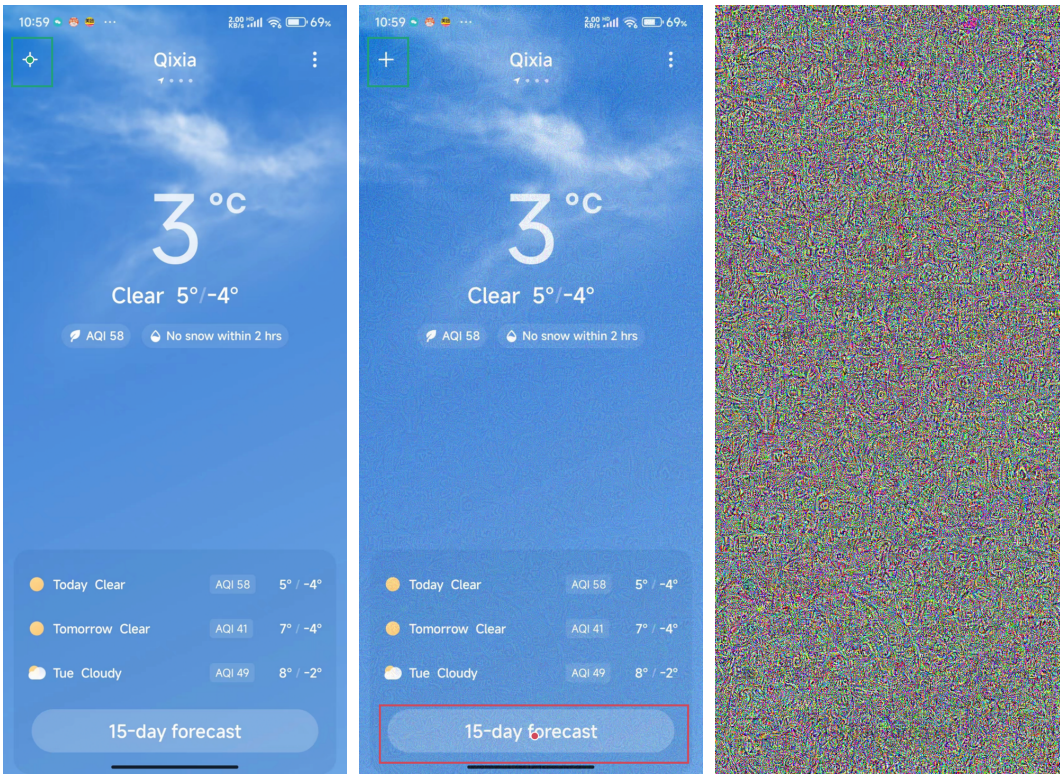


Figure 12: Same-sample comparison at $\varepsilon = 16/255$. The visualization rule is identical to Figure 10.