

Telligram: Text-Driven Calligram Generation via Diffusion-Guided Skeleton Optimization

Tianci Shi[✉] and Pengfei Xu[†][✉]

CSSE, Shenzhen University, China

Abstract

Compact calligram generation aims to form a semantic shape while keeping letters recognizable. Most existing methods are shape-conditioned and mainly solve downstream letter layout inside a given contour. We study text-only calligram generation without an input contour. This setting is difficult because semantic shape formation and letter readability strongly interfere with each other when optimized in a single stage. Pushing the word toward a clear figure can easily damage glyph structure, while preserving readable letters can weaken the target shape. To address this difficulty, we present Telligram, a training-free, low-tuning, two-stage framework composed of Semantic Occupancy Prior Formation and Readability-Constrained Glyph Realization. The first stage uses Variational Score Distillation (VSD) with structured skeleton optimization and hierarchical gradient projection to produce a semantic occupancy prior. The second stage converts this occupancy prior into per-letter regions and reconstructs readable glyph layouts through lightweight geometric processing. The framework generates coherent and creative word-level semantic calligrams directly from text prompts.

CCS Concepts

• **Computing methodologies** → **Computer graphics**;

1. Introduction

Calligrams are a form of visual art in which letters or words are arranged to form an image that reflects the meaning of the text [Wik26]. By blending readability with visual form, calligrams are widely used in logos, advertisements, posters, and book covers to achieve both clarity and aesthetic appeal. Creating a calligram involves balancing two goals: maintaining legible text and shaping it into a recognizable figure [MBS*11, ZCR*16]. Legibility is an important criterion in this setting, because a successful calligram should remain readable while forming a recognizable figure. This dual requirement makes calligram generation a joint optimization problem over textual structure and geometric shape.

Unlike long-text micrography, which typically relies on line-level alignment with minimal glyph deformation [MBS*11], calligrams demand greater geometric control over the placement and deformation of individual glyphs. If the design optimizes solely for shape similarity, letter integrity may be compromised; if it focuses only on readability, the intended figure may become unclear. Thus, the central challenge lies in finding a stable compromise among semantic fidelity, topological coherence, and spatial coverage.

In practice, designers usually choose a target word, find a semantically compatible shape, and then refine letter placement and lay-

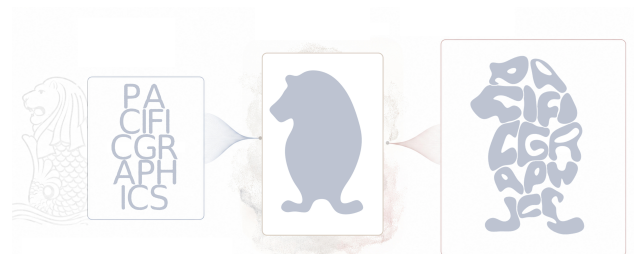


Figure 1: Overview of Telligram. Given the pre-arranged word “PACIFIC GRAPHICS” and the semantic prompt “Merlion”, the method first deforms the glyphs to form a semantic occupancy prior and then generates a readable final calligram result.

out. The dominant cost is typically incurred at the front-end word-to-shape selection stage rather than in downstream local tuning. In other words, the hardest step is identifying structures that are both semantically appropriate and geometrically writable.

This practical bottleneck helps explain the boundary of many automatic pipelines. Existing methods have established strong results for given-shape layout, but they typically assume that the target contour is already available. Shape-conditioned approaches such as Calligraphic Packing [XK07] rely on explicit containers and

[†] Corresponding author: Pengfei Xu (xupengfei.cg@gmail.com)

do not address automatic shape acquisition. Legible Compact Calligrams [ZCR*16] improve the readability–fidelity balance, but still depend on manually selected input images and their geometry. Decoupled pipelines [BSC*25] that generate an image first and overlay text afterward often cause letters to concentrate in major body regions and fail to stably cover thin branches and boundary details. Consequently, the expensive front-end stage remains largely unaddressed, and the missing piece is not merely a stronger optimizer, but a unified framework that jointly supports semantic shape formation and letter-level structural writability.

To close this gap, we study **text-only calligram generation without input images**. The system takes only a target word and a semantic prompt, without a manually provided contour, and directly generates a word-level structure. This shifts the problem from layout inside a fixed container to semantic-to-structure generation under readability constraints.

A second challenge is supervision. There is no dedicated large-scale dataset built around skeleton parameter optimization targets for calligrams. Training an additional learned guidance module in this space is costly, and the valid deformation range of the same letter can change across words, such as the letter *c* in *camel* and *crocodile*. Existing readability signals are often defined at the image level, OCR level, or user-study level, which are useful for evaluation but difficult to convert into stable optimization gradients in the skeleton parameter space. This makes a single learned guidance rule in the skeleton parameter space hard to define and generalize.

Our main view is that, with current training limitations, Variational Score Distillation (VSD) [WLW*23] is more reliable for predicting occupancy regions (where letters should be placed) than for producing directly readable stroke topology (how strokes connect). Based on this observation, we use a region-first strategy. We first generate a semantic occupancy prior and then convert it into per-letter regions for later glyph filling. This strategy turns a strongly coupled joint problem into a conditional geometric problem, which lowers optimization complexity and reduces instability transfer across stages.

In this paper, we present a training-free text-only pipeline for calligram generation. The method combines semantic occupancy prior formation with later glyph realization and uses hierarchical gradient projection to improve the stability of the optimization stage.

The main contributions of this paper are as follows. First, we formulate text-only calligram generation without an input contour, where semantic shape support must be inferred from text rather than provided in advance. Second, we propose a training-free two-stage framework in which hierarchical gradient projection stabilizes semantic occupancy prior formation under diffusion guidance. Third, we develop a readability-constrained geometric realization stage that converts the occupancy prior into fillable letter regions through lightweight geometric processing.

2. Related Work

Recent studies have advanced calligram-related tasks under different assumptions. Some assume a given target shape. Some focus on concept-driven glyph editing. Some study font generation. Others address large-scale text layout. For clarity, we review them by

target assumptions, structural scope, and geometric control. This view matches our setting: whole-word calligram generation in an image-free setting.

2.1. Shape-Conditioned Calligram and Image-Guided Text Composition

Shape-conditioned methods are the most established direction in automatic calligram generation. Their common assumption is that the target contour is known. Calligraphic Packing [XK07] divides a container region and assigns letters to local subregions, giving strong contour-filling results. This design is effective because its objectives are explicit and its constraints are geometric. Its limitation is equally clear: it solves layout *inside* a given shape, but does not generate that shape from semantic input.

Legible Compact Calligrams [ZCR*16] improve readability by combining contour skeletons, local protrusion handling, and letter-anchor matching in a hierarchical pipeline. A key lesson is that readability should be optimized together with geometric fitting, not only in post-processing. However, the method still needs external target geometry, so it cannot decide whole-word structure on its own in an image-free setting.

Related image-guided typography and composition methods, including ornamental typeface synthesis [ZWXL17], word paintings [ZYY*22], and ShapeWordle [WCZ*20], also show that text can serve as a strong visual primitive. In most cases, global structure is controlled by an external image or shape, while text is aligned and filled locally. These works offer useful geometric operations, but they do not directly solve image-free semantic structure generation. In short, shape-conditioned calligram layout is mature, but it still assumes given target geometry.

2.2. Semantic Typography and Word-as-Image

With diffusion models and text-image prior knowledge, recent work has shifted from fitting predefined contours to making text itself carry semantic meaning. Word-as-Image [IVH*23] is a representative example. It injects concept cues into letterforms, enabling local correspondence between glyph shape and concept. Its key contribution is to reduce dependence on external contours and show that semantic-driven glyph editing is feasible.

At the task level, these methods mainly target single letters or short character groups. This is closer to concept-driven glyph editing than to full word-shape construction. Put simply, they show how one character can reflect a concept, but not how a full word can become a coherent figurative object with readable order.

Later diffusion-based artistic typography methods, such as DS-Fusion [TWMZ23] and Anything to Glyph [WWL*23], improve style richness and material appearance, but still focus on pixel-space stylization and often rely on extra shape cues or layout priors. More general multimodal generative models, such as GPT Image 2.0 [Ope26], show a similar limit: they can usually preserve either global contour or local letter readability, but not both. Typical outputs are semantic images built from repeated copies of one word, or non-semantic structures formed by only a few letters. Thus, a gap remains for strict whole-word calligram generation.

Khattat [HEH*25] combines semantic prompting, font selection, and text-recognition readability constraints in one framework, marking an important step for multi-character semantic typography. Related work such as FontCLIP [TSQ*24] supports this direction. The strength of this line is joint optimization over semantics, style, and legibility. Still, it targets semantic stylization and local readability improvement rather than full figurative whole-word structure in an image-free setting. For our task, it offers useful supervision, but not a direct structural solver.

2.3. Font Generation, Glyph Modeling, and Layout Synthesis

Font and glyph generation methods mainly study how to produce high-quality, editable glyphs. Their main goal is not to induce word-level figurative structure. DeepVecFont [WL21], VecFontSDF [XXL23], and VecFusion [TLA*24] improve vector glyph quality through joint image-vector representation, signed-distance-shape modeling, and diffusion generation. These methods provide strong foundations for representation quality, style consistency, and editability.

Their typical scope remains in font space, such as reconstruction, transfer, and stylization, rather than whole-word object formation. High local glyph quality does not automatically produce valid word-level structure. This suggests that local glyph modeling and word-level structural optimization should be treated as related but separate layers.

Related work on text layout synthesis, such as text-logo layout [WPL*22], focuses on character scale, affine transforms, line breaking, and global composition balance. It usefully distinguishes local glyph quality from word-level organization, which is directly relevant to whole-word structure.

2.4. Large-Scale Text Layout and Word Clouds

Word-cloud and large-scale text layout research has long studied the placement of many text units within bounded regions. Core concerns include collision avoidance, density control, and readability balance. Classical work [KL07] analyzes 2D layout cost and space use. Later studies add temporal stability via context-preserving dynamic word clouds [CWL*10], interaction constraints via ManiWordle [KLKS10], and multi-attribute organization via FacetClouds [WSK*13]. This line offers reusable layout tools, but also shows that layout is already strongly constrained even when shape is known.

Word-cloud settings are fundamentally different from calligrams. Word clouds usually treat words or short phrases as independent units and mainly optimize position, scale, and orientation. Whole-word calligrams require letter-level continuity and preserve reading order. Their shape constraints are also usually external, for example, in shape-bounded ShapeWordle settings [WCZ*20]. Thus, word-cloud methods are better treated as layout toolkits than as direct solutions for image-free calligram generation.

3. Prior Knowledge

3.1. VSD Semantic Prior

Score Distillation Sampling (SDS) uses a pretrained diffusion model as a prior and performs conditional generation by differentiable parameter optimization. The main idea is to avoid training a new generator. Instead, SDS uses denoising score signals from the diffusion model to provide semantic gradient constraints on the target parameters. Variational Score Distillation (VSD) can be viewed as a variational extension of SDS. SDS treats the optimization target as a single deterministic point, while VSD lifts it to a random variable over a distribution and improves gradient stability and diversity through a distribution-level variational view [PJB23, WLW*23].

Let the optimizable parameters be Θ , and let $x(\Theta)$ be the observation produced by parameterized rendering. At noise timestep t , a common SDS gradient form is

$$\nabla_{\Theta} \mathcal{L}_{\text{SDS}} \propto \mathbb{E}_{t, \epsilon} \left[w(t) (\hat{\epsilon}_{\phi}(x_t, y, t) - \epsilon) \frac{\partial x}{\partial \Theta} \right], \quad (1)$$

where x_t is the noisy observation at timestep t , ϵ is the sampled Gaussian noise, y is the text condition, and $\hat{\epsilon}_{\phi}$ is predicted by the frozen diffusion model. On top of this, VSD extends point-estimate optimization to distribution-level variational matching. This helps reduce oversaturation, premature convergence, and instability that can appear in single-point SDS optimization.

In our method, VSD provides semantic guidance for the current glyph structure. Specifically, the glyph parameters Θ are rendered into an occupancy image $x(\Theta)$, and the diffusion prior then produces semantic gradients $\nabla_{\Theta} \mathcal{L}_{\text{VSD}}$ that guide the current structure toward the target semantic concept.

Overall, VSD supplies semantic gradients, while geometric parameterization keeps the optimization structured and controllable.

3.2. Voronoi Partition

Voronoi partition is a standard geometric tool for region assignment and boundary construction. Given a set of sample points $\{\mathbf{s}_k\}_{k=1}^K$ in the plane, the Voronoi cell of the k -th point is defined as

$$V_k = \{\mathbf{x} \in \mathbb{R}^2 \mid d(\mathbf{x}, \mathbf{s}_k) \leq d(\mathbf{x}, \mathbf{s}_j), \forall j \neq k\}, \quad (2)$$

where $\mathbb{R}^2$ is the 2D plane and $d(\mathbf{x}, \mathbf{s}_k)$ is the Euclidean distance between points $\mathbf{x}$ and $\mathbf{s}_k$. In other words, V_k contains all points that are closer to $\mathbf{s}_k$ than to any other point. This construction divides space into non-overlapping regions according to nearest-site distance.

In graphics and geometry processing, Voronoi partition is often used to separate neighboring structures, assign local support regions, and construct boundary-aware constraints. When the sites are sampled from skeletons or centerlines, the induced partition gives a natural way to decide which spatial area belongs to which structure. The boundaries between Voronoi cells can also be interpreted as separating lines that preserve local detail and reduce unwanted merging between nearby components. In our method, Voronoi partition is used as a boundary-aware spatial separator during later geometric realization.

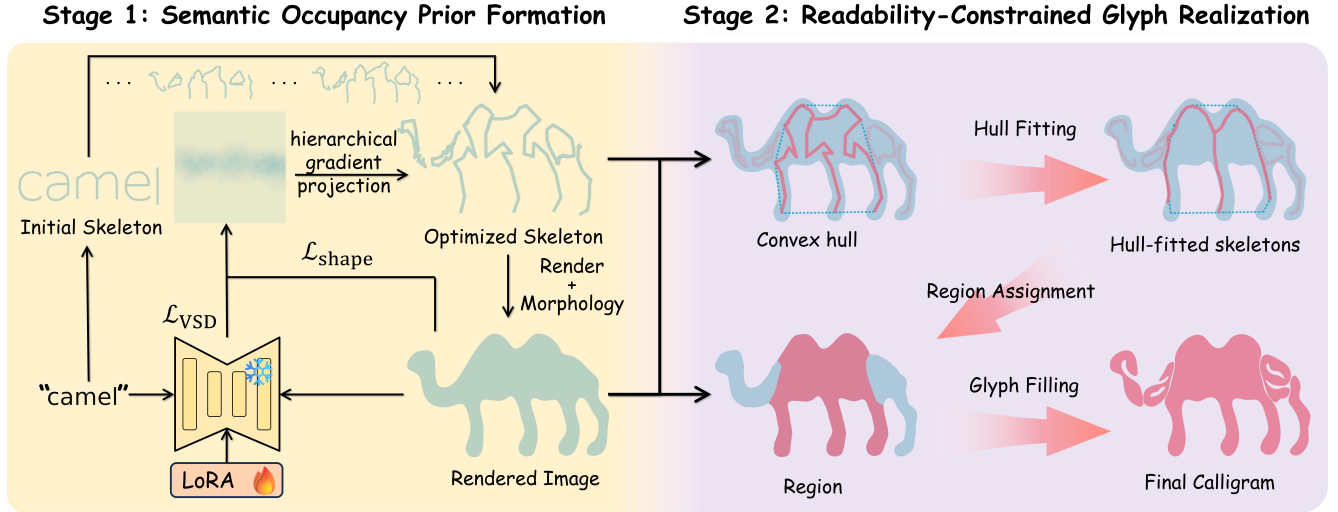


Figure 2: Overview of our two-stage method. In Stage 1, the input word is converted into parameterized letters and optimized under VSD semantic guidance, shape regularization, and hierarchical gradient projection to produce a semantic occupancy prior. In Stage 2, the resulting occupancy mask and skeleton are converted into a readable final calligram through convex-hull extraction, hull-guided skeleton fitting, region assignment, and glyph filling.

4. Method

This method is a training-free, two-stage pipeline for text-only calligram generation. It aims to form a figurative word-level structure while keeping letter readability and geometric feasibility. The first stage, Semantic Occupancy Prior Formation, builds a semantic occupancy prior from parameterized letters under diffusion guidance and lightweight shape regularization. The second stage, Readability-Constrained Glyph Realization, converts this occupancy prior into per-letter regions and readable glyphs through lightweight geometric processing, including convex-hull extraction, hull-guided skeleton fitting, region assignment, and Voronoi-assisted filling. As shown in Fig. 2, the two stages separate semantic structure formation from topology-sensitive glyph realization.

Skeleton-based parameterization provides a compact deformation space that preserves letter topology more reliably under large semantic motion. In our setting, Bezier-based control-point deformation was more sensitive to hyperparameter choice and more likely to damage recognizable letter structure. The skeleton representation therefore provides a more stable balance between semantic mobility and topological stability. A two-stage design is used because semantic formation and readability preservation act on competing geometric tendencies. Stage 1 first forms a semantically plausible occupancy prior, and Stage 2 then restores letter structure under that support. Directly coupling these two goals in a single optimization loop was less stable in our experiments, so the final pipeline uses a single semantic-then-realization pass.

4.1. Semantic Occupancy Prior Formation

This stage takes a word and a semantic prompt and optimizes the letters toward a recognizable semantic occupancy prior. The goal is

not to directly produce readable final glyphs, but to obtain a stable occupancy prior that captures where letters should be placed.

We first convert the input word into parameterized letters and initialize an optimizable state $\Theta = \{\mathbf{T}, \mathbf{S}, \mathbf{R}, \mathbf{J}\}$, where $\mathbf{T}$, $\mathbf{S}$, and $\mathbf{R}$ denote per-letter translation, scale, and rotation, and $\mathbf{J}$ denotes local skeleton coordinates. Each glyph is rendered from a skeleton with fixed radii, where each skeleton segment is given a fixed stroke width, and its occupied area is approximated by a union of capsule primitives, that is, short rounded stroke pieces.

From this state, differentiable rendering produces a soft mask $\alpha \in [0, 1]^{H \times W}$, where $H \times W$ is the image size, and each pixel value indicates how strongly that pixel is covered. We then apply shape-growing and shape-shrinking operations to obtain a more semantically usable mask. Specifically, we use LogSumExp-based soft pooling, a smooth approximation of max/min pooling, to define a soft close operation, which first expands and then shrinks the mask,

$$\text{Close}_r^\tau(\alpha) = \epsilon_r^\tau(\delta_r^\tau(\alpha)). \quad (3)$$

Here δ_r^τ and ϵ_r^τ denote soft dilation and soft erosion with radius r and temperature τ , respectively. Dilation means expanding the mask outward, erosion means shrinking it inward, r sets the operation range, and τ controls how sharp or smooth the approximation is. The role of the soft morphological close is to introduce weak pseudo-connections in local gaps, so semantic guidance does not need to force true letter overlap in order to form a coherent figure.

To avoid false boundary connections, we use a boundary-band mask $B \in \{0, 1\}^{H \times W}$, where 1 marks a narrow image-border band in which closing is suppressed, and apply a border-safe close. We then run reconstruction-style hole filling from boundary markers:

$$m^{(t+1)} = \min(\delta_r(m^{(t)}), \beta), \quad \beta = 1 - \alpha_{\text{safe}}, \quad (4)$$

and obtain $\alpha_{\text{fill}} = 1 - m^{(*)}$ after convergence, where α_{safe} is the border-safe mask after suppressing closing near the image boundary. Here, δ_r denotes standard binary dilation in the reconstruction step, that is, one step of binary mask expansion with radius r . β is the background area that filling is allowed to enter, and $m^{(*)}$ is the final marker after the iteration stops.

The optimized objective is written as

$$\mathcal{L}_{\text{total}} = w_{\text{sem}}\mathcal{L}_{\text{VSD}} + w_{\text{shape}}\mathcal{L}_{\text{shape}}. \quad (5)$$

Here, $\mathcal{L}_{\text{VSD}}$ is the semantic guidance from the diffusion model, applied to the rendered and morphologically processed mask. On the rendered mask, we use two lightweight geometric terms. Let Ω denote the full pixel grid of the rendered occupancy image. Let $\alpha_k(\mathbf{x})$ be the rendered body mask value of the k -th letter at pixel $\mathbf{x}$, and let $\alpha(\mathbf{x})$ denote the full rendered mask. We define

$$\mathcal{L}_{\text{overlap}} = \frac{1}{|\Omega|} \sum_{\mathbf{x} \in \Omega} \text{ReLU} \left(\sum_{k=1}^K \alpha_k(\mathbf{x}) - 1 \right), \quad (6)$$

$$\mathcal{L}_{\text{single}} = \frac{1}{|\Omega|} \sum_{\mathbf{x} \in \Omega} \alpha(\mathbf{x}) (1 - c(\mathbf{x})), \quad (7)$$

where $c(\mathbf{x})$ is a soft connectivity value obtained by iterative masked dilation from seed regions, i.e., repeated masked expansion from initial seed pixels; it measures whether pixel $\mathbf{x}$ belongs to the connected main component. $\mathcal{L}_{\text{overlap}}$ is computed on the rendered letters and discourages excessive inter-letter interference. $\mathcal{L}_{\text{single}}$ encourages all letters to contribute to the main semantic body, preventing VSD from converging to a degenerate solution in which one enlarged letter explains most of the shape while the remaining letters shrink into small isolated blobs. The shape regularizer is then defined as

$$\mathcal{L}_{\text{shape}} = \lambda_{\text{overlap}}\mathcal{L}_{\text{overlap}} + \lambda_{\text{single}}\mathcal{L}_{\text{single}}. \quad (8)$$

Together, these terms keep the semantic optimization from collapsing into unreadable or heavily unbalanced letter layouts.

The remaining question is how to update these variables without collapsing the layout. A common strategy in packing or collage optimization is to impose explicit lower and upper bounds on per-element scale [SZYX26], so that no element becomes too large or too small. However, this strategy is too restrictive for compact calligrams. To form a semantic figure, some letters may need to become noticeably larger than others, such as the letter m in a camel-shaped word. Hard scale bounds can therefore block useful semantic deformation. Without such control, optimization over coarse global deformation and fine local deformation often converges to degenerate shortcuts, where a few letters dominate the main body, and the others nearly disappear. We therefore use a hierarchical gradient projection as a soft alternative to hard scale bounds. As illustrated in Fig. 3, the update is organized in a coarse-to-fine order instead of directly injecting the full gradient into skeleton joints.

This design avoids two unstable extremes. A manually staged schedule for low-dimensional affine deformation and high-dimensional skeleton deformation is hard to tune. Directly injecting the full gradient into skeleton joints makes the update sensitive to high-frequency noise, while smoothing such joint-space updates may also suppress useful coarse semantic motion. We therefore let

low-dimensional affine deformation absorb the large structured part of the gradient first, and pass only the remaining signal to the local skeleton variables. The subsequent smoothing step is then applied only to this residual local update, which reduces information loss and improves stability. Radius freezing, meaning that stroke width is kept fixed, is maintained throughout this stage.

Let $\mathbf{G} = -\nabla_{\mathbf{J}}\mathcal{L}_{\text{total}}$ be the back-propagated force field on skeleton joints. We first project this gradient onto a low-dimensional affine subspace that includes translation, isotropic scaling, rotation, and anisotropic scaling,

$$\begin{aligned} \mathbf{a}_{\text{aff}} &= \arg \min_{\mathbf{a}} \|\mathbf{G} - \mathbf{B}_{\text{aff}}\mathbf{a}\|_2^2 + \lambda_{\text{aff}}\|\mathbf{a}\|_2^2, \\ \mathbf{G}^{(\text{loc})} &= \mathbf{G} - \mathbf{B}_{\text{aff}}\mathbf{a}_{\text{aff}}. \end{aligned} \quad (9)$$

where $\mathbf{B}_{\text{aff}}$ denotes the basis matrix of the low-dimensional affine deformation subspace, meaning the set of allowed coarse deformation directions, and $\mathbf{a}_{\text{aff}}$ contains the corresponding affine coefficients. We then decompose the remaining signal into a graph-smooth skeleton component and a high-frequency residual on the skeleton graph $\mathcal{T} = (\mathbf{J}, \mathbf{E})$, where $\mathbf{E}$ is the set of skeleton edges connecting neighboring joints.

$$\begin{aligned} \mathbf{G}_{\text{skel}} &= (\mathbf{I} + \lambda_{\text{skel}}\mathbf{L})^{-1}\mathbf{G}^{(\text{loc})}, \\ \mathbf{G}_{\text{res}} &= \mathbf{G}^{(\text{loc})} - \mathbf{G}_{\text{skel}}. \end{aligned} \quad (10)$$

where $\mathbf{L}$ is the graph Laplacian of $\mathcal{T}$, a standard matrix that encodes which skeleton joints are neighbors and therefore favors smooth changes along the skeleton. The residual term is computed only to make the decomposition explicit, and is discarded in the final update to avoid unstable high-frequency local distortion. The final update is written as

$$\Delta\Theta_{\text{aff}} = -\eta_{\text{aff}}\mathbf{a}_{\text{aff}}, \quad (11)$$

$$\Delta\mathbf{J} = -\eta_{\text{skel}}\mathbf{G}_{\text{skel}}, \quad (12)$$

where $\Delta\Theta_{\text{aff}}$ updates the affine variables in Θ and $\Delta\mathbf{J}$ updates the skeleton coordinates, that is, the point locations on the letter centerline. Here η_{aff} and η_{skel} are the step sizes for the affine and skeleton updates. The full state update is therefore

$$\Theta_{t+1} = \{\mathbf{T}_{t+1}, \mathbf{S}_{t+1}, \mathbf{R}_{t+1}, \mathbf{J}_{t+1}\}. \quad (13)$$

This hierarchical projection reduces tuning burden while preserving the freedom needed for semantic deformation. In practice, large semantic motion is first absorbed by low-dimensional affine transforms, while local skeleton deformation is used only when necessary. The output of Stage 1 is therefore a semantic occupancy prior rather than a directly readable final glyph layout.

4.2. Readability-Constrained Glyph Realization

The output of Stage 1 provides a semantic occupancy prior, but it is not yet a readable final calligram. This stage is a lightweight geometric post-processing step that converts the Stage 1 occupancy mask and skeleton into per-letter regions and readable letter shapes. The process follows four steps: convex-hull extraction, hull-guided skeleton fitting, region assignment, and Voronoi-assisted filling. Since this stage is not the main contribution, we keep it lightweight and training-free.

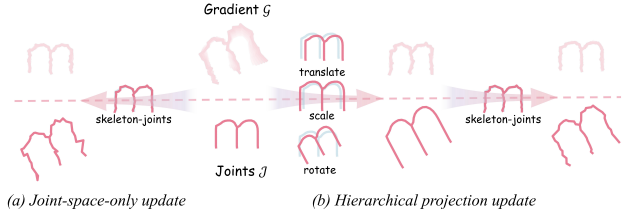


Figure 3: Illustration of joint-space-only update and hierarchical projection update on the skeleton of the letter *m*. The pale blurry shape visualizes the current gradient-induced motion field, and the pink curve shows the current skeleton. In (a), the full gradient is directly injected into joint space, so coarse displacement and local deformation remain entangled throughout the update. In (b), the update flow is split into stages: low-dimensional affine motion first absorbs coarse semantic displacement, and only the residual local signal is passed to the skeleton variables. This ordering makes the optimization trajectory easier to follow and reduces unstable shortcuts dominated by coarse deformation.

We begin by extracting a convex hull, the smallest convex region that covers the input skeleton support, of each letter. This hull provides a coarse outer shape that preserves how far the letter can extend inside the occupancy mask. We then fit the input skeleton into this hull to obtain a more readable skeleton configuration. The purpose of this step is not to exactly recover the original glyph, but to restore a clearer letter structure before final filling.

Let the convex hull of the k -th letter be Π_k and let $\mathcal{S}_k$ denote its fitted skeleton, that is, the centerline used to rebuild that letter. The fitted skeleton is constrained to remain geometrically compatible with Π_k , so that it follows the allocated semantic mass while avoiding excessive distortion.

Once fitted skeletons are available, we assign a region to each letter. The per-letter region is determined jointly by the Stage 1 occupancy mask and the fitted skeleton layout. Intuitively, the mask provides the available semantic support, while the fitted skeleton determines which part of that support should belong to each letter.

Formally, let $M \subset \mathbb{Z}^2$ be the Stage 1 occupancy mask, where $\mathbb{Z}^2$ denotes the 2D integer pixel grid. For each pixel $\mathbf{x} \in M$, we assign it to the nearest fitted skeleton, where $d(\mathbf{x}, \mathcal{S}_k)$ is the Euclidean distance from pixel $\mathbf{x}$ to skeleton $\mathcal{S}_k$,

$$L(\mathbf{x}) = \arg \min_k d(\mathbf{x}, \mathcal{S}_k), \quad (14)$$

and obtain per-letter regions $M_k = \{\mathbf{x} \in M \mid L(\mathbf{x}) = k\}$. These per-letter regions serve as the support for final glyph filling.

With a fitted skeleton $\mathcal{S}_k$ and its region M_k , the remaining problem is to decide how strongly the skeleton should be thickened so that the letter fills the region while remaining readable. We therefore generate candidate fillings by skeleton-radius dilation, meaning that we thicken the skeleton by a radius r ,

$$G_k(r) = \text{Dilate}(\mathcal{S}_k, r) \cap M_k, \quad (15)$$

and use TrOCR [LLC*23], a text recognition model, as an auxiliary

scorer to select a radius that balances readability and region coverage. Here, $\text{Dilate}(\mathcal{S}_k, r)$ means the set of pixels whose distance to skeleton $\mathcal{S}_k$ is at most r .

Readability score alone is not sufficient. For some letters, a very small radius keeps the skeleton recognizable and therefore receives a relatively high score, but leaves the region underfilled. Conversely, a large radius may fill the region well but erase thin structures and lower readability. To reduce this failure mode, we introduce a Voronoi-based constraint for letters whose local details are easily removed by direct thickening.

Specifically, we compute a Voronoi partition from the fitted skeletons, meaning that each pixel is assigned to its nearest skeleton, and use the induced boundary lines as local separators. After thickening these boundary lines into a narrow protection band, meaning a thin forbidden strip near letter boundaries, we subtract the band from the target region before final filling. Denoting this protection band by C_k , the final realization is written as

$$\hat{G}_k = (\text{Dilate}(\mathcal{S}_k, r_k) \cap M_k) \setminus C_k, \quad (16)$$

where r_k is the selected filling radius for the k -th letter, that is, the final stroke thickness used for that letter. This simple refinement preserves local details and inter-letter separation, while still allowing each glyph to fill its assigned region as much as possible.

In summary, the method consists of two parts. Semantic Occupancy Prior Formation generates stable occupancy through rendered-mask semantic supervision, lightweight shape regularization, and hierarchical gradient projection, while keeping stroke width fixed during optimization. Readability-Constrained Glyph Realization converts this occupancy prior into readable glyphs through convex-hull extraction, hull-guided skeleton fitting, region assignment, and Voronoi-assisted filling. This design separates semantic occupancy formation from readable glyph realization, while keeping the second stage lightweight and geometric.

5. Experiments

This section evaluates both the optimization behavior of the proposed framework and the quality of the generated calligrams. We first study how the proportion of early low-dimensional affine updates affects optimization behavior, where low-dimensional affine updates refer to coarse global motion such as translation, rotation, and scaling. We then test whether hierarchical gradient projection improves semantic alignment while avoiding collapse patterns caused by overly strong coarse deformation. Finally, we report OCR-based readability results and representative qualitative ablations. Unless otherwise noted, all table values in this section are reported in units of $\times 10^{-2}$.

For quantitative evaluation, we use OpenAI CLIP ViT-B/32 [RKH*21]. CLIPScore is averaged over 10 different prompts and 15 different random seeds for each compared setting. We compare three update rules. **Direct joint update** applies the full gradient directly to the skeleton joints. **Smoothed joint update** first computes a graph-smoothed joint update $\mathbf{G}_{\text{smooth}} = (\mathbf{I} + \lambda \mathbf{L})^{-1} \mathbf{G}$ and then applies it to the joints, where $\mathbf{L}$ is the skeleton graph Laplacian and λ is the smoothing weight. **Hierarchical update** first lets low-dimensional affine deformation absorb the structured part

Table 1: Runtime and platform settings used in the experiments.

Item	Value
Stage 1 steps	2000
Semantic resolution	96×96
VSD model	DeepFloyd-IF Stage I [DS23]
GPU	NVIDIA Quadro P6000
GPU memory	24 GB
Stage 1 runtime	about 6 min per letter
Stage 2 runtime	within 5 min

Table 2: Core hyperparameters used in the experiments.

Parameter	Value
Soft-close temperature τ	0.6
Word-level close radius r	15
Semantic weight w_{sem}	120
Single-component weight λ_{single}	12
Overlap weight λ_{overlap}	1
Smoothing weight λ	0.5

of the gradient, and then applies the same smoothed joint update only to the remaining local signal.

5.1. Coarse-Deformation Ratio Analysis

We study how the ratio of early low-dimensional affine updates affects optimization behavior. Table 3 summarizes the four schedule settings and their quantitative results. We use CLIPScore as the primary metric. We also report *max region ratio* and *region variance* only as descriptive indicators of how concentrated or imbalanced the final letter areas become under different schedules.

The main trend is reflected by CLIPScore. Under the staged affine schedule, earlier participation of high-dimensional skeleton deformation consistently leads to better semantic alignment. This suggests that relying too much on early low-dimensional affine motion encourages coarse shortcuts, while earlier skeleton deformation gives the optimization more freedom to match the target concept. The max region ratio and region variance provide only a descriptive view of how concentrated the letter areas become under different schedules. They are not used here as direct quality criteria. Taken together, these results show that the main difficulty is not simply whether coarse motion is enabled, but how much of the optimization budget is assigned to it at early stages. This also explains why manually staged schedules are difficult to tune in practice.

5.2. Hierarchical Projection Analysis

We compare the three update rules defined above under the same overall budget: **Direct joint update**, **Smoothed joint update**, and **Hierarchical update**. The first is equivalent to the **None** setting in Table 3; the two names describe the same method from two different views. CLIPScore remains the primary metric, while max ratio and region variance are used only to describe size concentration and area imbalance.

Table 3: Effect of the early low-dimensional affine update ratio. None corresponds to 0% early affine steps, while Low, Medium, and High are stochastic schedules centered around approximately 25%, 50%, and 75% of the optimization process. All values are reported in units of $\times 10^{-2}$.

Early low-dim. update ratio	CLIPScore $\uparrow$	Max ratio	Reg. var.
High	33.314	42.446	1.735
Medium	33.566	39.232	1.285
Low	33.868	33.178	0.994
None	34.227	32.530	0.937

Table 4: Main comparison of optimization strategies. The full method is expected to provide the best semantic alignment while avoiding dominant-letter collapse. All values are reported in units of $\times 10^{-2}$.

Method	CLIPScore $\uparrow$	Max ratio	Reg. var.
Direct joint update	34.227	32.530	0.937
Smoothed joint update	34.288	31.581	0.742
Hierarchical update	34.314	36.845	1.593

The hierarchical update achieves the highest CLIPScore among the compared settings. This result indicates that explicitly separating low-dimensional affine motion from high-dimensional skeleton deformation improves semantic alignment more effectively than either direct joint update or smoothed joint update alone. Notably, the hierarchical update does not aim to minimize size imbalance. Its max ratio and region variance can be higher than those of direct joint update, because it allows more flexible letter-size adaptation. The advantage is that this flexibility is introduced in a structured way, so semantic alignment improves without relying on unstable joint-wise noise.

Overall, the experiments support two conclusions: high early low-dimensional update ratios favor dominant-letter shortcuts, while the hierarchical update provides the best semantic alignment with structured and flexible letter-size adaptation.

5.3. OCR-Based Readability Evaluation

Since calligram quality depends on both semantic shape and character recognizability, we complement the semantic evaluation with an OCR-based readability experiment on cropped single-character images. Although TrOCR is used in the local candidate-selection step of Stage 2, the readability evaluation here is conducted with separate OCR backbones so that the final analysis remains independent of that selection process. Because the inputs are already character-level crops, we use character recognizers rather than full text-detection pipelines. We report results from two representative backbones, CRNN [SBY17] and PARSeq [BA22], which provide two recognition views on the generated glyphs.

For each generated word, we split the final calligram into individual letter images and evaluate whether each crop is still recog-

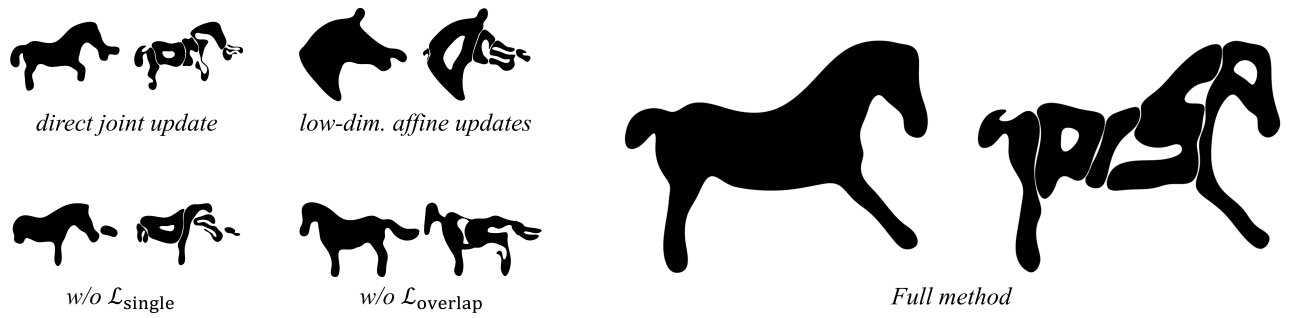


Figure 4: Representative qualitative ablations for Stage 1 modules. From left to right, we show the effects of prolonged early affine updates, skeleton-only deformation, removing L_{single} , and removing L_{overlap} . These examples illustrate how the individual components affect dominant-letter collapse, semantic motion, disconnected small letters, and excessive inter-letter interference.

Table 5: OCR-based readability evaluation on cropped single-character images. PARSeq and CRNN provide two independent recognition views. For CRNN, we also report the mean ground-truth posterior. “Berio et al.” and “Zou et al.” denote results reproduced from [BSC*25, ZCR*16], while “Fleming” denotes the expert-designed examples shown in Fig. 5.

Group	PARSeq Acc. $\uparrow$	CRNN Acc. $\uparrow$	Mean $P_{\text{gt}} \uparrow$
Berio et al.	38.89	22.22	0.135
Fleming	46.00	50.00	0.244
Zou et al.	56.00	58.00	0.286
Ours	44.00	38.00	0.220

nized as its ground-truth character. We report character-level top-1 accuracy for both recognizers and, for CRNN, also the mean ground-truth posterior $P(\text{gt} | x)$. OCR is used here as an auxiliary readability measure rather than a complete perceptual metric, since our method also explicitly targets smooth boundaries and the suppression of spurious branches.

As shown in Table 5, the overall trend is consistent across both recognizers. Zou et al. achieve the highest OCR accuracy, while Berio et al. give the lowest OCR scores among the compared methods. Ours performs better than Berio et al. under the same no-mask setting and remains comparable to the expert-designed Fleming examples on PARSeq and on the CRNN mean ground-truth posterior.

Boundary style may partly explain the remaining gap. The Fleming examples are manually designed, and our method uses smooth geometric realization. In contrast, Zou et al. produce sharper contours that contain local discriminative cues easily recognized by OCR models. Accordingly, the OCR results should be interpreted together with boundary quality and branch suppression rather than in isolation. CRNN top-1 accuracy, in particular, is lower for smooth-boundary results than for sharper-contour baselines.

5.4. Qualitative Ablation Study

Fig. 4 provides representative qualitative ablations for the Stage 1 modules. Prolonged early affine updates tend to produce a dominant-letter shortcut, while skeleton-only deformation weakens semantic alignment. Removing L_{single} allows detached small components, and removing L_{overlap} increases inter-letter interference. Taken together, these ablations show that the full Stage 1 design balances coarse semantic motion, local structural adaptation, whole-word connectivity, and inter-letter separation.

6. Comparison and Results

6.1. Comparison with Prior Methods

We compare Telligram with contour-conditioned calligram methods, including the method of Zou et al. [ZCR*16] and the B-spline-based abstraction method of Berio et al. [BSC*25]. These methods require a given image or contour to define the target outer shape, whereas our method starts from text alone. As shown in Fig. 5, this difference lets our method form the semantic outer structure and letter placement jointly, so the resulting branches are more naturally supported by glyph mass instead of being stretched only to satisfy a prescribed outline. We do not report direct quantitative comparison with these two methods because neither paper provides official code, so in practice we can only perform qualitative comparison using the images reported in their papers.

A second comparison shows the role of skeleton-guided semantic prior formation. Berio et al. [BSC*25] directly prompt the model with descriptions such as “silhouette of a seagull”. This can produce a semantically recognizable outer contour, but it may also create thin branches that are hard to fill with readable glyphs. As shown in Fig. 6, our method uses letter skeletons as a structural prior when forming the semantic occupancy prior. This makes it easier to preserve semantic contour quality while reducing branches that are too thin to be occupied by letters in a reasonable way.

A third comparison uses matched semantic information, namely the concept “boat”, together with the same word “successful”. As shown in Fig. 7, Zou et al. [ZCR*16] require explicit guiding lines to arrange multiple text lines, whereas our method starts from a

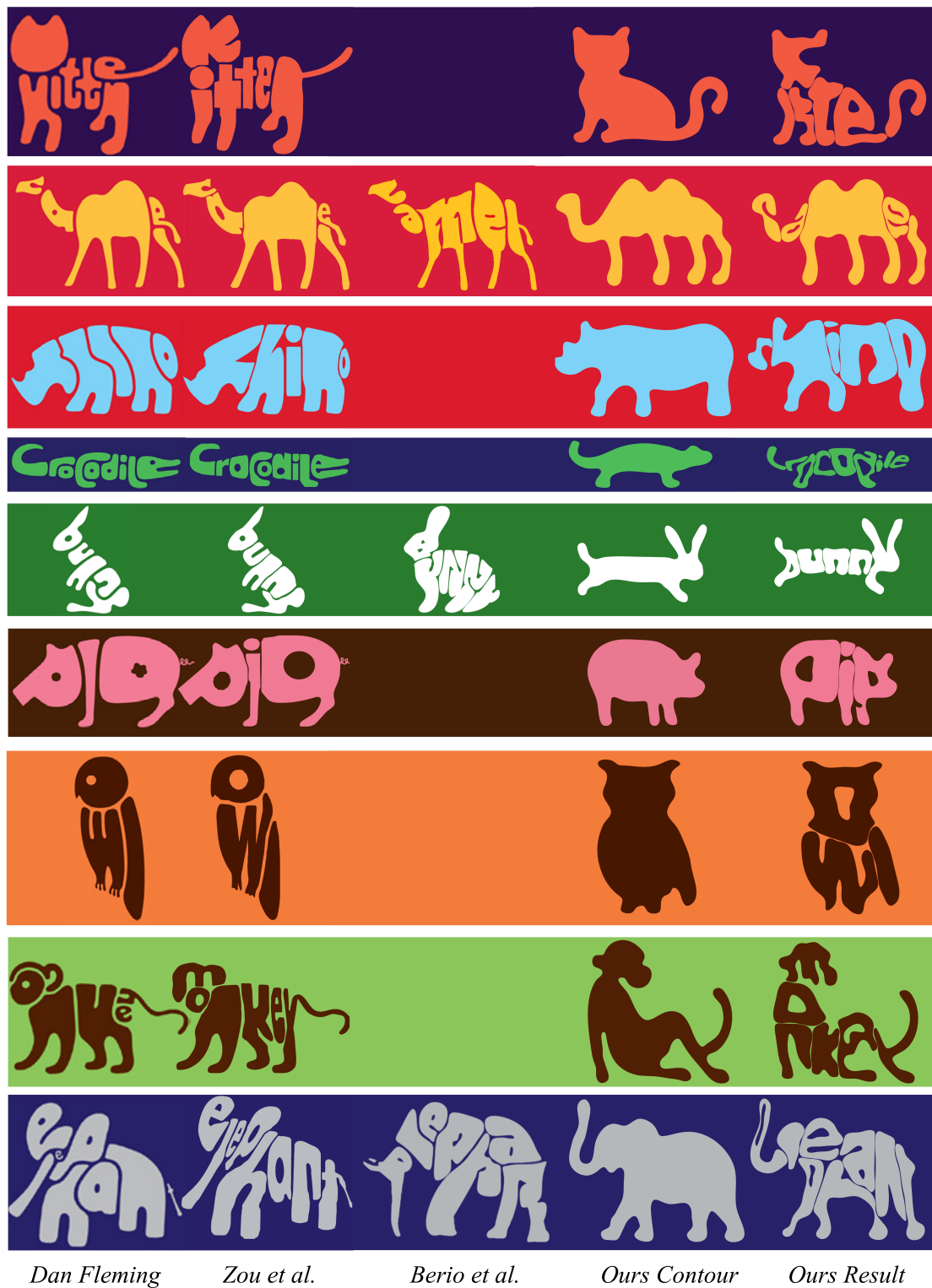


Figure 5: Comparison with methods that generate calligrams from a given image or contour. The human-designed examples by Dan Fleming and the automatic results of Zou et al. are reproduced from Legible Compact Calligrams [ZCR*16], and the B-spline-based results of Berio et al. are from Neural Image Abstraction Using Long Smoothing B-Splines [BSC*25]. The rightmost examples are generated by our method from text alone. Blank entries indicate cases that were not reported in the original source.



Figure 6: Comparison between direct silhouette prompting and skeleton-guided semantic prior formation. The left examples are generated by Berio et al. [BSC*25] from prompts such as “silhouette of a seagull”, and the right examples are produced by our method under the same semantic target.

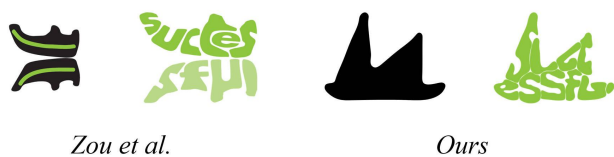


Figure 7: Comparison under matched semantic information, namely the concept “boat”, and the same word “successful”. The left examples are from Zou et al. [ZCR*16], and the right examples are produced by our method. For the Zou et al. example, the leftmost black silhouette is the contour image shown together with their result in the original paper.

multi-line text layout and forms the semantic figure through the same semantic occupancy optimization.

6.2. Additional Qualitative Results

We further show qualitative results on different prompts and word structures. These examples illustrate the visual diversity of the generated calligrams and show that the method can produce recognizable figurative layouts without an input contour image.

Figure 8 shows controlled variations under different random seeds, different line orders, and different words under the same prompt. Here, line-order variation refers to rearranging the reading lines of a multi-line word while keeping the same letter set, for example, by changing the order of the top and bottom lines. Beyond these controlled settings, Fig. 9 demonstrates broader prompt coverage. Across a wide range of semantic prompts, Telligram produces varied calligram layouts without collapsing to a single repeated template.

7. Conclusion

This paper presented Telligram, a training-free two-stage framework for text-only calligram generation. The method first builds a semantic occupancy prior under diffusion guidance and then converts this prior into readable final glyphs through lightweight geometric processing. The experiments showed that hierarchical gradient projection provides better semantic alignment than direct joint

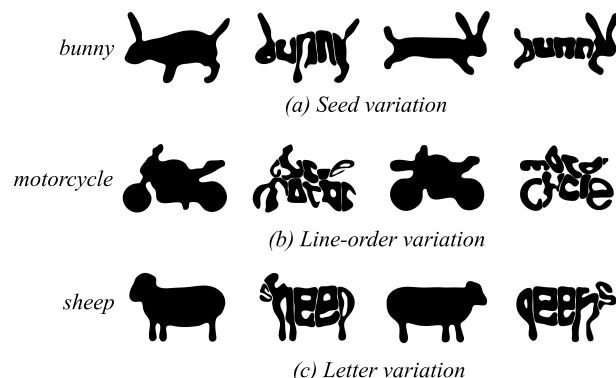


Figure 8: Examples of output diversity. (a) shows results generated from the same prompt, word, and layout under different random seeds. (b) shows line-order variation, namely different line arrangements of the same word such as exchanging the upper and lower line while preserving the same set of letters. (c) shows results of the same letters arranged in different orders under the same prompt.

update or smoothed joint update, while avoiding the heavy tuning burden of manually staged schedules.

The current framework also has clear limitations. Because the representation is based on silhouettes and skeletons with finite thickness, it can lose fine-grained appearance cues and has limited ability to restore letters once severe deformation has already occurred. The current system also uses DeepFloyd guidance together with a low semantic resolution of 96×96 , so different prompts may sometimes produce similar global shapes.

Fig. 10 illustrates three representative failure modes in the current training-free setting. First, silhouette-like outputs may remain semantically related while still being ambiguous under category recognition; for example, the giraffe silhouette may be recognized as *camel* in CLIP zero-shot top-1 prediction. On a balanced test set of 36 target categories with 15 random seeds per category (540 generated samples in total), we evaluate semantic recognition using CLIP zero-shot classification over a combined label space consisting of ImageNet-1k labels and our target labels. Under a top-5 check, this gives a semantic recognition rate of 72.03%. Here, a top-5 semantic match means that the predictions contain either the target category itself or a more specific label with the same target noun, such as *bear* matched by *grey bear*; visually similar but semantically different categories are not counted as matches. This number is used only as a coarse category-level indicator for silhouette-like outputs. Second, hierarchical gradient projection reduces instability caused by entangled coarse and local deformation during Stage 1, but it does not fully prevent failures driven mainly by local identity-damaging skeleton updates. If a letter identity is already heavily weakened before entering Stage 2, the later convex-hull-based restoration may still be insufficient to recover the intended glyph structure, as illustrated by the highlighted region in the giraffe example. Third, even when Stage 2 restores the coarse

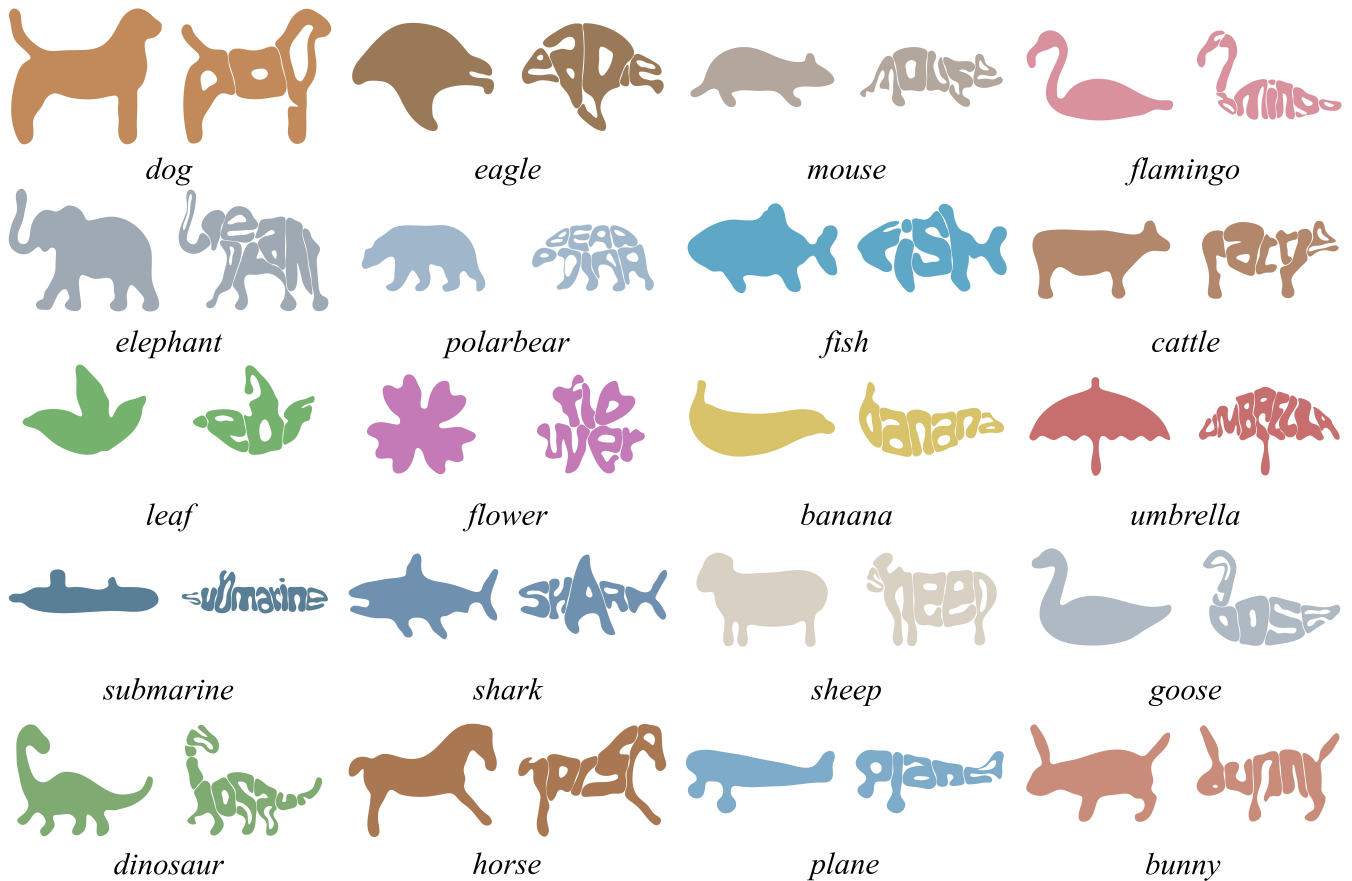


Figure 9: Additional qualitative results of Telligram under diverse prompts. Across different semantic targets, the method produces creative and varied calligram layouts while maintaining recognizable figurative structure.

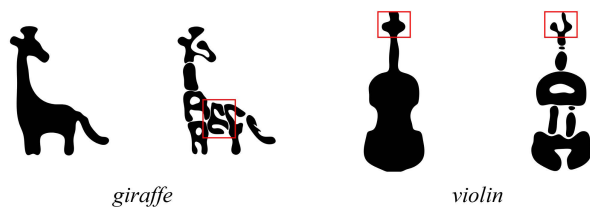


Figure 10: Representative failure cases. Left: the giraffe silhouette remains semantically related but may still be recognized as a nearby category such as camel, and severe intermediate deformation can weaken the intended letter identity, as highlighted in the red box. Right: even when the coarse letter shape is restored, branch-like support may still persist in the final glyph, as highlighted in the v of the violin example.

letter shape, branch-like support may still persist in the final glyph, as in the highlighted v of the violin example.

These limitations are tied in part to our decision to keep the pipeline training-free. A promising future direction is therefore to

relax this constraint and learn a task-specific readability guider or letter-identity discriminator that provides stable identity-preserving gradients already in Stage 1. Such a model could reduce severe intermediate deformation before Stage 2, alleviate later restoration failures, and support a more tightly coupled formulation of semantic shape formation and readability preservation. Another possible extension is to enrich the current geometric hierarchy with additional deformation variables for finer boundary refinement once stable identity-preserving guidance becomes available.

Acknowledgments

We thank the anonymous reviewers for their constructive feedback. This work was supported by the National Natural Science Foundation of China (62472287) and the Natural Science Foundation of Shenzhen City (JCYJ20250604181519025).

References

- [BA22] BAUTISTA D., ATIENZA R.: Scene text recognition with permuted autoregressive sequence models. In *ECCV* (2022), Springer, pp. 178–196. [7](#)

- [BSC*25] BERIO D., STROH M., CALINON S., LEYMARIE F. F., DEUSSEN O., SHAMIR A.: Neural image abstraction using long smoothing b-splines. *ACM TOG* 44, 6 (2025), 1–11. 2, 8, 9, 10
- [CWL*10] CUI W., WU Y., LIU S., WEI F., ZHOU M. X., QU H.: Context preserving dynamic word cloud visualization. In *PacificVis* (2010), pp. 121–128. 3
- [DS23] DEEPFLOYD, STABILITY AI: If-i-m-v1.0 model card. Hugging Face model card for DeepFloyd/IF-I-M-v1.0, 2023. Stage-I pixel diffusion model with frozen T5 text encoder. URL: <https://huggingface.co/DeepFloyd/IF-I-M-v1.0>. 7
- [HEH*25] HUSSEIN A., ELSETOHY A., HADHOUD S., BAKR T., ROHAIM Y., ALKHAMISSI B.: Khattat: Enhancing readability and concept representation of semantic typography. In *ECCV Workshops* (2025), Springer, pp. 278–295. 3
- [IVH*23] ILUZ S., VINKER Y., HERTZ A., BERIO D., COHEN-OR D., SHAMIR A.: Word-as-image for semantic typography. *ACM TOG* 42, 4 (2023), 1–11. 2
- [KL07] KASER O., LEMIRE D.: Tag-cloud drawing: Algorithms for cloud visualization. In *Tagging and Metadata for Social Information Organization (WWW 2007 Workshop)* (2007). 3
- [KLKS10] KOH K., LEE B., KIM B., SEO J.: Maniwordle: Providing flexible control over wordle. *IEEE TVCG* 16, 6 (2010), 1190–1197. 3
- [LLC*23] LI M., LV T., CHEN J., CUI L., LU Y., FLORENCIO D., ZHANG C., LI Z., WEI F.: Trocr: Transformer-based optical character recognition with pre-trained models. In *AAAI* (2023), pp. 13094–13102. 6
- [MBS*11] MAHARIK R., BESSMELTSEV M., SHEFFER A., SHAMIR A., CARR N.: Digital micrography. *ACM TOG* 30, 4 (2011), 100:1–100:12. 1
- [Ope26] OPENAI: Introducing chatgpt images 2.0, 2026. Accessed: 2026-06-09. URL: <https://openai.com/index/introducing-chatgpt-images-2-0/>. 2
- [PJB23] POOLE B., JAIN A., BARRON J. T., MILDENHALL B.: Dreamfusion: Text-to-3d using 2d diffusion. In *ICLR* (2023). 3
- [RKH*21] RADFORD A., KIM J. W., HALLACY C., RAMESH A., GOH G., AGARWAL S., SASTRY G., ASKELL A., MISHKIN P., CLARK J., KRUEGER G., SUTSKEVER I.: Learning transferable visual models from natural language supervision. In *ICML* (2021), vol. 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. 6
- [SBY17] SHI B., BAI X., YAO C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE TPAMI* 39, 11 (2017), 2298–2304. 7
- [SZYX26] SHAO Z., ZHOU J., YANG H., XU P.: Score: Semantic collage by optimizing rendered elements. In *AAAI* (2026), pp. 2038–2046. 5
- [TLA*24] THAMIZHARASAN V., LIU D., AGARWAL S., FISHER M., GHARBI M., WANG O., JACOBSON A., KALOGERAKIS E.: Vecfusion: Vector font generation with diffusion. In *CVPR* (2024), pp. 7943–7952. 3
- [TSQ*24] TATSUKAWA Y., SHEN I., QI A., KOYAMA Y., IGARASHI T., SHAMIR A.: Fontclip: A semantic typography visual-language model for multilingual font applications. *CGF* 43, 2 (2024), e15043. 3
- [TWMZ23] TANVEER M., WANG Y., MAHDAVI-AMIRI A., ZHANG H.: Ds-fusion: Artistic typography via discriminated and stylized diffusion. In *ICCV* (2023), pp. 374–384. 2
- [WCZ*20] WANG Y., CHU X., ZHANG K., BAO C., LI X., ZHANG J., FU C., HURTER C., LEE B., DEUSSEN O.: Shapewordle: Tailoring wordles using shape-aware archimedean spirals. *IEEE TVCG* 26, 1 (2020), 991–1000. 2, 3
- [Wik26] WIKIPEDIA CONTRIBUTORS: Calligram — wikipedia, the free encyclopedia, 2026. Accessed: 2026-05-28. URL: <https://en.wikipedia.org/wiki/Calligram>. 1
- [WL21] WANG Y., LIAN Z.: Deepvecfont: Synthesizing high-quality vector fonts via dual-modality learning. *ACM TOG* 40, 6 (2021). 3
- [WLW*23] WANG Z., LU C., WANG Y., BAO F., LI C., SU H., ZHU J.: Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. In *NeurIPS* (2023). 2, 3
- [WPL*22] WANG Y., PU G., LUO W., WANG Y., XIONG P., KANG H., LIAN Z.: Aesthetic text logo synthesis via content-aware layout inferring. In *CVPR* (2022), pp. 2436–2445. 3
- [WSK*13] WALDNER M., SCHRAMMEL J., KLEIN M., KRISTJANS-DOTTIR K., UNGER D., TSCHELIGI M.: Facetclouds: Exploring tag clouds for multi-dimensional data. In *Graphics Interface* (2013), pp. 17–24. 3
- [WWL*23] WANG C., WU L., LIU X., LI X., MENG L., MENG X.: Anything to glyph: Artistic font synthesis via text-to-image diffusion model. In *SIGGRAPH Asia* (2023), ACM, pp. 1–11. 2
- [XK07] XU J., KAPLAN C. S.: Calligraphic packing. In *Graphics Interface* (2007), Canadian Human-Computer Communications Society, pp. 43–50. 1, 2
- [XXL23] XIA Z., XIONG B., LIAN Z.: Vecfontsdif: Learning to reconstruct and synthesize high-quality vector fonts via signed distance functions. In *CVPR* (2023), pp. 1848–1857. 3
- [ZCR*16] ZOU C., CAO J., RANAWEEA W., ALHASHIM I., TAN P., SHEFFER A., ZHANG H.: Legible compact calligrams. *ACM TOG* 35, 4 (2016). 1, 2, 8, 9, 10
- [ZWXL17] ZHANG J., WANG Y., XIAO W., LUO Z.: Synthesizing ornamental typefaces. *CGF* 36, 1 (2017), 64–75. 2
- [ZYZ*22] ZHANG J., YANG Z., JIN L., LU Z., YU J.: Creating word paintings jointly considering semantics, attention, and aesthetics. *ACM TAP* 19, 3 (2022), 1–21. 2