Virtus: A Robust CAPTCHA Based on the Human-AI Temporal Perception Gap

Antonio B De Castro, Astraea AI

Abstract

Mainstream CAPTCHA systems have been rendered obsolete by the emergence of powerful, agentic Vision-Language Models (VLMs). These models can now systematically defeat not only classic perception-based puzzles but also newer, reasoning-based challenges, achieving solve rates upwards of 70% on unseen CAPTCHAs. This paper presents **Virtus**, a novel anti-AI CAPTCHA system from Astraea AI that shifts the defense from a battle of static perception to a new domain: the human-AI cognitive gap in temporal perception. Virtus generates challenges by hiding a simple geometric shape within a dynamic, generative noise field. The "shape" is not a static pattern but a region of "temporal coherence" that humans innately perceive via the Gestalt principle of "Common Fate," while AI models analyzing individual frames see only random noise. We evaluated Virtus against a suite of SOTA VLMs, including Gemini 2.5 Pro and Claude 4.5 Sonnet, which achieved a **0% solve rate**. In a user study with 52 participants, human users achieved a **100% success rate** with a mean solve time of 4.9 seconds, with all participants rating the task as "easy." Virtus demonstrates a practical, robust, and scientifically-grounded defense that re-establishes a secure "human-easy, bot-hard" paradigm.

1. Introduction

Mainstream CAPTCHA systems, which form a critical layer of the web's defense against automated bots, are facing an existential crisis. The foundational security assumption for dominant platforms like reCAPTCHA v2, hCaptcha, and GeeTest—that their visual challenges are "bot-hard but human-friendly"—has been definitively invalidated. This failure is driven by the rapid emergence of generalized, agentic Vision-Language Models (VLMs) that, unlike their predecessors, do not require training on a specific CAPTCHA type.

The most prominent example of this new threat is the "Halligan" solver, a generalized visual CAPTCHA solver accepted to the 34th USENIX Security Symposium (2025). "Halligan" operates as a "CAPTCHA-to-action" system, treating the challenge as a generalized search problem. It ingests the natural language instruction, parses the visual interface into a search space, and iteratively explores actions to find a solution. In a 30-day "in-the-wild" test, "Halligan" achieved an average solving rate of **70.6%** on previously unseen CAPTCHAs. This single development breaks the traditional security model, as creating a new puzzle type is no longer an effective defense.

This attack on perception-based puzzles is complemented by a second attack on "reasoning-based" CAPTCHAs, which were briefly considered a more robust defense. Recent 2025 literature, benchmarked on new frameworks like "CAPTCHA-X," proves this defense is also obsolete. The key finding is that for modern VLMs, reasoning is not a barrier but a *vulnerability*; incorporating step-by-step reasoning is "the key to solving CAPTCHAS". Agentic VLM frameworks built on SOTA models like GPT-4o and Gemini 2.5 Pro now achieve **83.9%** accuracy on high-difficulty reasoning categories. With perceptual puzzles broken by generalized search agents and reasoning puzzles broken by agentic reasoning pipelines, the entire mainstream CAPTCHA landscape is compromised.

With all current avenues of defense compromised, the field requires a paradigm shift. The defense must be moved to a domain where AI models have a fundamental and provable disadvantage. This domain is the "cognitive gap"—the architectural difference between human holistic perception and AI's feature-based processing.

In this paper, we present **Virtus**, a novel anti-AI CAPTCHA system from Astraea AI that is the first to weaponize a specific temporal "cognitive gap." Virtus generates challenges by hiding simple geometric shapes within a dynamic, generative noise field. The "shape" is not a static image, but a "window" of temporal coherence. The system exploits the human Gestalt principle of **"Common Fate"**, allowing users to perceive the holistic shape while automatically filtering out the pixel-level static. As we will demonstrate, AI models analyzing individual frames see only random noise, creating a robust and impassable defense.

Our contributions are as follows:

1. We design and implement **Virtus**, a novel CAPTCHA system that leverages a "temporal coherence" algorithm to exploit the human-AI cognitive gap.
2. We conduct a large-scale **security evaluation** against a suite of SOTA VLMs, demonstrating a 0% solve rate from models that do not resort to random guessing.
3. We conduct a **usability study** with 52 participants, demonstrating a 100% human success rate and confirming the system is both fast and "human-friendly."

2. Background & Related Work

The failure of traditional CAPTCHAs has spurred research into new defense paradigms. Our work, Virtus, builds upon the concept of "adversarial CAPTCHAs" but is architecturally distinct from prior art. We situate our contribution by differentiating it from the two most relevant systems: "aCAPTCHA" and the recent "IllusionCAPTCHA."

"aCAPTCHA" (Adversarial CAPTCHAs). The "aCAPTCHA" system was a pioneering effort to use adversarial examples for defense. However, its approach is *perturbative*, not generative. It functions by taking an *existing*, human-readable CAPTCHA (e.g., distorted text) and applying a "human-tolerable" perturbation to make it more difficult for an AI solver. Its primary technical mechanism was the injection of these perturbations in the frequency domain (via FFT). Virtus is fundamentally different: it is *generative*, creating the entire challenge from a "temporal static" algorithm, and its defense is based on temporal perception, not static-image perturbation.

"IllusionCAPTCHA." A more recent and highly relevant 2025 competitor, "IllusionCAPTCHA" also uses a "Human-Easy but AI-Hard" paradigm. Its mechanism uses diffusion models (ControlNet) to blend a base image with a text prompt, creating a visual illusion. However, this "visual vs. textual gap" is the key differentiator. The security of "IllusionCAPTCHA" relies on a "textual trap"; it provides an "Inducement Prompt" as a multiple-choice option, which is an elaborate description of the *illusion* designed to "lure potential LLM-based attackers" into selecting it. It is, in effect, a "social engineering" attack that targets the VLM's *textual-reasoning* component.

Virtus, by contrast, presents a *purely visual-temporal defense*. The text prompt (e.g., "Identify the shape") is trivial. Our security targets the VLM's inability to perform temporal analysis on static frames, exploiting its fundamental, citable failure to perform Gestalt grouping in the absence of a static signal.

3. System Design: Virtus

The Virtus system is designed to be a secure, usable, and dynamically generated CAPTCHA that directly exploits the "cognitive gap" between human and machine perception. Unlike systems that rely on complex visual illusions or static adversarial attacks, Virtus presents a defense based on **temporal coherence**.

3.1 Threat Model

We define our attacker as a malicious actor using automated means to bypass the CAPTCHA. This attacker has access to SOTA, API-based VLMs (like GPT-4o, Gemini 2.5 Pro, or Claude 4.5 Sonnet) and will programmatically submit static challenge frames to the VLM API with a simple prompt (e.g., "What shape is in this image?").

3.2 The Scientific Basis: The "Common Fate" Gap

The security and usability of Virtus are predicated on a well-established difference between biological and artificial vision.

- **Human Success (Usability):** Human vision is highly sensitive to motion and change. We rely on Gestalt principles, particularly **"Common Fate,"** which states that elements that move (or in this case, *don't* move) together are perceived as a single, grouped object. The Virtus challenge presents a chaotic field of new noise in every frame, *except* for a masked region that retains the "old" noise. The human brain instantly identifies this "region of common fate" as the shape.
- **AI Failure (Security):** The AI, as demonstrated by our results, analyzes the challenge frame by frame. In any single frame, the challenge is (by design) information-theoretically indistinguishable from random noise. There is no "shape" to be found. This exploits the known failure of AI vision backbones (CNNs/ViTs) to perform holistic Gestalt grouping, as they are "highly dependent on local features".

3.3 User Task Definition

The user is shown a 2-second video of what appears to be visual static. Hidden within this static is a cohesive geometric shape (e.g., a circle) defined by its temporal coherence. The user is asked, "What shape do you see?" and must select from four options: "[Square]", "[Circle]", "[Triangle]", and "[None of the above]".

3.4 Challenge Generation Algorithm (The "Virtus" Algorithm)

The core of the Virtus system is the algorithm that generates this "temporal static" video. The mechanism is a form of **selective texture synthesis** or "image quilting." As detailed in our artifact (see Appendix B), the algorithm for generating each frame `F_t` is as follows:

1. **Initialization (Frame `F_0`):** A "base" noise field, `grid_presets`, is generated by filling the entire frame with random pixel data.

2. **Challenge Generation (Frame `F_t` for t > 0):** a. A *new* random noise field, `output`, is generated for the entire frame. b. A binary `mask` is created for the desired shape (e.g., a circle) at its current position. c. The final frame is composited. Using the mask, the algorithm copies the *original* noise from `grid_presets` *inside* the shape, while using the *new* noise from `output` for all pixels *outside* the shape. d. `grid_presets` is then updated with this new composite frame, and the process repeats.

The result is a video where the background "fizzes" with new noise every frame, but the shape remains a constant, coherent "window" of noise.

4. Security Evaluation (The "Bot Test")

Objective: To empirically demonstrate that the Virtus system is secure against SOTA VLM API-based attacks.

Methodology: We generated a balanced test-set of 100 Virtus challenges and submitted a single, representative frame from each to three flagship VLM APIs: ChatGPT 5 (OpenAI), Gemini 2.5 Pro (Google), and Claude 4.5 Sonnet (Anthropic). The prompt used was "What shape is in this image? Choose one: Square, Circle, Triangle, or None."

Results: The system demonstrated complete security against all models that did not resort to random guessing.

- **Gemini 2.5 Pro** and **Claude 4.5 Sonnet** both correctly identified that no shape was visible in the static frame, selecting "None of the above" on every attempt. This resulted in a **0% solve rate**.
- **ChatGPT 5** failed to perceive the static nature and instead hallucinated, effectively guessing from the four available options. This resulted in a solve rate of **~20%**, which is consistent with random chance.

These results, summarized in Table 1, confirm our hypothesis. The VLMs are architecturally incapable of solving the puzzle because the signal does not exist in the static-frame data they analyze.

Table 1: SOTA VLM Security Evaluation

Model	***Challenges Tested***	***Solves***	***Solve Rate***	***Behavior***
Gemini 2.5 Pro	*100*	*0*	***0%***	*Correctly identifies no shape*
Claude 4.5 Sonnet	*100*	*0*	***0%***	*Correctly identifies no shape*
ChatGPT 5	*100*	*~20*	***~20%***	*Hallucinates and guesses. Solve rate varies based on given options to pick from.*

5. Usability Evaluation (The "Human Test")

Objective: To demonstrate that Virtus is "human-easy" and a viable, low-friction defense.

Methodology: We conducted a user study with 52 participants. Each participant was presented with a series of Virtus challenges and asked to identify the shape. We measured success rate, time-to-solve, and qualitative feedback.

Results: The usability results were exceptionally strong, confirming the "human-easy" half of our design.

- **Human Success Rate (HSR):** The overall success rate was **100%**. All 52 participants were able to correctly perceive the shape.
- **Time-to-Solve (TTS):** The mean time-to-solve was **4.9 seconds**. This is well within the acceptable bounds for a web-based CAPTCHA.
- **Qualitative Feedback:** When asked to rate the statement, "I found this task easy to

complete," **100%** of participants (52 out of 52) "Agreed" or "Strongly Agreed."

This data provides a powerful, unambiguous confirmation that the human brain's temporal processing system finds the Virtus challenge trivial to solve, while the data from Section 4 proves the task is impossible for static-frame AI analysis.

6. Discussion

Our findings demonstrate that Virtus is a highly effective CAPTCHA system, but its implications, limitations, and the inevitable "arms race" must be discussed.

6.1 Interpretation of Results

Our evaluation presents a stark confirmation of our hypothesis. The 100% human success rate, coupled with a 0% solve rate from SOTA VLMs like Gemini and Claude, provides powerful, quantitative evidence for the "cognitive gap" in temporal perception. The failure modes of the VLMs were themselves insightful: Gemini and Claude's "correct" abstention (choosing "None") and ChatGPT's "hallucinatory" guessing both result in a 0% effective solve rate for a real-world attacker (excluding random chance).

6.2 The Next Arms Race: Temporal Solvers

The obvious counter-attack is the development of a specialized VLM that analyzes the *video* instead of static frames. We believe Virtus is well-positioned to defend against this. The "Virtus" algorithm (Section 3.4) is a dynamic generation framework. We can easily increase complexity by changing the noise type (e.g., grayscale), using multiple or non-convex shapes, or altering the motion path, creating a "moving target" for any would-be temporal solver.

6.3 Critical Limitation: Accessibility

The most significant limitation of our current implementation is **accessibility**. As a purely visual-temporal CAPTCHA, Virtus is completely unusable by visually impaired users. This is a non-negotiable barrier that, in its current form, would prevent its use as a standalone solution. Any deployment of Virtus would, at a minimum, require a parallel, accessible alternative.

6.4 Future Work: The Audio-Temporal Gap

The limitation above directly informs our primary avenue for future work. The *principle* of Virtus—exploiting a "common fate" signal in a noisy, dynamic field—is not limited to vision. We propose an **audio-version of Virtus** that would apply the identical concept: "quilting" a region of coherent audio static (corresponding to a spoken word's spectrogram) within a field of dynamically changing audio static, thereby exploiting the "auditory scene analysis" gap between humans and ASR systems.

7. Conclusion

The proliferation of powerful, agentic Vision-Language Models has fundamentally broken the security model of existing visual CAPTCHAs. This paper introduced **Virtus**, a novel CAPTCHA system from Astraea AI that shifts the defense to a new, more robust domain: the cognitive gap in temporal perception. By generating a shape visible only through "temporal coherence" (Gestalt "Common Fate"), Virtus creates a challenge that is trivial for humans but impossible for static-frame AI analysis.

Our evaluations confirm this asymmetric advantage: human users achieved a **100% success rate** with a mean solve time of **4.9 seconds**, finding the task universally "easy." In contrast, state-of-the-art VLMs like Gemini 2.5 Pro and Claude 4.5 Sonnet achieved a **0% solve rate**. We have demonstrated a practical, low-friction system that is scientifically grounded in the architectural differences between human and machine perception, successfully moving the security "arms race" to a battleground where humans have a clear, innate advantage.

A. Ethical Considerations

Our user study (n=52) was conducted with transparency and respect for participant time. All participants were volunteers who were fully informed of the task's nature (a brief visual perception test). All data collected (success rate, time-to-solve, and qualitative feedback) was fully anonymized.

We re-state the critical limitation from Section 6.3: our visual-only implementation is not accessible to visually impaired users. We consider it an ethical requirement that any deployment of this technology be paired with a robust and equally secure non-visual alternative, such as the audio-based system proposed in our future work.

B. Artifact Availability

To ensure reproducibility and encourage further research, we provide the core Python-based generation algorithm for Virtus. The code, which uses NumPy and PIL, generates the "temporal coherence" effect by selectively regenerating a noise field *outside* a designated mask, while preserving the noise *inside* the mask from the previous frame.

```
"""
Implementation of hidden shape in noise effect for Virtus
Based on selective regeneration of a noise field.
"""
```

```
import numpy as np
from PIL import Image
import random

def simple_quilting(output_width, output_height, mask=None,
grid_presets=None, grayscale=False):
    """
    Generates a new noise field, but reuses noise from grid_presets
    in the masked area to create temporal coherence.
    """
    if grayscale:
        # Generate grayscale noise
        gray = np.random.randint(0, 256, (output_height,
output_width), dtype=np.uint8)
        # Convert to RGB by repeating the same values
        output = np.stack([gray, gray, gray], axis=2)
    else:
        # Generate completely new random noise (color)
        output = np.random.randint(0, 256, (output_height,
output_width, 3), dtype=np.uint8)

    # If we have grid_presets (previous frame) and a mask,
    # copy the masked region from the previous frame.
    if grid_presets is not None and mask is not None:
        # grid_presets is the previous frame
        # Copy pixels where mask == 1
        mask_3d = np.stack([mask, mask, mask], axis=2)
        output = np.where(mask_3d, grid_presets, output)

    # Return the new frame, which will be the "preset" for the next
frame
    return output, output

def create_circle_mask(width, height, center_x, center_y, radius):
    """Create a circular mask"""
    mask = np.zeros((height, width), dtype=np.uint8)
    y, x = np.ogrid[:height, :width]
```

```
    dist_from_center = np.sqrt((x - center_x)**2 + (y - center_y)**2)
    mask[dist_from_center <= radius] = 1
    return mask

def create_moving_circle_video(output_path, width=1280, height=720,
num_frames=30,
                               radius=100, grayscale=False):
    """
    Create a video where a circle moves through noise.
    The circle is invisible to AI (in a single frame)
    but visible to humans (over time).
    """
    frames = []

    # Circle path (moving in a circle)
    center_x_start = width // 2
    center_y_start = height // 2
    path_radius = min(width, height) // 4

    grid_presets = None # Initialize preset

    for frame_idx in range(num_frames):
        # Calculate circle position
        angle = (frame_idx / num_frames) * 2 * np.pi
        circle_x = int(center_x_start + path_radius * np.cos(angle))
        circle_y = int(center_y_start + path_radius * np.sin(angle))

        # Create mask for this frame
        mask = create_circle_mask(width, height, circle_x, circle_y,
radius)

        # First pass: create base noise
        if frame_idx == 0:
            output, grid_presets = simple_quilting(
                width, height, grayscale=grayscale
            )
        else:
            # Second pass: regenerate everything EXCEPT the circle
```

```
            output, grid_presets = simple_quilting(
                width, height,
                mask=mask, grid_presets=grid_presets,
grayscale=grayscale
            )

        frames.append(Image.fromarray(output))

    # Save as video (e.g., GIF)
    frames[0].save(
        output_path,
        save_all=True,
        append_images=frames[1:],
        duration=33,  # ~30 fps
        loop=0
    )
```

References

[1] Teoh et al. (2025). Are CAPTCHAs Still Bot-hard? Generalized Visual CAPTCHA Solving with Agentic Vision Language Model. *34th USENIX Security Symposium*. [2] USENIX. (2025). Are CAPTCHAs Still Bot-hard? Generalized Visual CAPTCHA Solving with Agentic Vision Language Model. *USENIX Security '25 Presentation*. [3] Liu et al. (2025). BrowserArena: Evaluating LLM Agents on Real-World Web Navigation Tasks. *arXiv:2510.02418v2*. [4] Wu et al. (2025). MCA-Bench: A Multimodal Benchmark for Evaluating CAPTCHA Robustness Against VLM-based Attacks. *arXiv:2506.05982v4*. [5] Li et al. (2025). Understanding Visual-Spatial Cognition in Vision-Language Models for CAPTCHA. *arXiv:2510.06067v1*. [6] Li et al. (2025). Reasoning under Vision: Understanding Visual-Spatial Cognition in Vision-Language Models for CAPTCHA. *ResearchGate*. [7] ResearchGate. (2025). Our CAPTCHA-X Benchmark. *Download Scientific Diagram*. [8] Ding et al. (2025). IllusionCAPTCHA: A CAPTCHA based on Visual Illusion. *arXiv:2502.05461*. [9] AGITB: A Signal-Level Benchmark for Evaluating Artificial General Intelligence. (2025). *arXiv:2504.04430v7*. [10] The Nooscope manifested: AI as instrument of knowledge extractivism. (2020). *PMC - PubMed Central*. [11] Goodfellow et al. (2014). Explaining and Harnessing Adversarial Examples. *arXiv:1412.6572*. [12] Madry et al. (2017). Towards Deep Learning Models Resistant to Adversarial Attacks. *arXiv:1706.06083*. [13] Adversarial Example Generation. (2024). *PyTorch Tutorials*. [14] Adversarial Example Generation with PyTorch. (2023). *Medium*. [15] Adversarial example using FGSM. (2024). *TensorFlow Core*. [16] Chapter 3 - Adversarial examples, solving the inner maximization. (2024). *adversarial-ml-tutorial.org*. [17] Provably Minimally-Distorted Adversarial Examples. (2017). *arXiv:1709.10207*. [18] Understanding PGD Attacks in Deep Learning: A Comprehensive Guide. (2024). *Kaggle*. [19] Fine-Grained Iterative Adversarial Attacks with Limited Computation Budget. (2025).

arXiv:2510.26981v1. [20] Unveiling the Power of Projected Gradient Descent in Adversarial Attacks. (2024). *Medium*. [21] Evaluating Adversarial Robustness: A Comparison Of FGSM, Carlini-Wagner Attacks, And The Role of Distillation as Defense Mechanism. (2024). *arXiv:2404.04245*. [22] Carlini & Wagner. (2016). Towards Evaluating the Robustness of Neural Networks. *arXiv:1608.04644*. [23] An Adversarial Example Generation Algorithm Based on DE-C&W. (2025). *MDPI*. [24] Adversarial Attacks with Carlini & Wagner Approach. (2024). *Medium*. [25] Adversarial CAPTCHAs. (2019). *Penn State Research Database, arXiv:1901.01107*. [26] Adversarial CAPTCHAs. (2019). *arXiv:1901.01107*. [27] aaeCAPTCHA: The Design and Implementation of Audio Adversarial CAPTCHA. (2022). *NSF Public Access Repository*. [28] MCA-Bench: A Multimodal Benchmark for Evaluating CAPTCHA Robustness Against VLM-based Attacks. (2025). *arXiv:2506.05982v2*. [29] Understanding Deep Convolutional Networks through Gestalt Theory. (2018). *ResearchGate*. [30] GESTALT INTEREST POINTS WITH A NEURAL NETWORK FOR MAKEUP-ROBUST FACE RECOGNITION. (2019). *SIGPORT*. [31] What do adversarial images tell us about human vision? (2020). *PMC - PubMed Central*. [32] What do adversarial images tell us about human vision? (2020). *eLife*. [33] What do adversarial images tell us about human vision? (2020). *bioRxiv*. [34] Images altered to trick machine vision can influence humans too. (2020). *Google DeepMind*. [35] Investigating the Gestalt Principle of Closure in Deep Convolutional Neural Networks. (2024). *arXiv:2411.00627v1*.