Does the Source of Carrier Image Affect Steganographic Detectability?

Midway Project Proposal

Project 2.2, Department of Advanced Computing Sciences

Abdul Moiz Akbar Malo Coquin Daria Gjonbalaj Nico Müller-Späth
Jimena Narvaez del Cid David Wicker Nikolas Zouros
Maastricht University · February 2026

1. Motivation and Problem Statement

Image steganography is an important topic in online security and privacy, with both beneficial and adversarial uses [1, 2, 10]. In practice, detectability depends not only on the embedding method but also on the underlying statistics of the carrier image [3, 4]. Steganalysis works by detecting subtle distributional perturbations in pixel values, residuals, and frequency coefficients after embedding. This is exactly why carrier selection matters: two carriers with identical payload and identical embedding can produce different detectability outcomes.

Most existing steganalysis pipelines were developed and benchmarked on real camera photographs, where sensor noise and compression traces are relatively well characterized. This assumption is increasingly fragile. Photorealistic synthetic images from modern generators such as Stable Diffusion and StyleGAN3 are now widespread [5, 6], and they are produced by different mechanisms than camera pipelines. Prior synthetic-image forensics shows that generated images can contain distinct statistical traces [12, 13], which may directly influence steganographic detectability.

The core question is therefore: **do steganalysis detectors validated on photographs remain equally effective on ML-generated carriers under identical embedding settings?** This matters for three reasons:

- **Security:** if ML-generated carriers are harder to detect, an attacker may evade existing detectors by changing carrier source only.
- **Scientific:** the interaction between generative-model distributions and embedding perturbations remains underexplored.
- **Practical:** as AI-generated images become common in communication channels, the steganographic attack surface expands.

The problem statement for this project is to quantify whether *carrier origin alone* (real vs. ML-generated) causes measurable detection differences when embedding, payload, and detector settings are fixed. If origin effects are large, detector calibration and risk assumptions built on photo-only datasets may not transfer to mixed real/synthetic traffic.

The closest precedent is De et al. [7], who demonstrate AI-generated images for steganographic secret sharing, but without a controlled real-vs-ML carrier comparison using standardized LSB/DCT pipelines and classical steganalysis endpoints. In other words, feasibility of secret sharing on generated images has been shown, but comparative detectability under matched conditions is still unclear.

This study addresses that gap with a controlled $2\times2\times3\times2$ design over 1,000 images (500 real, 500 ML-generated), two embedding methods, three payload levels, and two primary detectors. By keeping embedding, payload, and detector settings fixed while varying carrier origin, we isolate whether origin itself is a meaningful source of detection-performance differences. Results will guide benchmarking and steganalysis policy in mixed real/synthetic environments.

2. Research Questions

RQ1 (Carrier Origin)
Is there any effect of carrier-image origin (real vs. ML-generated) on detectability of hidden data?
Verification: Compare real vs. ML AUC with RS and SRM+FLD at identical settings; report significance and effect size.

RQ2 (Payload)
Does increasing payload size widen the detectability gap between real and ML-generated carriers?
Verification: Measure AUC gap across Low/Medium/High payloads and test trend plus carrier×payload interaction.

RQ3 (Encryption)
Does encrypting payload before embedding make steganography harder or easier to detect, and does origin change this effect?
Verification: Compare plain vs. AES-256-CBC AUC by carrier and method; test significance of the difference.

RQ4 (Embedding Method)
Do different embedding methods (spatial LSB vs. frequency-domain DCT) interact differently with carrier origin in terms of detectability?
Verification: Run a two-way ANOVA on AUC for carrier origin and embedding method; inspect interaction term.

RQ5 (Image Quality)
Is image quality affected by embedding method and payload size?
Verification: Compare PSNR/SSIM/FSIM across conditions and check against quality targets (PSNR >40 dB, SSIM >0.95).

3. Chosen Approaches

3.1 Datasets

Real images (500 total). We draw from three established photographic datasets chosen for their diversity and research accessibility: **RAISE** [14] contributes 250 RAW-demosaiced DSLR images spanning outdoor, indoor, portrait, and macro scenes; **COCO** [15] contributes 150 images from its validation split; and **Flickr30k** [16] contributes 100 images. All images are normalized to 512×512 px, RGB, 8-bit, lossless PNG. RAISE is preferred as the primary source because its RAW format preserves camera sensor noise structure, which is the natural image statistics that steganalysis exploits.

ML-generated images (500 total). We generate two matched sets of 250 images each using **Stable Diffusion v2.1** [5] (via the `diffusers` library on Apple MPS) and **StyleGAN3** [6] (official NVIDIA PyTorch implementation). Prompts for SD are derived directly from COCO/Flickr30k captions to achieve semantic alignment with real images. A BRISQUE ≤ 50 quality gate rejects perceptually degraded outputs.

These two generative paradigms (latent diffusion and GAN-based synthesis) represent the dominant architectures in open-source image generation and are expected to impose distinct statistical signatures on the output.

3.2 Embedding Methods

We implement two canonical steganographic methods spanning the two principal domains:

LSB substitution (spatial domain) replaces the k least significant bits of each pixel channel value with pseudorandom message bits, using a PRNG-keyed pixel selection mask. We test $k=1$ (Low, Medium payload) and $k=2$ (High payload). Payloads are optionally pre-encrypted with AES-256-CBC before embedding, addressing RQ3.

DCT-based embedding (frequency domain) partitions each image channel into non-overlapping 8×8 blocks, computes the 2D DCT, and embeds bits into mid-frequency coefficients (zigzag positions 10–54) via **Quantization Index Modulation** (QIM) [11]:

$$C_i' = \Delta \cdot \text{round}\left(\tfrac{C_i}{\Delta}\right) \pm \tfrac{\Delta}{4},$$

where the sign encodes the message bit. DCT embedding is chosen alongside LSB to test whether frequency-domain methods are more sensitive to carrier origin (RQ4), since DCT coefficients reflect the generative model's learned spectral distribution directly.

3.3 Steganalysis Detectors

Our detector selection is deliberately scoped to *classical signal processing and statistics*, consistent with this project's cryptography/steganography focus:

Table 1: Steganalysis detector comparison.

Detector	Type	Training	Domain	Time
RS Analysis [8]	Statistical	None	Any	∼2 s/img
χ^2 attack [9]	Statistical	None	LSB	<1 s/img
SRM+FLD [4]	Classical ML	Labeled	LSB+DCT	<30 min

RS Analysis [8] partitions an image into pixel groups and classifies each as Regular or Singular by a smoothness function; LSB embedding shifts the R/S ratio predictably, yielding an analytical estimate $\hat{p}$ of embedding rate. Since it requires no training, any detection difference between real and ML-generated carriers is attributable solely to carrier statistics and not to classifier bias.

SRM + Fisher Linear Discriminant (FLD) ensemble [4] extracts ∼35,000-dimensional co-occurrence feature vectors from high-pass residuals, then classifies with an ensemble of FLD classifiers. It handles DCT embedding better than the training-free methods and its hand-crafted features are hypothesised to generalise better across the real/ML boundary than learned neural network representations. Implemented with `scikit-learn`'s `SGDClassifier`; 3-fold stratified CV.

The χ^2 attack [9] is applied as a supplementary check on LSB results.

3.4 Validation

Detection: ROC-AUC (primary, threshold-independent); accuracy at Youden's J; EER; FPR at 5% FNR. **Image quality:** PSNR (>40 dB target), SSIM (>0.95 target), FSIM. **Statistics:** Two-way ANOVA (carrier × method) on AUC with payload as covariate; Wilcoxon signed-rank for pairwise comparisons; Cohen's d effect sizes; Bonferroni correction ($\alpha_{\text{adj}} = 0.05/6 \approx 0.0083$) across six hypotheses.

4. Experiments

Each research question maps to exactly one experiment; every experiment links back to its RQ.

Exp. 1 (RQ1: Carrier Origin). Apply RS Analysis and SRM+FLD to all 1,000 images across payload levels and methods. Compute AUC per carrier type and compare real vs. ML-generated performance with Wilcoxon signed-rank and Cohen's d (Bonferroni-corrected).

Exp. 2 (RQ2: Payload). From Exp. 1 results, plot AUC vs. payload level (Low/Medium/High) separately for real and ML-generated carriers, with separate curves per embedding method. Test whether the real–ML AUC gap increases monotonically using Spearman's ρ, and test carrier × payload interaction.

Exp. 3 (RQ3: Encryption). For each carrier type and embedding method, compare AUC between plain-payload and AES-256-CBC-encrypted-payload conditions. Test whether encryption changes detectability and whether the effect differs by carrier origin.

Exp. 4 (RQ4: Embedding Method). Run a 2×2 two-way ANOVA with factors carrier origin (real/ML) and embedding method (LSB/DCT) on SRM AUC scores. A significant interaction indicates method-dependent detectability differences across carrier origin.

Exp. 5 (RQ5: Image Quality). Compute PSNR, SSIM, and FSIM for each stego condition relative to its cover image. Compare quality metrics across payload levels and embedding methods, and confirm whether quality remains within target ranges.

5. Prototype

Vertical prototype (algorithm depth): We validate the core components separately before scaling: LSB embedding/extraction, DCT-QIM embedding, RS Analysis, and SRM feature extraction/classification on a small test subset.

Horizontal prototype (integration breadth): We then run an integrated pipeline on a mixed 50-image subset (25 real + 25 ML) at medium payload to validate interfaces and outputs before executing the full 1,000-image study.

6. Related Work

6.1 Classical Embedding and Detection Foundations

Image steganography has predominantly focused on embedding secret data into photographic carriers using spatial- and frequency-domain techniques [1–3]. Canonical examples include LSB substitution, which perturbs pixel intensities directly, and DCT-based Quantization Index Modulation (QIM), which embeds information in transform coefficients [11]. We adopt these established post-hoc embedding strategies rather than introducing a new embedding algorithm, to preserve comparability with prior steganalysis literature and isolate the effect of carrier origin.

On the detection side, RS analysis and the χ^2 attack provide analytical baselines for LSB-like embedding [8, 9]. Rich-model approaches remain strong classical baselines: Fridrich and Kodovský [4] introduced SRM features that capture high-dimensional residual co-occurrences linked to embedding artifacts.

6.2 Generative and Coverless Steganography

Recent generative steganography methods use GAN- or diffusion-based models to generate stego-images directly from secret messages [19, 20]. Related coverless approaches encode information by selecting or generating content rather than modifying a fixed image [21]. Our setup differs in a key way: ML-generated images are treated as *passive carriers*, and standardized post-hoc embedding (LSB and DCT) is applied afterward. This separation between generation and embedding lets us test carrier-origin effects under matched embedding conditions.

6.3 AI-Generated Carriers and Closest Prior Work

The closest related study is De et al. [7], which demonstrates AI-generated photorealistic images for steganographic secret sharing. However, it does not perform a controlled real-versus-ML comparison under identical embedding pipelines, nor does it evaluate detectability using classical steganalysis endpoints such as ROC-AUC. Our factorial design (carrier origin × embedding method × payload size × encryption) makes steganalysis detectability the primary outcome.

6.4 Cross-Domain and Synthetic-Image Forensics

Cross-domain steganalysis has typically examined distribution shifts between camera sources [4]. The real-versus-synthetic shift is broader, because the full image-generation process changes when moving from camera capture to GAN or diffusion synthesis. Synthetic-image forensics reports distinct statistical traces in generated images [12, 13], supporting our hypothesis that carrier origin can affect steganographic detectability.

6.5 Classical vs. Deep Steganalysis

Recent deep-learning steganalysis methods can achieve strong benchmark accuracy [17], but often with higher computational requirements and reduced interpretability. We therefore prioritize SRM+FLD and RS analysis to maintain interpretability and CPU feasibility, in line with the cryptography-focused scope of this project. Overall, prior work has studied embedding techniques, steganalysis models, generative steganography, and synthetic-image detection largely in isolation; our contribution is a controlled evaluation of how carrier origin influences detectability.

7. Relation to Curriculum

This project connects directly to **Computer Security** (AES-256-CBC and adversarial threat modeling), **AI and Machine Learning** (SRM+FLD feature-based classification and cross-domain evaluation), **Software Engineering and Architectures** (modular, testable pipeline design), and **Statistics** (ANOVA, non-parametric testing, effect sizes, and confidence intervals for rigorous comparison).

8. Planning

The project is divided into two phases aligned with the Semester 2 academic calendar. Detailed Gantt charts are provided in Appendix A.

Phase 2: Implementation (Period 5, 30 Mar–15 May 2026, 7 weeks) covers three parallel workstreams: dataset construction and ML image generation (Wk 1–2), steganography pipeline implementation including LSB, DCT, and AES-256 encryption (Wk 2–3), and detection, analysis, and writing (Wk 3–7).

Phase 3: Completion (Project Period, 25 May–12 Jun 2026, 3 weeks) covers completing any remaining implementation, verifying and rerunning experiments, and finalising all deliverables (presentation slides, poster, and paper).

9. Minimal Passing Requirements

In terms of approaches, the minimum is one encryption algorithm and two embedding methods: one in the spatial domain and one in the frequency domain.

From the five research questions, at minimum RQ3 (encryption) and RQ4 (embedding-method interaction) must be answered.

References

[1] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, “Information hiding: a survey,” *Proc. IEEE*, vol. 87, no. 7, pp. 1062–1078, 1999.

[2] A. Cheddad, J. Condell, K. Curran, and P. McKevitt, “Digital image steganography: Survey and analysis of current methods,” *Signal Process.*, vol. 90, no. 3, pp. 727–752, 2010.

[3] M. Hussain, A. W. A. Wahab, Y. I. B. Idris, A. T. S. Ho, and K. H. Jung, “Image steganography in spatial domain: A survey,” *Signal Process.: Image Commun.*, vol. 65, pp. 46–66, 2018.

[4] J. Fridrich and J. Kodovský, “Rich models for steganalysis of digital images,” *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 3, pp. 868–882, 2012.

[5] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proc. IEEE CVPR*, pp. 10684–10695, 2022.

[6] T. Karras, M. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Alias-free generative adversarial networks,” in *Proc. NeurIPS*, vol. 34, pp. 852–863, 2021.

[7] A. De, W. Kinzel, and I. Kanter, “Steganographic secret sharing via AI-generated photorealistic images,” *EURASIP J. Wireless Commun. Netw.*, art. 108, 2022.

[8] J. Fridrich, M. Goljan, and R. Du, “Detecting LSB steganography in color and grayscale images,” *IEEE Multimedia*, vol. 8, no. 4, pp. 22–28, 2001.

[9] A. Westfeld and A. Pfitzmann, “Attacks on steganographic systems,” in *Proc. 3rd Int. Workshop Information Hiding*, LNCS 1768, pp. 61–76, 1999.

[10] N. Provos and P. Honeyman, “Hide and seek: An introduction to steganography,” *IEEE Security Privacy*, vol. 1, no. 3, pp. 32–44, 2003.

[11] B. Chen and G. W. Wornell, “Quantization index modulation: A class of provably good methods for digital watermarking and information embedding,” *IEEE Trans. Inf. Theory*, vol. 47, no. 4, pp. 1423–1443, 2001.

[12] S. Y. Wang, O. Wang, R. Zhang, A. Owens, and A. A. Efros, “CNN-generated images are surprisingly easy to spot. . . for now,” in *Proc. IEEE CVPR*, pp. 8695–8704, 2020.

[13] R. Corvi, D. Cozzolino, G. Zingarini, G. Poggi, K. Nagano, and L. Verdoliva, “On the detection of synthetic images generated by diffusion models,” in *Proc. IEEE ICASSP*, pp. 1–5, 2023.

[14] D.-T. Dang-Nguyen, C. Pasquini, V. Conotter, and G. Boato, “RAISE: A raw images dataset for digital image forensics,” in *Proc. ACM MMSys*, pp. 219–224, 2015.

[15] T.-Y. Lin et al., “Microsoft COCO: Common objects in context,” in *Proc. ECCV*, LNCS 8693, pp. 740–755, 2014.

[16] P. Young, A. Lai, M. Hodosh, and J. Hockenmaier, “From image descriptions to visual denotations,” *Trans. Assoc. Comput. Linguist.*, vol. 2, pp. 67–78, 2014.

[17] Y. Luo et al., “Deep learning for steganalysis of diverse data types: A review,” *Neurocomputing*, Elsevier, 2024.

[18] V. Holub, J. Fridrich, and T. Denemark, “Universal distortion function for steganography in an arbitrary domain,” *EURASIP J. Inf. Security*, art. 1, 2014.

[19] P. Hu et al., “A coverless image steganography based on generative adversarial networks,” *Electronics*, vol. 12, no. 5, art. 1253, 2023.

[20] X. Liu et al., “Message-driven generative steganography using GAN,” *IEEE Trans. Dependable Secure Comput.*, 2024.

[21] X. Duan, D. Jia, B. Li, D. Guo, E. Zhang, and C. Qin, “Coverless steganography based on generative adversarial network,” *EURASIP J. Image Video Process.*, art. 46, 2020.

A. Project Gantt Charts

Phase 2: Implementation

Period 5: 30 March – 15 May 2026 (7 weeks)

WBS	Task	Start	End	Days	Wk 1	Wk 2	Wk 3	Wk 4	Wk 5	Wk 6	Wk 7
1	**Data Construction**										
1.1	Dataset collection (RAISE/COCO/Flickr30k)	30.03	10.04	10	🟦	🟦					
1.2	ML image generation (SD v2.1 + StyleGAN3)	30.03	10.04	10	🟦	🟦					
1.3	BRISQUE quality gate & normalisation	07.04	17.04	8		🟦	🟦				
2	**Steganography Implementation**										
2.1	LSB embedding pipeline ($k=1,2$)	07.04	17.04	8		🟥	🟥				
2.2	DCT-QIM embedding pipeline	07.04	17.04	8		🟥	🟥				
2.3	AES-256-CBC payload encryption	14.04	17.04	4			🟥				
3	**Detection & Analysis**										
3.1	RS Analysis + χ^2 detector	14.04	24.04	8			🟩	🟩			
3.2	SRM + FLD training	14.04	01.05	14			🟩	🟩	🟩		
3.3	Image quality metrics (PSNR/SSIM/FSIM)	21.04	01.05	8				🟧	🟧		
3.4	Encryption-effect experiments (RQ3)	28.04	08.05	8					🟪	🟪	
3.5	Statistical analysis (ANOVA, Wilcoxon)	28.04	08.05	8					🟪	🟪	
4	**Writing & Revision**										
4.1	Visualisations + report writing	04.05	15.05	8						▨	▨
4.2	Buffer / final revision	11.05	15.05	5							▨

🟦 Data 🟥 Steganography 🟩 Detection 🟧 Evaluation 🟪 Analysis ▨ Writing

Milestones: **M1** (end Wk 2): Dataset ready **M2** (end Wk 3): Embedding pipelines verified **M3** (end Wk 5): All experiments complete **M4** (end Wk 7): Report submitted

Phase 3: Completion

Project Period: 25 May – 12 June 2026 (3 weeks)

WBS	Task	Start	End	Days	Wk 1	Wk 2	Wk 3
1.1	Complete remaining implementation	25.05	05.06	10	🟦	🟦	
1.2	Verify results & rerun experiments	25.05	05.06	10	🟩	🟩	
1.3	Statistical verification & effect sizes	01.06	05.06	5		🟧	
2.1	Presentation slides	01.06	08.06	6		🟪	🟪
2.2	Poster	05.06	10.06	4			🟥
2.3	Final paper write-up	01.06	12.06	10		▨	▨
2.4	Buffer / revision / submission	10.06	12.06	3			▨

🟦 Implementation 🟩 Verification 🟧 Analysis 🟪 Slides 🟥 Poster ▨ Paper

Milestones: **M5** (end Wk 1): Implementation finalised **M6** (end Wk 2): Results verified, slides ready **M7** (end Wk 3): All deliverables submitted