Master's Thesis

Systematic Parameter Optimization for ComfyUI Workflows Using Multi-Agent Systems

Naqqash Khan

October 30, 2026

Advisor: Tobias Dick

Reviewer: Prof. Dr. Sven Apel
Prof. Dr. Anna Maria Feit





Chair of Software Engineering ◇ Saarland Informatics Campus ◇ Saarland University

Naqqash Khan: *Systematic Parameter Optimization for ComfyUI Workflows Using Multi-Agent Systems,* © October 2026

Erklärung Statement

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und ohne die Beteiligung dritter Personen verfasst habe, und dass ich keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Alle Stellen der Arbeit, die wörtlich oder sinngemäß aus Veröffentlichungen oder aus anderweitigen fremden Äußerungen entnommen wurden, sind als solche kenntlich gemacht. Insbesondere bestätige ich hiermit, dass ich bei der Erstellung der nachfolgenden Arbeit mittels künstlicher Intelligenz betriebene Software (z. B. ChatGPT) ausschließlich zur Textüberarbeitung/-korrektur und zur Code-Vervollständigung und nicht zur Bearbeitung der in der Arbeit aufgeworfenen Fragestellungen zu Hilfe genommen habe. Alle mittels künstlicher Intelligenz betriebenen Software (z. B. ChatGPT) generierten und/oder bearbeiteten Teile der Arbeit wurden kenntlich gemacht und als Hilfsmittel angegeben. Ich erkläre mich damit einverstanden, dass die Arbeit mittels eines Plagiatsprogrammes auf die Nutzung einer solchen Software überprüft wird. Mir ist bewusst, dass der Verstoß gegen diese Versicherung zum Nichtbestehen der Prüfung bis hin zum Verlust des Prüfungsanspruchs führen kann.

I hereby declare that I have written this thesis independently and without the involvement of third parties, and that I have used no sources or aids other than those indicated. All passages taken directly or indirectly from publications or other external sources have been identified as such. In particular, I confirm that I have used AI-based software (e.g., ChatGPT) exclusively for the following permitted sub-tasks: text rewriting/revision and code completion, and not to address or formulate the main research questions of the thesis. All parts of the thesis that were generated and/or edited using AI-based software (e.g., ChatGPT) have been disclosed and documented in accordance with the documentation requirements. I agree that the thesis may be checked using plagiarism detection software, including checks for the use of such software. I am aware that any violation of this declaration may result in failing the examination and lead to losing the right to be examined.

Saarbrücken, 03.11.2026 
(Datum Date) (Unterschrift Signature)

Einverständniserklärung (optional) Declaration of Consent (optional)

Ich bin damit einverstanden, dass meine (bestandene) Arbeit in beiden Versionen in die Bibliothek der Informatik aufgenommen und damit veröffentlicht wird.

I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Saarbrücken, 03.11.2026 
(Datum Date) (Unterschrift Signature)

Abstract

Modern generative Artificial Intelligence (AI) image workflows are becoming increasingly complex, as users must understand not only prompts and models but also how technical parameters influence visual outcomes. This challenge is especially evident in text-to-image diffusion models, which expose many interacting parameters that demand expert tuning for prompt adherence and image quality. Workflow environments such as ComfyUI address part of this complexity by making generation pipelines explicit and visually inspectable, but the individual nodes within these workflows still expose numerous parameters whose combined effects remain difficult to reason about and optimize. This thesis presents a Model Context Protocol (MCP)-based multi-agent optimization system integrated into ComfyUI as a chat-based sidebar, in which a configuration agent proposes inference-time settings and an evaluator agent diagnoses the resulting trials through a shared MCP server that exposes the parameter space, prior findings, and the trial history as structured resources. A factorial study finds that quality and runtime are driven by largely disjoint subsets of the four parameters examined, and a comparison against random search, grid search, and a Tree-structured Parzen Estimator (TPE) sampler shows that the agent reaches the same quality ceiling as the baselines in substantially fewer trials. A within-subjects user study with thirteen participants finds that the assistant lifts perceived usability for every participant and sharply reduces perceived mental effort. Together, these results suggest that a prior-informed assistant can match established parameters in substantially fewer trials while reducing the user's tuning burden, and that the approach is well matched to problems that already admit a small, exploitable prior.

Contents

1 Introduction 1
2 Background 3
3 Approach 9
3.1 Proposed System 10
3.1.1 Implementation 11
3.1.2 User Interaction 12
3.2 Research Questions 14
4 Evaluation 17
4.1 RQ1 17
4.1.1 Operationalization 17
4.1.2 Results 18
4.1.3 Discussion 24
4.2 RQ2 25
4.2.1 Operationalization 25
4.2.2 Results 26
4.2.3 Discussion 32
4.3 RQ3 33
4.3.1 Operationalization 33
4.3.2 Results 34
4.3.3 Discussion 41
4.4 Discussion 42
4.5 Threats to Validity 42
4.6 Future Work 44
5 Related Work 47
6 Concluding Remarks 51
A Appendix 53
A.1 Evaluation prompt set 53
A.2 Per-category main-effect sizes (RQ1) 54
A.3 Agent inference cost 54

Statement on the Usage of Generative Digital Assistants 57

Bibliography 59

List of Figures

Figure 2.1 A simple example of a ComfyUI workflow. ComfyUI is a high-level visual programming environment for designing complex systems that can generate different types of media. 5

Figure 3.1 MCP-based multi-agent architecture for optimizing diffusion pipeline configurations in a ComfyUI environment. A configuration agent proposes inference settings executed by the sampler, while an evaluator agent analyzes resulting metrics. MCP resources store prompts, configurations, generated images, and evaluation artifacts to support iterative optimization and traceability. 9

Figure 3.2 The optimizer panel docked in the ComfyUI sidebar during a session, with the details expander of a completed trial opened. 13

Figure 3.3 The version picker shown at the end of a session. 14

Figure 4.1 Marginal distributions of the four outcome metrics across all 1,440 completed runs. Dashed red lines indicate the empirical mean. . . . 19

Figure 4.2 Spearman correlations between outcome metrics. 20

Figure 4.3 Parameter-level effects on the outcome metrics. Stars denote significance levels: $^{*}\ p < 0.05$, $^{**}\ p < 0.01$, $^{***}\ p < 0.001$; unmarked cells are not statistically significant. 20

Figure 4.4 Distribution of each outcome metric grouped by guidance scale (Classifier-Free Diffusion Guidance (CFG)). 21

Figure 4.5 Distribution of each outcome metric grouped by inference step count. 21

Figure 4.6 Two-way Analysis of Variance (ANOVA) partial η^2 for every parameter pair on every metric. Stars denote significance levels: $^{*}\ p < 0.05$, $^{**}\ p < 0.01$, $^{***}\ p < 0.001$; unmarked cells are not statistically significant. 22

Figure 4.7 Interaction between sampler and scheduler. 23

Figure 4.8 Interaction between CFG and inference steps. 23

Figure 4.9 Interaction between sampler and CFG. 23

Figure 4.10 Marginal distributions of the three quality metrics per search method across all trials. 26

Figure 4.11 Best-so-far (BSF) curves grouped by prompt category. 27

Figure 4.12 Direction-aligned Area Under the Curve (AUC)-BSF per prompt, one panel per quality metric. 28

Figure 4.13 Empirical Cumulative Distribution Function (CDF) of the final BSF score per prompt. Each method traces a step function whose height at a given value is the fraction of the eight prompts scoring at or below it. 29

Figure 4.14 Trials required to reach 90 % of the best score any method achieves on the same prompt, per method and quality metric. 29
Figure 4.15 Trials to beat the *default* baseline configuration. 30
Figure 4.16 Mean Gower distance from each new trial to the set of previously tried configurations. 30
Figure 4.17 Mean regret over time per method. 31
Figure 4.18 Overall procedure of the RQ3 user study. 33
Figure 4.19 Distribution of prior experience across the 13 participants. 35
Figure 4.20 Best-so-far quality per participant over the trials of each condition, aggregated as median with interquartile band. 35
Figure 4.21 Final configuration selected by each participant, per condition. . . . 36
Figure 4.22 Perceived usability by condition: (a) paired System Usability Scale (SUS) totals per participant with the Wilcoxon signed-rank and Cliff's δ annotated in the panel, (b) Bangor adjective ratings [3] selected per condition, and (c) per-item SUS contribution with Benjamini–Hochberg-corrected paired Wilcoxon significance marked above each item pair. 37
Figure 4.23 Post-task satisfaction with the produced image. 38
Figure 4.24 Direct choice comparison items. Negative values (orange) indicate the response leans toward the without-agent condition; positive values (green) toward the treatment condition; zero denotes no difference between conditions. 38
Figure 4.25 Response distribution across the seven agent-only Likert items, ordered by mean rating. 39
Figure 4.26 Spearman rank correlation between three measures of prior experience and each agent-only item. 40
Figure A.1 One-way ANOVA η^2 per parameter and per prompt category, computed independently for each of the four outcome metrics. 55

List of Tables

Table 4.1 Parameter space for the primary RQ1 experiment. 17
Table 4.2 Mean of each metric by prompt category, averaged across all 1,440 completed runs. Arrows indicate the direction of improvement. . . . 19
Table 4.3 Candidate space for each method in the RQ2 comparison. 25

Acronyms

AGIQA Artificial Intelligence-Generated Image Quality Assessment
AI Artificial Intelligence
API Application Programming Interface
AUC Area Under the Curve
AWS Amazon Web Services
BRISQUE Blind/Referenceless Image Spatial Quality Evaluator
BSF Best-so-far
CDF Cumulative Distribution Function
CDK Cloud Development Kit
CFG Classifier-Free Diffusion Guidance
CLIP Contrastive Language-Image Pre-Training
DPM Diffusion Probabilistic Model
EC2 Elastic Compute Cloud
FID Fréchet Inception Distance
HPS Human Preference Score
IQA Image Quality Assessment
IS Inception Score
JSON JavaScript Object Notation
LLM Large Language Model
LPIPS Learned Perceptual Image Patch Similarity
MCP Model Context Protocol
S3 Simple Storage Service
SDK Software Development Kit
SUS System Usability Scale
TPE Tree-structured Parzen Estimator
VPC Virtual Private Cloud
ANOVA Analysis of Variance

1 Introduction

Text-to-image diffusion models have made image generation far more accessible than earlier generative approaches, but producing a semantically aligned and high-quality image is still not a one-click problem. Modern diffusion models can synthesize detailed images from natural-language prompts, and latent diffusion makes this process computationally practical at high resolution [46]. Yet in real use, output quality depends not only on the prompt, but also on how the generation process is configured. Configuration choices such as guidance strength, denoising schedule, solver, and sampling budget can substantially affect prompt adherence, visual quality and computational cost. Classifier-free guidance explicitly trades off sample quality against diversity [24]. Similarly, the noise schedule strongly affects the final result, and the best setting can vary by task [7]. Also, improved sampling methods such as Diffusion Probabilistic Model (DPM)-Solver can produce high-quality images in fewer steps, which makes generation faster and more efficient [36].

This difficulty becomes even more visible in practice when generation moves from a single model call to a workflow-based environment such as ComfyUI. ComfyUI embeds image generation in a modular graph of components, models, and parameterized nodes rather than in a single fixed pipeline. This modularity is powerful, but it also makes the system harder to understand, debug, and optimize. Recent work on ComfyUI-based assistants and workflow-generation systems consistently highlights this tension, pointing to issues such as complex node structures, scattered documentation, configuration errors, and the difficulty of producing valid workflows automatically [28, 61, 62]. Taken together, these studies suggest that the challenge is no longer only how to generate images, but how to manage the growing configuration complexity of modern generative pipelines.

This challenge becomes especially important once we delegate optimization to an automated system. While a human user may often judge directly whether one generated image appears better than another, an autonomous optimizer requires explicit evaluation signals in order to compare configurations systematically. A second difficulty, therefore, is deciding what it means for one configuration to be better than another in a machine-usable way. Iterative optimization depends on automatic evaluation, but different metrics capture different aspects of image quality, semantic alignment, perceptual naturalness, and human preference, so they should not be treated as interchangeable [37, 52, 55, 59]. These works show that evaluating generative outputs is inherently multi-dimensional, with each metric reflecting a different notion of success.

Current generative pipelines expose many meaningful control variables, but there is still limited support for systematically exploring them under realistic constraints. Users often rely on defaults, intuition, or laborious trial-and-error, even though the literature already shows that generation behavior is highly sensitive to these choices [7, 24, 36]. The central

idea of this thesis is therefore to study whether an autonomous, agent-based optimization framework can make that search more effective, more interpretable, and more reproducible.

A central part of this goal is to investigate whether an MCP can serve as the structured grounding layer between the agent system and the contextual information required to drive the optimization loop, such as the parameter space, prior findings, and the history of completed trials, including their outcomes, generated images, and recorded diagnostic feedback. The MCP specification defines it as an open protocol for connecting Large Language Model (LLM) applications to external data sources and tools, and organizes interaction around primitives such as *tools*, *resources*, and *prompts* [1]. This structure makes it well suited to the grounding role investigated here, since it provides a standardized way to expose the information to the agent through a single, well-defined interface.

Using our MCP-based agent, we aim to optimize diffusion pipeline configurations with respect to two main objectives. First, it aims to improve semantic alignment, that is, how well a generated image matches the intent expressed in the prompt. Second, it aims to improve perceptual quality, including visual coherence, naturalness, and the reduction of obvious artifacts. Prior work motivates these objectives as generated-image assessment involves both prompt-related content accuracy and perceptual quality, and that different automatic evaluators capture different aspects of this problem [37, 52, 55, 59].

A further goal is interpretability. We aim not only to identify better-performing parameter settings, but also explain which parameters matter, how they interact, and how the use of MCP affects the structure and traceability of the optimization workflow. To evaluate these goals, the thesis combines technical experiments with a human-centered study. This additional human-centered perspective is valuable because even if automated evaluation can assess image quality and optimization outcomes, it cannot by itself show whether the resulting workflow traces and explanations are actually understandable and useful for human users. The technical evaluation will examine how parameter interactions influence output quality and whether agents can map high-level design intentions to effective parameter settings. The user study will investigate whether MCP-grounded traces and explanations support understanding and configuring generative workflows.

2

Background

This section introduces the technical foundations of the thesis by following the logic of the problem from generation to optimization. It begins with text-to-image diffusion models as the underlying generation paradigm and then explains why their parameters make configuration choices consequential for output quality, alignment, and efficiency. Building on this, it describes how workflow-based environments make these choices explicit in modular pipelines. It then presents MCP as a protocol layer through which agents can interact with such workflows, tools, and artifacts in a structured way. Finally, because any optimization process requires a criterion for judging improvement, the section motivates automated image evaluation as a multi-signal problem.

Text-to-image diffusion. Modern text-to-image generation commonly builds on diffusion models, which formulate image synthesis as an iterative denoising process. In simplified terms, a diffusion model learns how to reconstruct an image progressively from noisier versions of it, and generation then works by starting from random noise and gradually refining it into an image through many denoising steps. Ho et al. [23] showed that this general approach can produce high-quality image synthesis and established diffusion as a strong generative modeling framework.

For text-to-image generation, the model conditions the denoising process on a natural-language prompt, guiding the generated image toward the content described by the user. Latent diffusion models perform denoising not in pixel space but in a learned latent space produced by an autoencoder. This reduces computational cost while preserving much of the important visual structure of the image, which makes high-resolution generation more practical. Latent Diffusion Models further introduce cross-attention layers so that text information can guide image synthesis effectively during denoising [46].

Configuration parameters in diffusion pipelines. A diffusion pipeline exposes several controllable inference-time variables that determine how the pipeline generates an image from noise. These parameters matter because the model does not produce the final image in a single step. Instead, it gradually transforms random noise into a structured image over a sequence of denoising operations. As a result, changing these settings can alter how closely the image follows the prompt, how natural or detailed it appears, how diverse the outputs remain, and how much computation is required.

One important parameter is *guidance strength*. In simplified terms, guidance controls how strongly the model pushes generation toward the prompt. If the guidance is too weak, the resulting image may drift away from the intended content. Similarly, if it is too strong, the image may become less diverse or develop unnatural artifacts. CFG shows that guidance

introduces a trade-off between sample quality and diversity, which makes it a meaningful control variable rather than a cosmetic setting [24].

Another important concept is the *noise schedule*. During diffusion, different stages of the process correspond to different levels of noise, and the schedule determines how these levels are distributed across the denoising process. Chen [7] show that the noise schedule determines the distribution of noise levels seen during training and that it is crucial for performance, with the best schedule depending on the task and even on image size. This means that the schedule is not just a hidden implementation choice, but one of the factors that shapes how effectively the model learns and generates.

At inference time, the *solver* and the *number of denoising steps* determine how the model moves through the denoising trajectory from noise to image. The number of steps controls how many refinement stages the sampler performs. More steps can improve quality, but they also increase runtime, while too few steps may lead to incomplete or unstable generation. The solver determines how these steps are taken, that is, how the model approximates the denoising process numerically. Different solvers can therefore reach different quality–efficiency trade-offs even under the same step budget. Lu et al. [36] show that DPM-Solver accelerates diffusion sampling substantially and produces high-quality samples with far fewer function evaluations than standard approaches, highlighting that solver design and step count directly influence both output behavior and computational efficiency.

ComfyUI as a workflow-based environment. ComfyUI is a node-based environment for building and executing generative AI workflows. Instead of interacting with a single model through one prompt box, users construct a pipeline by connecting modular components that represent different processing steps, such as prompt encoding, model loading, sampling, decoding, or post-processing, as shown in Figure 2.1. This makes the generation process explicit. The users can see how data moves through the workflow and can modify individual stages without rebuilding the entire pipeline. ComfyUI has also grown into a widely used ecosystem. According to Xu et al. [62], it serves over 4 million active users and is supported by a community contributing more than 12,000 components.

A key characteristic of ComfyUI is that workflows are graph-structured and machine-processable. In practice, ComfyUI represents a workflow as nodes with their connections and parameter settings, and can export it in a structured format such as JavaScript Object Notation (JSON) [62].

At the same time, the flexibility of ComfyUI also creates complexity. Since users can combine many different nodes, models, and parameter settings, effective workflow construction often requires substantial practical knowledge. Prior work on ComfyUI-based assistants highlights challenges such as complex node structures, scattered documentation, configuration errors, and the difficulty of producing valid workflows automatically [28, 62].

Model Context Protocol. The Model Context Protocol (MCP) [1] is an open protocol for structured interaction between LLM-based applications and external capabilities such as software tools, data sources, and reusable prompt templates. Rather than embedding all functionality directly inside a single application, MCP defines a standardized interface through which an LLM system can discover, invoke, and coordinate external components in a controlled way. In architectural terms, this allows an LLM-based client or host application

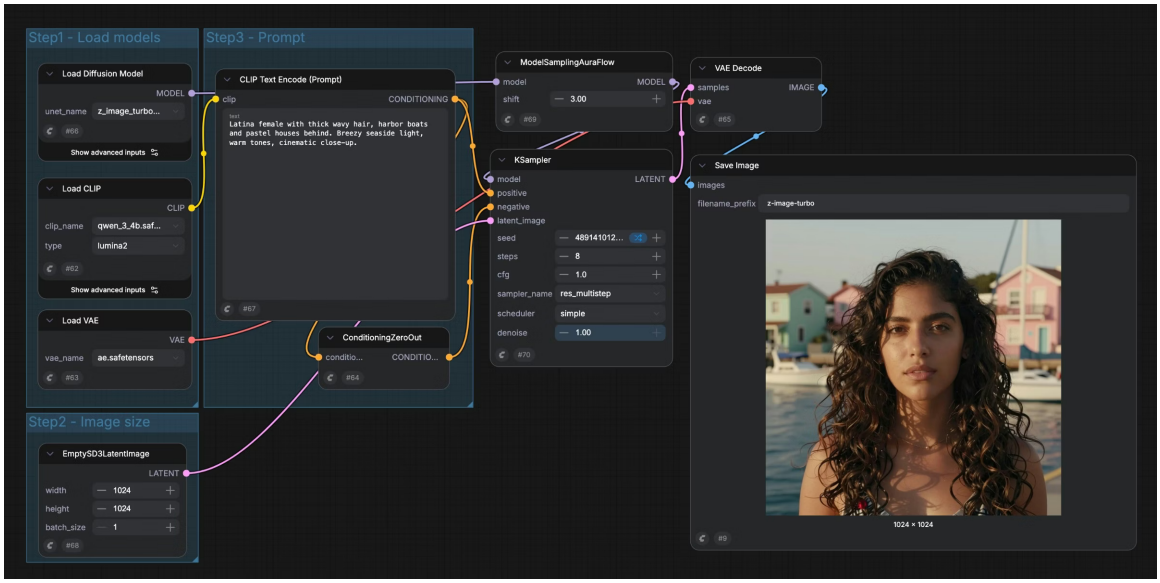


Figure 2.1: A simple example of a ComfyUI workflow. ComfyUI is a high-level visual programming environment for designing complex systems that can generate different types of media. Source: `https://docs.comfy.org/development/core-concepts/workflow`, accessed on April 10, 2026.

to communicate with one or more MCP servers that expose capabilities through a common protocol layer.

At the conceptual level, the protocol distinguishes between several core abstractions. *Tools* represent executable functions that the model can call, for example to query historical experimental results, retrieve specific records by a given criterion, or fetch derived feedback about the current state of a task. *Resources* represent structured or retrievable context that the model reads rather than executes. They range from textual artifacts such as prior experiment summaries, reference findings, or current state snapshots, to binary content such as previously generated images that the model can inspect directly. *Prompts* here represent reusable instruction templates rather than the end-user's typed request. The server offers these templates, which include system messages, role framings, or task instructions, to the agent. As a result, the instructions an agent operates under flow through the same mechanism as its tools and data. Together, these abstractions separate action, context, and instruction into explicit protocol objects rather than leaving them implicit inside one monolithic prompt.

Operationally, an MCP-based interaction proceeds as a sequence of structured exchanges. A host application mediates between the language model and external MCP endpoints. During execution, the host application provides the model with available tools, resources, and prompts. It can then select appropriate actions, invoke tools with structured arguments, retrieve relevant resources, and incorporate the returned results into subsequent reasoning steps. In this way, MCP turns external interaction into an inspectable workflow of typed operations instead of an opaque sequence of ad hoc Application Programming Interface (API) calls.

Image evaluation metrics. A central concept in this thesis is that evaluation of generated images is multi-dimensional. A generated image can be judged from several perspectives at once. Whether it matches the prompt, whether it looks perceptually natural, whether it contains visible artifacts, and whether humans would actually prefer it over alternatives. Because the optimizer produces novel images without ground-truth counterparts to compare against, we restrict the evaluators considered here to reference-free metrics. This excludes widely used but reference-based measures such as Fréchet Inception Distance (FID) [22], Inception Score (IS) [48], and Learned Perceptual Image Patch Similarity (LPIPS) [64], which either assume a reference distribution of real images or a paired ground-truth image, and motivates the no-reference evaluators discussed below, each targeting a different axis of quality.

A useful starting point for several of the evaluators used in this thesis is Contrastive Language-Image Pre-Training (CLIP). CLIP is a vision-language model trained on large-scale image-text pairs to map images and text into a shared representation space. This allows users to address visual concepts through natural language and supports strong zero-shot transfer to downstream tasks without task-specific retraining [45]. Building on this representation, Hessel et al. [21] proposed CLIP Score, a reference-free metric for image–text alignment that scores a generated image against its prompt as the cosine similarity between their CLIP embeddings. Because it directly measures how well the visual content reflects the prompt's textual intent without requiring a ground-truth reference image, CLIP Score is a natural choice for evaluating semantic alignment in text-to-image generation. It also has known limitations, most notably that cosine similarity in CLIP space captures coarse semantic correspondence and is less sensitive to fine compositional or perceptual detail, which motivates combining it with other evaluators that target different aspects of image quality.

Blind/Referenceless Image Spatial Quality Evaluator (BRISQUE) [37] represents a more classical no-reference image quality assessment perspective. Instead of relying on prompt information or human preference data, it estimates quality from the image alone by measuring deviations from natural scene statistics in the spatial domain. Concretely, it extracts locally normalized luminance coefficients at multiple scales and maps their statistics to a scalar score through a support vector regressor trained on subjective human quality ratings, where lower scores indicate images closer to natural-scene statistics. Intuitively, it aims to capture how natural or visually plausible an image appears, and to detect distortions such as blur, noise, or compression artifacts that reduce this naturalness. Although BRISQUE was not originally designed for AI-generated images, it remains useful as a complementary signal because it provides a prompt-independent estimate of perceptual degradation or artifact severity.

Human Preference Score (HPS) [59] addresses a different notion of quality. Not only whether an image is natural or aligned with a prompt, but whether people actually prefer it. The method fine-tunes CLIP on large-scale human preference data so that it can predict pairwise preferences between generated images more accurately than generic similarity-based metrics. This makes HPS especially relevant that for a thesis that studies optimization, because many practical improvements in image generation are ultimately meaningful only if they correspond to outputs that users judge as better. In other words, HPS provides

a preference-oriented signal that complements more traditional perceptual or semantic evaluators.

3

Approach

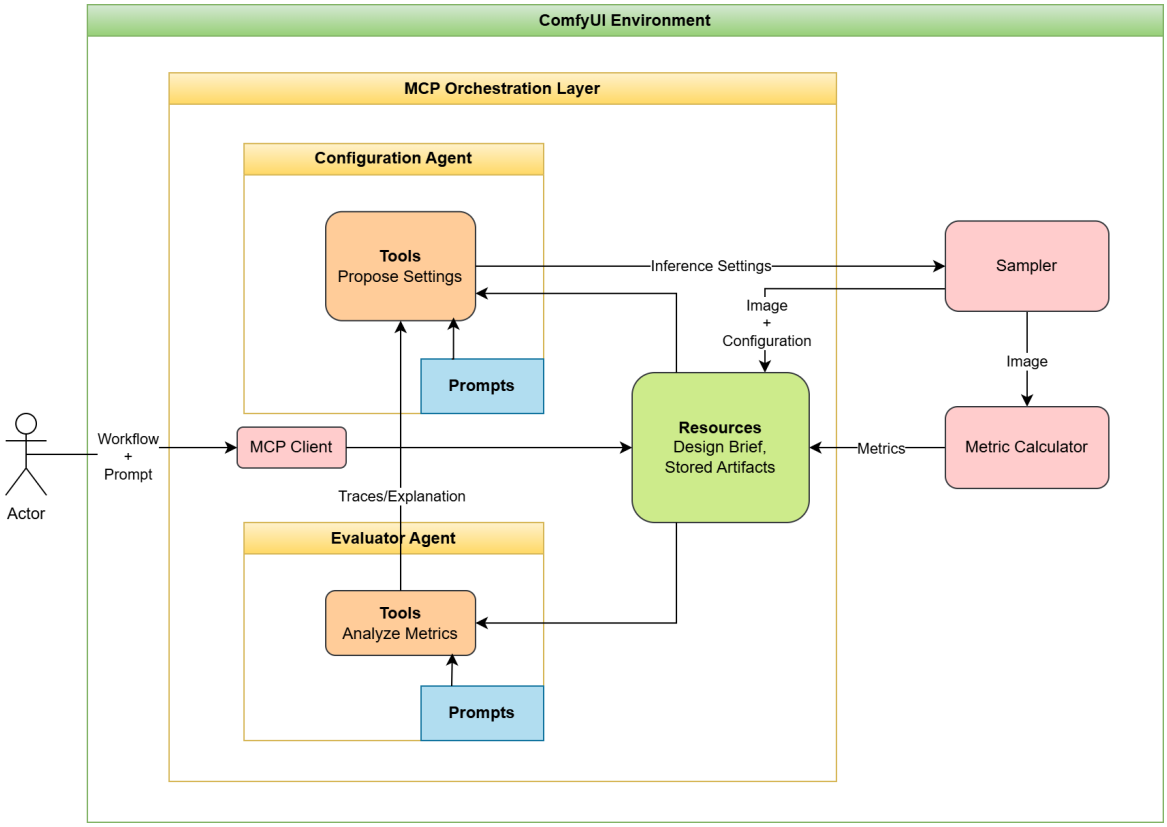


Figure 3.1: MCP-based multi-agent architecture for optimizing diffusion pipeline configurations in a ComfyUI environment. A configuration agent proposes inference settings executed by the sampler, while an evaluator agent analyzes resulting metrics. MCP resources store prompts, configurations, generated images, and evaluation artifacts to support iterative optimization and traceability.

This chapter presents the proposed technical approach of the thesis and the research questions that guide its empirical evaluation. The central idea is to develop an MCP-based multi-agent system for configuration optimization in text-to-image diffusion workflows. As shown in Figure 3.1, we organize the system around a grounding layer that connects a configuration agent, an evaluator agent, shared resources, and workflow execution components. Rather than treating image generation as a single prompt-to-image operation, the approach models it as an iterative optimization process in which the agents propose, execute, evaluate,

and refine configurations. Building on this design, we then formulate the research questions that structure the evaluation in Chapter 4.

3.1 Proposed System

Figure 3.1 illustrates the overall architecture of the proposed system. Conceptually, a user provides a workflow and prompt to the orchestration layer. In our experimental setup, we instantiate this with a fixed workflow and a benchmark suite of different prompts. Within this layer, the configuration agent proposes inference settings based on the design brief, available prompts, and previously stored results. The orchestration layer then passes these settings to the sampler, which executes the workflow and produces an image.

The system stores the generated image and its configuration as resources and passes them to the metric calculator. The metric calculator then returns the evaluation signals to the shared resource layer, where they become available to the evaluator agent for analysis and to the configuration agent for subsequent proposal updates. In this way, the system forms an iterative feedback loop in which prior outcomes inform every configuration decision rather than leaving each one isolated.

The architecture separates high-level design intent from low-level execution choices while still connecting them through explicit artifacts and traces. This separation is important because it allows the optimization process to remain inspectable, that is, the system stores prompts, configurations, outputs, metrics, and explanatory traces, all of which can be revisited during analysis. At the same time, the shared resource layer ensures that both agents operate over the same evolving experimental context.

The system consists of two main agents. The *Configuration Agent* proposes candidate workflow settings for each optimization round, drawing on the design brief, the history of previous trials, and explicit search constraints. Concretely, we design the agent to balance exploration and exploitation by first proposing diverse configurations that cover different regions of the search space and then progressively concentrating on neighborhoods of high-performing settings as evaluation results accumulate. In this design, the system does not use the LLM as a stand-alone optimizer that replaces conventional search, but as a context-sensitive proposal mechanism that can incorporate natural-language design intent, prior traces, configuration constraints, and direct user feedback when selecting the next candidates. Such feedback may express corrective preferences that are difficult to infer from the prompt and evaluation metrics alone, for example that the current image emphasizes one aspect too strongly while another desired element remains underrepresented. This is particularly useful in a workflow setting where optimization is not defined solely by a fixed numeric objective, but also by prompt-dependent goals, mixed parameter types, and the need to avoid infeasible or uninformative configurations. In this way, the system supports not only autonomous search but also iterative human-in-the-loop refinement.

The *Evaluator Agent*, in turn, analyses each completed trial to produce the structured feedback that guides subsequent proposals. It receives the quantitative metrics, the history of previously completed trials, and the generated image. Its role is to diagnose which parameters most plausibly drove the observed quality and runtime outcome. This object is persisted with the trial as part of the experiment trace and forwarded directly to the

configuration agent on the following step. Together, the two agents form an iterative optimization loop in which language-grounded reasoning guides proposal generation, while image- and metric-based evaluation provides the empirical grounding needed for steering. The selection of retaining or refining configurations is an emergent property of the loop, shaped by how its diagnosis influences the next proposal.

Operationally, each optimization episode begins with a design brief and prompt that specify the desired content and visual characteristics of the output. The configuration agent proposes one or more candidate inference settings, such as sampler choice, scheduler, guidance scale, or number of inference steps. After execution, the system stores the resulting image, configuration, metric values, and relevant metadata as structured artifacts. These artifacts provide the basis for subsequent evaluation and for the next round of configuration proposals, and the loop continues until a stopping criterion is reached.

3.1.1 Implementation

We use the Qwen-Image diffusion model [58] for the text-to-image pipeline, which is executed inside a fixed ComfyUI workflow. We selected this model because, at the time of implementation, it was one of the top-performing open-weight text-to-image models available on public benchmarks such as GenEval [18] and integrates natively with ComfyUI, which together satisfy the reproducibility and fixed-workflow constraints of this study. The workflow is defined in a JSON export checked into the repository and is held constant across all experiments. The configurable parameters are exposed as inputs to this single workflow rather than realised by different workflow graphs. We generated images at 1024×1024 resolution under a fixed denoising strength of 1.0, with model-level parameters stored as part of an immutable configuration record so that every trial in the database can be traced back to identical model-level conditions.

The three quantitative quality metrics that we use are inside the ComfyUI workflow itself via custom nodes, so that the orchestrator and the agents observe the same numbers that any user of the workflow would. CLIP Score is computed against the ViT-B/16 backbone of OpenAI's CLIP model as the cosine similarity between the CLIP embeddings of the generated image and the prompt. HPS v2 is produced by a dedicated ComfyUI-HPSv2 node, and BRISQUE is provided by a no-reference Image Quality Assessment (IQA) node from the FRED node pack. We record runtime as wall-clock seconds reported by the ComfyUI execution backend running on a single `NVIDIA L40S Tensor Core` GPU, so the reported runtimes are valid as relative measurements within and across methods rather than as absolute throughput figures. The same nodes, weights, and node configurations are used for every trial.

The configuration agent and the evaluator agent calls to Claude Opus (`claude-opus-4-6` on Amazon Web Services (AWS) Bedrock at the time of writing), accessed through Anthropic's official Python Software Development Kit (SDK) and configured to support streaming responses and adaptive extended thinking. The two agents share the same model deployment but use distinct system prompts fetched from the MCP server. The exact model identifier is supplied via an environment variable so that the same agent code can be re-run against newer Claude releases without modification. Each agent call carries the same retry

policy: rate-limit and transient-error responses from Bedrock are retried with exponential backoff and jitter, so that long optimization runs remain robust to brief service interruptions.

We implemented MCP using the official Python FastMCP library which acts as the grounding layer. The system exposes its optimization state through the three abstractions of the protocol. *Tools* retrieve the parameter space, the history and outcomes of completed trials, and the diagnostic feedback from prior evaluator steps. *Resources* provide reference findings from the prior factorial analysis, the current state of a prompt's optimization run, and the generated image associated with each trial, which the evaluator agent inspects directly. *Prompts* supply the system messages that govern how the configuration agent proposes settings and how the evaluator agent interprets metric outcomes and visual content. The orchestrator spawns the server as a stdio subprocess once per optimization run and connects through a synchronous client wrapper that reuses the same session across all agent calls, so that every input the agents read—whether a tool result, a static reference, a generated image, or their own system prompt—travels through the MCP protocol rather than through an in-process function call.

Trial metadata, agent traces, and runner checkpoints are persisted in an Aurora PostgreSQL cluster accessed through SQLAlchemy 2.0; the schema is defined in code and applied via an idempotent initialisation routine. Each optimization trial row records the proposing method, prompt, parameter values, the four metrics, runtime, the image Simple Storage Service (S3) key, and a run identifier. Each agent trace row records the role (configuration or evaluator), the design brief, prior trials referenced, the rationale, and the image- and metric-grounded assessment suggested by the evaluator. Generated images are uploaded to an AWS S3 bucket whose name and prefix are configured via environment variables, with deterministic object keys derived from each trial's run identifier so that the trial image resource can resolve them at evaluation time. The supporting AWS infrastructure: Virtual Private Cloud (VPC), security groups, the Aurora cluster, the S3 bucket, and peering to the ComfyUI Elastic Compute Cloud (EC2) host is defined as an AWS Cloud Development Kit (CDK) stack in TypeScript alongside the source code, so that the empirical environment can be torn down and re-created from a single declarative specification.

3.1.2 User Interaction

The optimization loop described so far is autonomous, but the same system is also exposed to the user as an interactive assistant. We integrate the optimizer directly into the ComfyUI sidebar rather than as a separate application, so that the user configures a workflow without leaving the canvas on which that workflow is built. The extension presents itself as a chat-like panel that sits alongside the graph, as shown in Figure 3.2, and the graph itself remains visible and editable throughout, which means the user can watch the parameter widgets and the resulting image change as the conversation proceeds. This co-location is deliberate, that is, the optimization is meant to feel like an additional pane of the workflow editor.

The entry point of the panel is the session page. It lists all previous optimization sessions for the workflow, each labelled with its date, the number of versions it produced, and the opening words of the prompt it optimized, together with an option to start a new session for the workflow currently open on the canvas. When the user starts a new session, the

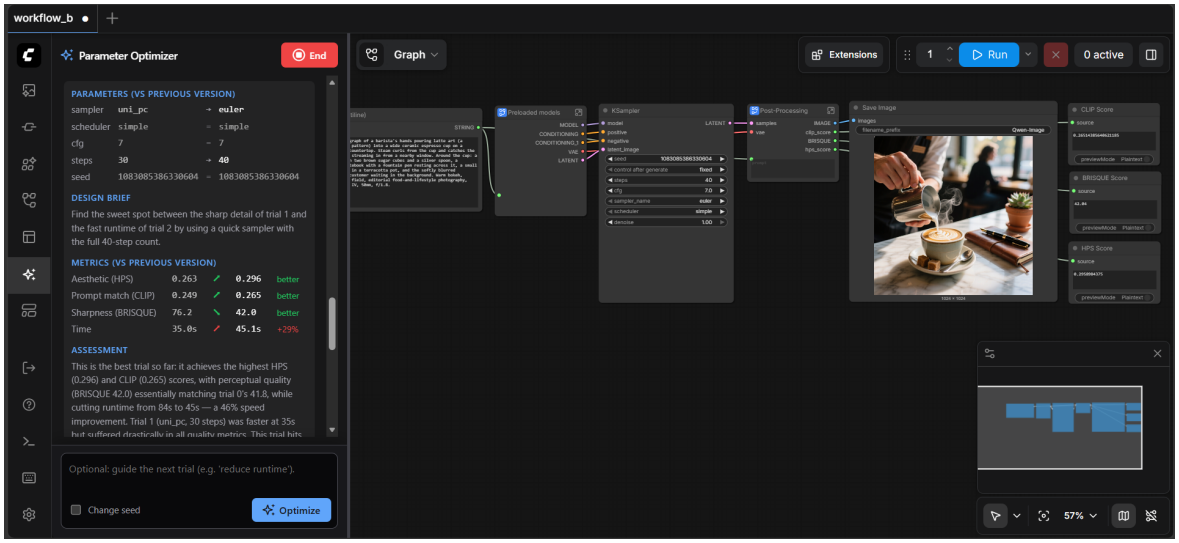


Figure 3.2: The optimizer panel docked in the ComfyUI sidebar during a session, with the details expander of a completed trial opened.

system extracts the prompt automatically from the text node of the workflow graph and shows it for confirmation. Once the session begins, the panel switches into conversation mode and the agents take over the loop.

Each iteration of that loop is initiated by the user through a single optimize action. An input box above it accepts an optional instruction that steers the next proposal, for example a request to reduce runtime or to preserve quality while doing so, but supplying one is not required, because the configuration agent is able to select the next configuration from the prompt, the parameter space, and the trials completed so far. When an instruction is given, it enters the same proposal step as the design brief and the trial history, so that user-directed preferences and accumulated evidence are weighed together. The agent first reasons about the category of the prompt and the settings that have historically suited it, and then proposes the configuration it expects to perform best in the current context.

The proposal is applied to the workflow without further user action. The system writes the proposed values into the `KSampler` widgets on the canvas, so that `sampler_name`, `scheduler`, `cfg`, `steps`, and `seed` visibly take on the values under test, and then triggers execution of the workflow. When the image has been generated and the CLIP Score, HPS, and BRISQUE nodes have produced their values, the evaluator agent inspects both the numbers and the image itself and returns its assessment. The right-hand side of Figure 3.2 shows the workflow in this state, with the sampler settings of the trial under test in place, the generated image in the save node, and the three metric nodes reporting the values that the evaluator agent receives. Every step of this sequence is surfaced in the panel as a message, so the announced proposal, the executed trial, and the resulting judgement form a single readable transcript. The generated image is embedded in the transcript next to the metrics of its trial and can be opened at full size with a click, which allows the user to check visual details that the aggregate metrics do not capture.

Because a full rationale would crowd the transcript, the messages are layered. The visible message states the decision and its outcome in plain language, while a details expander

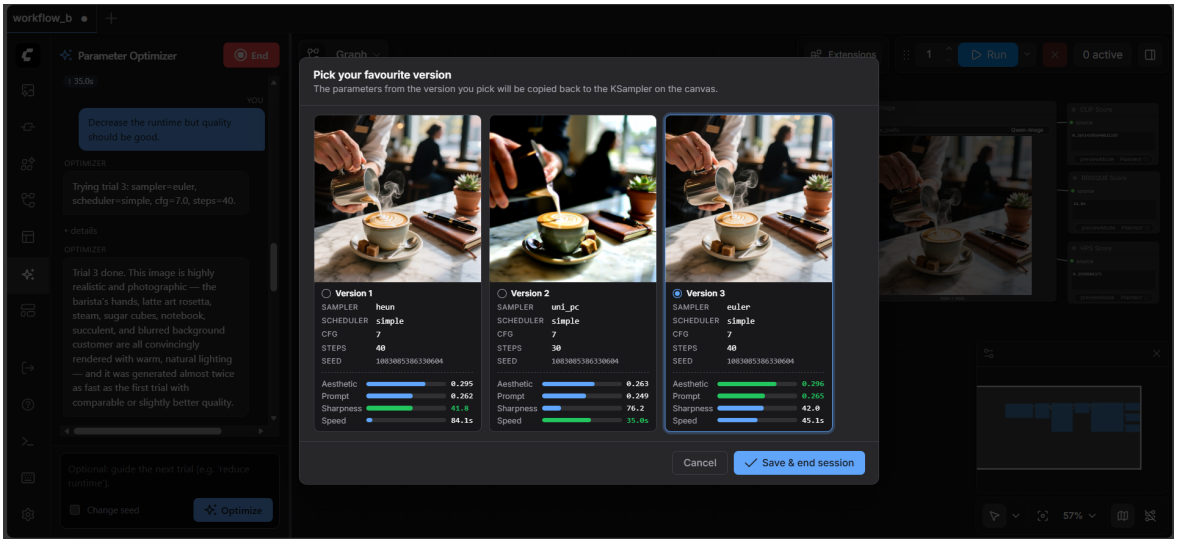


Figure 3.3: The version picker shown at the end of a session.

attached to it holds the technical account. The left-hand panel of Figure 3.2 shows such an expander in its opened state. It lists the proposed parameter values beside those of the previous trial and marks each one as changed or unchanged, states the design brief under which they were chosen, reports each metric with its previous value, its new value, and whether the change is an improvement, and closes with the assessment that explains the outcome in terms of the parameters that produced it. After reading the assessment the user proceeds to the next iteration, optionally attaching new guidance, and the loop repeats for as many trials as the user considers necessary.

The user terminates the session explicitly. Ending it opens a comparison window as shown in Figure 3.3, that presents every trial of the session side by side, each with its generated image, its full parameter set, its three quality metrics, and its runtime, and each metric drawn as a bar whose length encodes the value. The bar of the trial that holds the best value for a metric is colored with green, so that the strongest trial per dimension can be identified at a glance while the trade-offs between quality and runtime remain visible. The version the user selects is recorded as the outcome of the session and its parameters are copied back into the `KSampler` on the canvas, so the workflow is left in the state the user chose and can be executed or exported directly. The conversation itself remains available from the session page, so that it can be revisited later.

3.2 Research Questions

We organize the empirical evaluation around three research questions that together move from the structure of the configuration space, to the behavior of the optimizer over that structure, and finally to the value of the resulting system for its users.

RQ1: How do different configuration parameters interact and how do these interactions influence workflow performance in terms of generation quality and runtime

efficiency? This question examines the structure of the inference-time configuration space. It focuses on whether the effects of parameters such as sampler, scheduler, guidance scale, and inference steps are independent or whether they vary depending on one another and on prompt type, and on how these effects manifest across complementary evaluation dimensions.

RQ2: To what extent can autonomous agents explore the configuration space of a generative workflow and translate high-level design features into effective technical parameter settings?
Building on the structure identified in RQ1, this question evaluates whether the proposed optimization system can use the configuration space more effectively than simpler search strategies. The comparison considers achieved metric quality, search efficiency, and traceability of the decision process.

RQ3: Can users configure a generative workflow with less effort with the agent's assistance, and how useful do they find that support?
After RQ1 and RQ2 examine the optimization behavior, this question turns to the human-facing contribution. It has two parts. The first is effort, which we measure directly, that is, how long participants work and how many trials they run before they settle on a configuration, and what they achieve in doing so. The second is usefulness, that is, how helpful and how usable participants find the traces, stored artifacts, and explanations.

4

Evaluation

This chapter evaluates the three research questions introduced in Chapter 1. Together, they examine how inference-time parameters shape generation quality and efficiency, how automated optimization can navigate the resulting configuration space, and how our approach performs relative to practitioner-driven baselines. For each research question, we first describe our operationalization, including the experimental setup, the evaluation metrics, and the analytical procedure, so that the design decisions behind each study are transparent. We then report the results, combining quantitative evidence with illustrative examples where appropriate, and finally discuss potential implications, limitations, and threats to validity. Taken as a whole, the chapter aims to not only establish whether our approach is effective, but also to clarify under which conditions it works well, where it falls short, and what the findings suggest for practitioners who adopt similar workflows.

4.1 RQ1

This section examines the structure of the inference-time configuration space and asks how parameters such as sampler, scheduler, guidance scale, and inference steps shape generation quality and runtime efficiency, and whether their effects are independent or emerge only through their interactions.

4.1.1 Operationalization

To investigate RQ1, we conduct a study of controlled factorial experiment over a fixed workflow and model setup. The primary experiment varies four inference-time parameters that are both practically relevant and experimentally tractable: sampler, scheduler, guidance scale, and number of inference steps. Table 4.1 summarizes the parameter space used in the primary experiment.

Table 4.1: Parameter space for the primary RQ1 experiment.

Parameter	**Levels**	**Count**
Sampler	{Euler, Heun, DDIM, DPM 2, UniPC}	5
Scheduler	{simple, karras}	2
Guidance scale	{3, 5, 7}	3
Inference steps	{20, 30, 40}	3

This design yields 90 parameter combinations per prompt. The prompt set, listed in Appendix A.1, consists of 8 prompts divided across four categories: realism-oriented, stylized, composition-heavy, and multi-constraint prompts. We execute each configuration with two random seeds, resulting in 1,440 runs in total.

We selected these parameters because they represent core inference-time controls in diffusion workflows and directly affect the trade-off between prompt adherence, perceptual quality, and computational cost. The chosen levels cover a small but representative range for each parameter. The samplers provide diversity across commonly used solver types, the two schedulers let us test scheduler effects without inflating the design, and the guidance-scale and step-count levels capture low, medium, and high settings in a tractable way. We likewise designed the prompt set to balance diversity and feasibility. It captures different types of generation challenges while keeping the total number of runs manageable within the thesis timeline. We use two seeds per configuration to reduce seed-specific variation without introducing excessive computational cost.

For each run, the system records CLIP-based alignment, BRISQUE, HPS v2, runtime, and the full configuration. Together, these measures capture semantic, perceptual, preference-based, and efficiency-related aspects of workflow performance. The analysis evaluates how both individual parameters and parameter combinations influence these outcomes. It therefore considers main effects, that is, the average effect of a single parameter, as well as interaction effects, in which the effect of one parameter depends on the value of another. Where appropriate, the analysis includes prompt category as an additional factor in order to test whether these effects vary across prompt types. The analysis aims to identify which parameter settings behave reliably across conditions, which depend on context, and where trade-offs arise across the selected evaluation metrics.

4.1.2 Results

In RQ1, we executed all $5 \times 2 \times 3 \times 3 = 90$ parameter combinations against eight prompts, with two random seeds per combination, for 1,440 runs in total. Each run was scored on four metrics: CLIP (prompt–image alignment, higher is better), HPS (human-preference score, higher is better), BRISQUE (no-reference perceptual quality, lower is better), and execution time (wall-clock seconds, lower is better). We combined classical inferential statistics, namely one- and two-way ANOVA [16] with partial η^2 as the accompanying effect size [10] and Spearman rank correlations [51], so that every headline finding rests on at least two methodologically independent sources of evidence. Partial η^2 expresses the share of outcome variance attributable to a factor once the others are partialed out, with values near 0.01, 0.06, and 0.14 conventionally read as small, medium, and large, while Spearman ρ measures monotonic association on ranks and lies in $[-1, +1]$. We present the results as a single narrative i.e. first how the outcome metrics behave and relate to one another, then how the individual parameters move them, and finally how the parameters interact.

Outcome metrics and their relationships

We show the marginal distribution of each metric across all 1,440 runs in Figure 4.1. CLIP varies smoothly around a single peak, whereas HPS and runtime are each clearly bimodal.

Table 4.2: Mean of each metric by prompt category, averaged across all 1,440 completed runs. Arrows indicate the direction of improvement.

Prompt category	CLIP (↑)	HPS (↑)	BRISQUE (↓)	Runtime (s, ↓)
Composition-heavy	0.308	0.247	33.4	47.3
Multi-constraint	0.265	0.220	56.1	47.8
Realism-oriented	0.294	0.236	40.2	46.9
Stylized	0.325	0.220	62.0	47.0

This bimodal structure points to a single dominant parameter behind each of HPS and runtime, which we identify in the next section.

Table 4.2 reports the mean of each metric by prompt category. The clearest pattern is a tension within the stylized category, which attains the strongest prompt alignment yet the weakest perceptual quality. This is expected rather than contradictory: a stylized image can faithfully express the semantics a prompt asks for while deliberately departing from the natural-image statistics that BRISQUE rewards, so the two metrics pull in opposite directions. Multi-constraint prompts, which pack several requirements into one instruction, are consistently the hardest to align. Runtime, by contrast, is effectively identical across all four categories. Since prompt content changes what the model produces but not how long it takes, this is the first concrete sign that runtime is a property of the configuration alone.

The metrics correlation with each other determines how many objectives we are really optimizing (Figure 4.2). The three quality metrics move together. CLIP and HPS rise in step, and both fall as BRISQUE improves, so they can be treated as three views of one underlying quality axis rather than as competing targets. Runtime stands apart: it is essentially uncorrelated with all three, meaning a faster configuration is neither systematically better nor worse in quality. This near-orthogonality is one of the most useful structural facts in the experiment. It implies that quality and speed can be treated as two independent objectives, and that an optimizer can push on runtime without paying a hidden quality penalty. The trade-off between them is a deliberate choice, not an unavoidable cost.

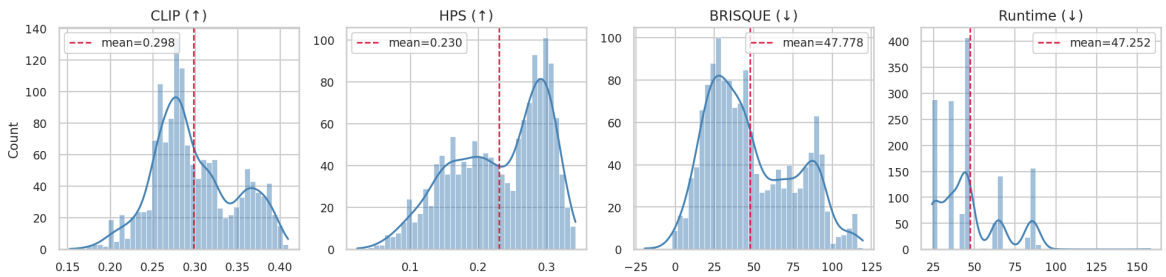


Figure 4.1: Marginal distributions of the four outcome metrics across all 1,440 completed runs. Dashed red lines indicate the empirical mean.

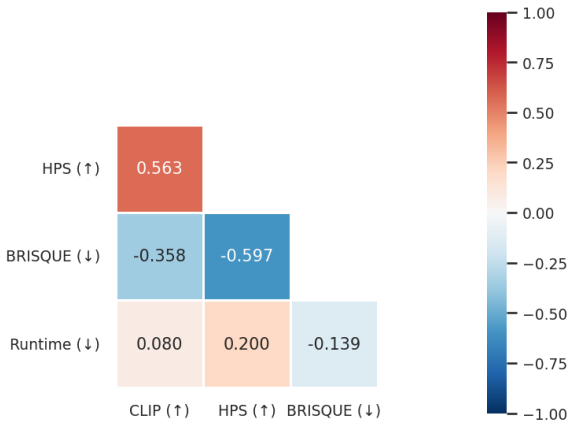
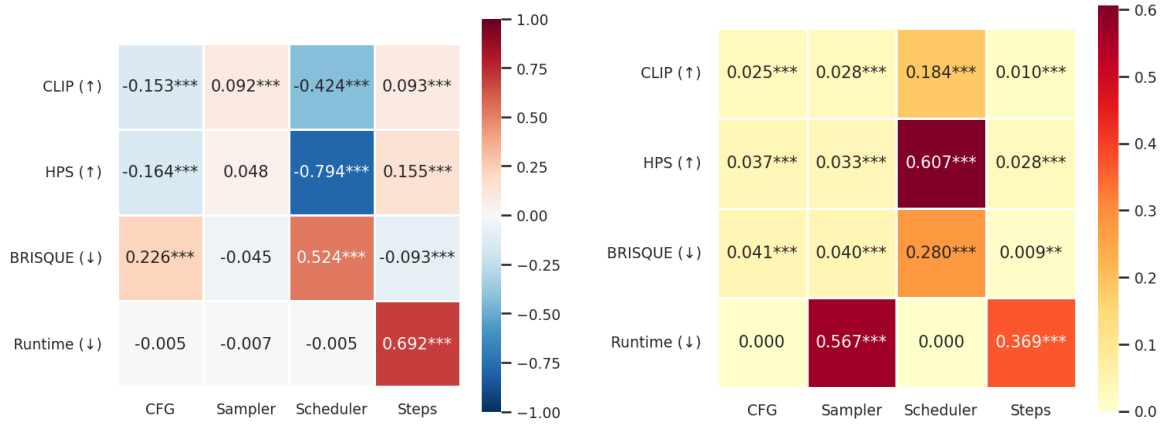


Figure 4.2: Spearman correlations between outcome metrics.



(a) Spearman parameter–metric correlations.

(b) One-way ANOVA η^2 effect sizes per parameter per metric.

Figure 4.3: Parameter-level effects on the outcome metrics. Stars denote significance levels: * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$; unmarked cells are not statistically significant.

Summary: The four outcome metrics can be grouped into two largely independent dimensions. CLIP, HPS, and BRISQUE vary together and therefore describe a single notion of image quality, whereas runtime varies almost independently of them. The pronounced bimodality of HPS and runtime further suggests that each dimension is controlled by a single dominant parameter.

Effects of individual parameters

Figure 4.3 reports the parameter–metric correlations and the one-way ANOVA effect sizes. On the quality axis the scheduler is decisive. It alone accounts for the majority of the variance in HPS ($\eta^2 = 0.607$) and is the leading contributor to the other two quality metrics as well. This is precisely the lever that split the HPS distribution into two regimes, and its dominance is the central result of the section. It is a single configuration choice, made

before any fine-tuning of the others, largely determines whether a run lands in the high- or low-quality mode. The finding is also somewhat counterintuitive. One might expect the sampler, the algorithm that actually integrates the diffusion process, to be the main determinant of quality, yet it explains almost none of it. Quality here is set by how the noise schedule is spaced, not by which solver traverses it.

Runtime tells a different but equally concentrated story. Wall-clock time is governed by the sampler and the step count and by nothing else: the samplers fall into a fast group and a group that is roughly twice as slow, and additional steps add time in near-linear fashion, so the two together produce the bimodal runtime distribution. Crucially, the two parameters that shape quality, the scheduler and CFG, have no measurable effect on runtime at all. This is the cleanest possible version of the orthogonality suggested by the metric correlations. The quality decisions and the speed decisions are made with disjoint sets of parameters.

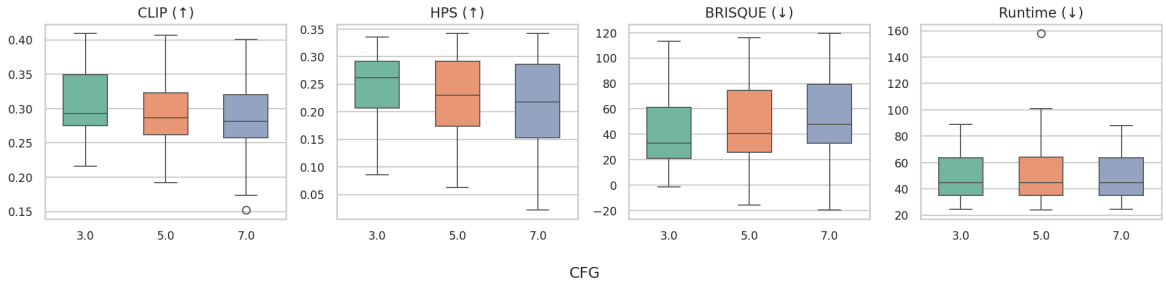


Figure 4.4: Distribution of each outcome metric grouped by guidance scale (CFG).

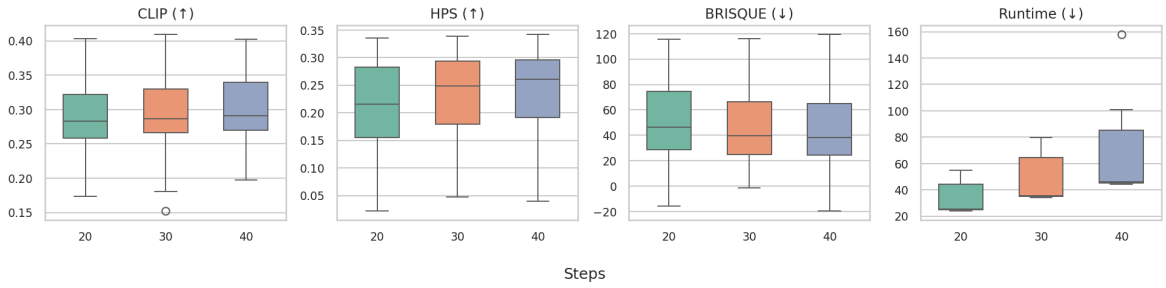


Figure 4.5: Distribution of each outcome metric grouped by inference step count.

We show in Figure 4.4 that raising CFG steadily trades quality away and, at the same time, widens the spread of outcomes. Stronger guidance pushes the model harder toward the prompt but makes its behavior less stable, the familiar quality–diversity trade-off. In Figure 4.5, adding inference steps improves quality with clearly diminishing returns. The jump from few to moderate step counts buys most of the available gain, and further steps add little beyond a point, while each step costs a fixed amount of time regardless of the other settings. Because the benefit of steps saturates but their cost does not, there is a natural sweet spot rather than a simple "more is better" rule.

Taken together, the results decompose along two axes. Quality is determined primarily by the scheduler, and runtime by the sampler and step count, with CFG and step count providing finer adjustment within each axis. Repeating the analysis separately within each prompt category confirms that this structure is not an artifact of averaging across

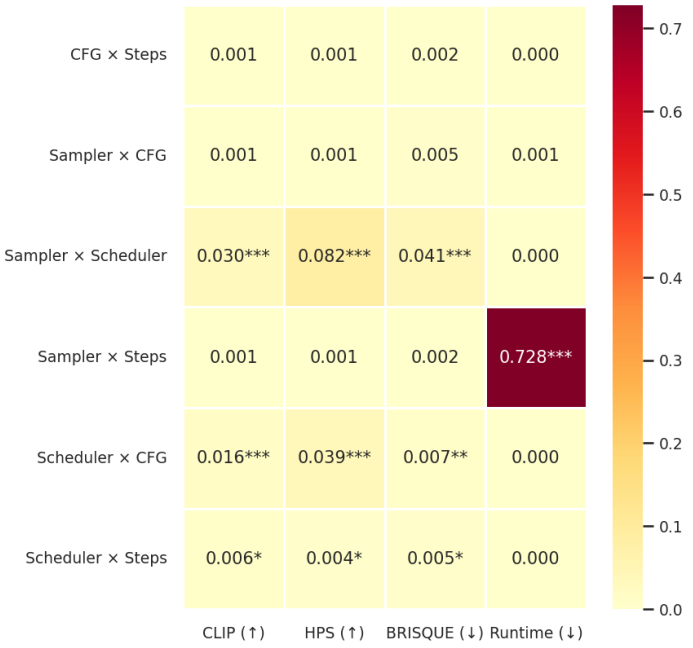


Figure 4.6: Two-way ANOVA partial η^2 for every parameter pair on every metric. Stars denote significance levels: * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$; unmarked cells are not statistically significant.

prompt types (see Appendix A.2, Figure A.1): the scheduler dominates human-preference quality in every category, and runtime remains parameter-driven throughout. The one genuine exception is worth noting, since it is the only place the global ranking breaks: for composition-heavy prompts, perceptual quality is shaped more by CFG than by the scheduler. We read this cautiously as a hint that guidance strength matters most when a prompt's difficulty lies in arranging elements correctly, but the effect is confined to a single category and metric and we do not lean on it further.

Summary: Image quality depends primarily on the scheduler, whereas runtime depends on the sampler and the number of inference steps. Guidance scale and step count contribute only minor, gradual adjustments, and this division remains stable across all prompt categories.

Interactions between parameters

Figure 4.6 shows the two-way ANOVA partial η^2 for every parameter pair. Most pairs interact negligibly, so their effects simply add up. The lone strong interaction and the single largest effect of any kind in the experiment is between sampler and step count on the runtime axis ($\eta^2 = 0.728$). This one is mechanical rather than surprising, because the slow samplers do more work per step, adding steps costs them more wall-clock time than it costs the fast samplers, so the two parameters compound on the runtime axis. On the quality axis, the only interactions of any practical size involve the scheduler, and even these are modest.

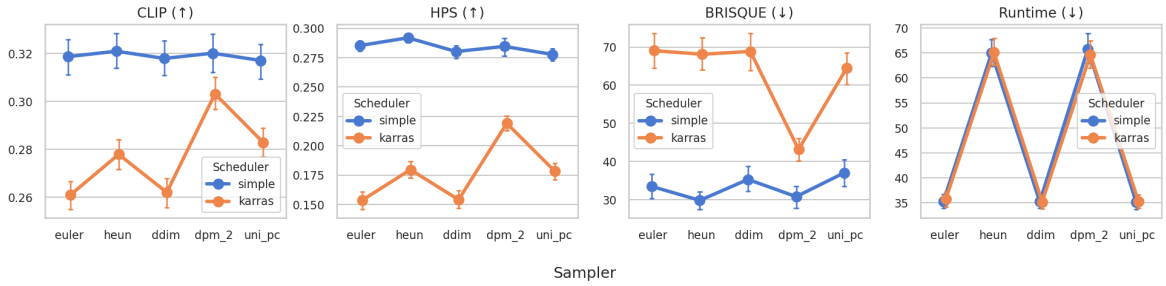


Figure 4.7: Interaction between sampler and scheduler.

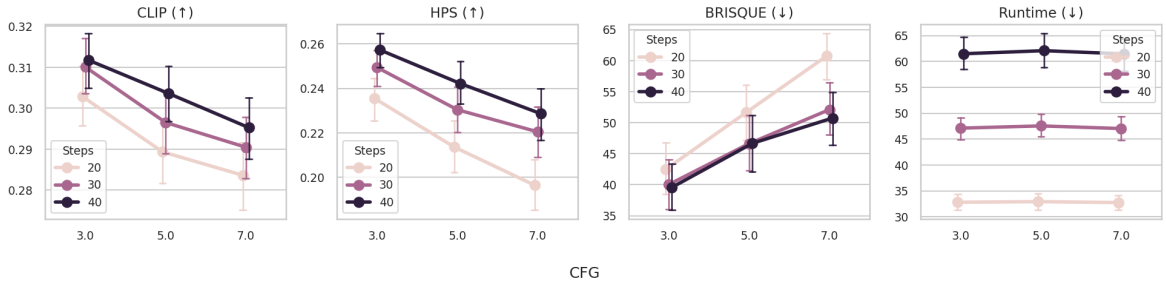


Figure 4.8: Interaction between CFG and inference steps.

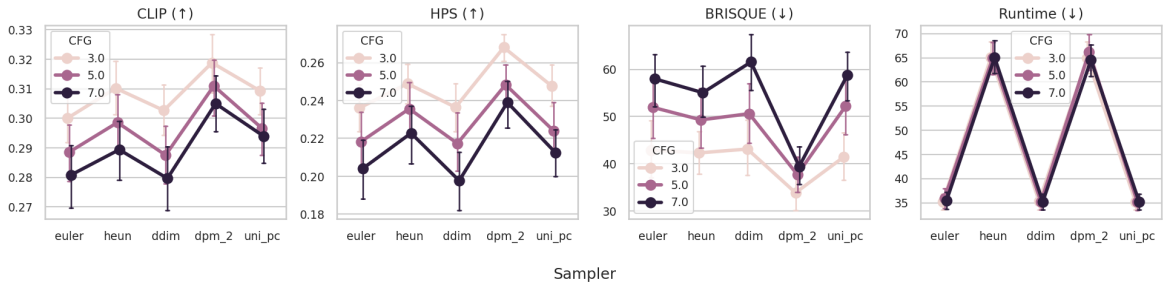


Figure 4.9: Interaction between sampler and CFG.

Figure 4.7 shows the interaction between sampler and scheduler, the strongest on the quality axis. Under `simple`, the choice of sampler is almost irrelevant: every sampler produces comparably high quality. It is only under `karras` that the samplers pull apart, and there `dpm_2` is the one sampler that partly resists the collapse in quality that the others suffer. The interaction therefore does not overturn the main finding so much as qualify it: sampler choice matters chiefly as damage control once a poor scheduler has already been selected, and is close to a free choice otherwise. The remaining pairs are essentially additive. CFG and steps influence quality independently, and runtime continues to depend on steps alone (Figure 4.8); sampler and CFG likewise combine without surprises, the only nuance being that `dpm_2` is notably more tolerant of high guidance than the other samplers (Figure 4.9), a second sign of the robustness already visible in its marginal distributions.

Once the scheduler is fixed to `simple`, the quality and runtime axes do not compete, almost no parameter pair needs to be tuned jointly, and the remaining parameters can be adjusted along largely independent, well-behaved gradients. A search procedure can

therefore exploit this near-decomposability rather than treating all ninety combinations as equally plausible and unrelated.

Summary: The parameters influence the outcomes largely independently of one another. The only substantial interaction occurs between the sampler and the number of inference steps, and it affects runtime alone. Once the scheduler is fixed, the remaining parameters can therefore be tuned separately, which considerably simplifies the search for well-performing configurations.

4.1.3 Discussion

The most consequential finding is that, within the parameter levels we tested, the four-dimensional configuration space decomposes almost cleanly into two parts: image quality is dominated by the scheduler, whereas runtime is dominated by the sampler and the number of inference steps, and the two groups do not measurably interact. This decomposition should not be read as a universal property of the workflow, since outside the tested scope the parameter dependencies may differ substantially. For a user tuning manually, this collapses the search into a small number of consequential decisions. The scheduler is the choice worth thinking about first, the sampler and step count are the levers to trade runtime against quality, and CFG is a fine adjustment that pulls quality one way at a modest cost in variance. A user who selects a well-suited scheduler within the first few trials has done most of the work, and the remaining choices need not be revisited when only runtime is being changed. For an automated optimizer, the same decomposition means that prior knowledge of the good scheduler frees most of the trial budget for runtime tuning rather than quality search, which is the empirical premise on which the RQ2 comparison rests.

A second point concerns how quality is measured, and it matters for both automated pipelines and users. CLIP responds to parameter changes far less than HPS or BRISQUE do, so a stopping criterion or an acceptance test that relies on CLIP alone will look flat even when the image genuinely improves, and an optimizer whose objective is CLIP will conclude that many configurations are roughly equivalent. HPS carries the strongest parameter signal in this workflow and is the metric to watch during tuning, while BRISQUE is a useful independent check but has more irreducible variance and reacts less predictably to individual choices. For this reason, we treat HPS as the primary quality signal in the remainder of the thesis, and the same choice is a reasonable default for a user or optimizer working within a comparable setup.

For composition-heavy prompts, perceptual quality is shaped more by CFG than by the scheduler, which suggests that guidance strength may matter most when a prompt's difficulty lies in arranging elements correctly. In practice, this means that choosing the right scheduler first is a good default but not universal, and a workflow used mostly for composition-heavy prompts may benefit from treating CFG, as the first decision on the quality axis. We read this cautiously, as the effect is confined to a single prompt category and a single quality metric, and the factorial study is not designed to isolate such prompt-dependent effects with precision. Future work could use the prompt itself to decide which

metrics and parameters to prioritize, rather than applying one fixed rule across all prompt categories.

4.2 RQ2

Building on the configuration space characterized in RQ1, RQ2 evaluates whether the agentic optimizer proposed in Chapter 3 can navigate that space more effectively than conventional search strategies. Because RQ1 showed that the good region of the space is narrow and largely determined by few parameters, the question is not whether the methods can eventually find good configurations but whether a method that already knows this shape converts that knowledge into faster and more focused search than methods that must discover it from scratch. This shifts the comparison away from ceiling quality and toward convergence and search behavior, with the further expectation that differences between methods will surface on the parameter-sensitive metrics identified in RQ1. We compare the proposed agentic optimizer against three baselines: random search, grid search, and a TPE sampler as implemented in Optuna [56], by looking at achieved quality, convergence speed, and how each method distributes its trial budget across the parameter space.

4.2.1 Operationalization

We investigate RQ2 through a comparative optimization experiment on the same workflow and prompt set. All four methods mentioned above vary the same four parameters (sampler, scheduler, guidance scale, and inference steps) but differ in the space they draw from and in how they choose the next configuration. Table 4.3 summarizes these differences.

Table 4.3: Candidate space for each method in the RQ2 comparison.

	Random	**Grid**	**TPE**	**Agent**
Sampler	{Euler, Heun, DDIM, DPM 2, UniPC}			
Scheduler	{simple, karras}			
Guidance scale	{3, 5, 7}	{3, 7}	[1.0, 10.0]	[1.0, 10.0]
Inference steps	{20, 30, 40}	{30}	[1, 50]	[1, 50]

Random search draws uniformly from the same discrete space used in the RQ1 factorial study, so it samples without structure but stays within values that RQ1 has already characterized. Grid search evaluates a small deterministic subset of that same discrete space that fills the per-prompt trial budget exactly, so the budget is spent on a deliberate cover rather than on revisiting points. These first two baselines are the conventional reference points for evaluating an optimization method under a small trial budget. Both are model-free, assume nothing about the shape of the response surface, and are the strategies against which sample-efficient search is commonly assessed [5]. TPE is included as a stronger, sample-efficient reference from the Bayesian optimization family. It fits a probabilistic surrogate over the parameter space from prior trial outcomes and selects the next configuration by maximizing expected improvement, which lets us contrast the agentic optimizer not only against model-

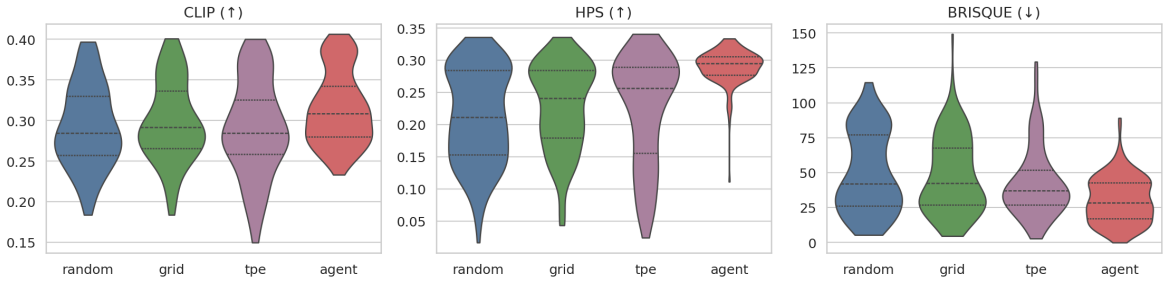


Figure 4.10: Marginal distributions of the three quality metrics per search method across all trials.

free search but also against an established sample-efficient method. It optimizes the same primary metric (HPS v2) that the agent's system prompt targets, uses five startup trials, and replays completed trials from the database on resume so its evidence base matches the other methods. TPE and the agentic optimizer both relax the numeric parameters to continuous ranges, so they can propose values that were not directly evaluated in the factorial study. Each method receives the same per-prompt trial budget of 20 evaluations and runs over all prompts under the same metric collection procedure as in RQ1.

We evaluate performance along three axes, beginning with end-of-budget quality: the best-found HPS v2, CLIP Score, and BRISQUE, together with the number of trials each method needs to first beat the default configuration. The second axis is anytime behavior, which we capture through per-prompt best-so-far curves, their normalized area, and a time-to-target measure. The third axis is search behavior, for which we report a Gower-distance exploration/exploitation curve alongside mean regret over time. The central question is whether the agentic system reaches better or more context-appropriate configurations than the baselines within the same evaluation budget.

4.2.2 Results

In RQ2 we compare four search strategies over the configuration space characterized in RQ2: RANDOM SEARCH, GRID SEARCH, TREE-STRUCTURED PARZEN ESTIMATOR (TPE), and the proposed AGENTIC OPTIMIZER. Each method is given a budget of 20 trials per prompt across the same 8 prompts (composition-heavy, multi-constraint, realism-oriented, and stylized), for a total of 160 trials per method. The *default* configuration is run once per prompt and serves purely as a reference baseline for the trials-to-beat comparison. We evaluate each method on the three quality dimensions established in RQ1 (CLIP, HPS, BRISQUE).

Overall search performance across methods

Figure 4.10 shows the marginal distribution of each quality metric for every method. On CLIP the four distributions overlap almost entirely, so all methods span the same range of prompt-alignment scores. The picture shifts on HPS, where the agent's distribution is visibly narrower and sits closer to the high-quality mode, while other methods retain a heavier low-HPS tail. On BRISQUE, where lower is better, the agent again concentrates near the good end of the range and truncates the long poor-quality tail that the other methods still exhibit. This is the first sign that the differences between search strategies show up chiefly on the

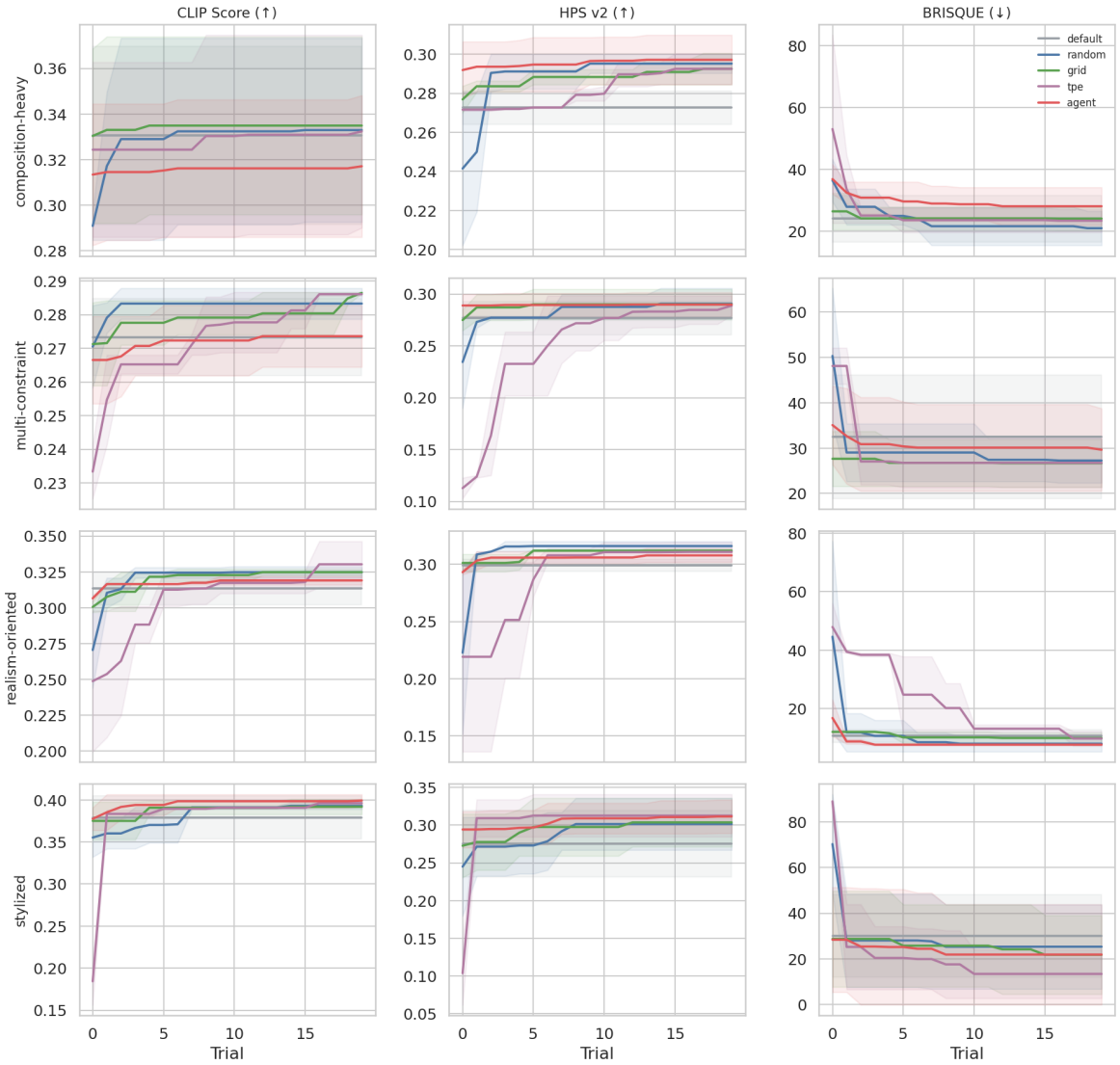


Figure 4.11: BSF curves grouped by prompt category.

HPS/BRISQUE axes rather than on CLIP, which is consistent with the RQ1 finding that CLIP is the least parameter-sensitive of the three quality metrics.

Marginal distributions summarize the full trial history but say nothing about how quickly a method reaches its best score. Figure 4.11 therefore plots the Best-so-far BSF trajectory of each method against trial, classified by prompt category, with bootstrapped 95 % confidence intervals. The three quality axes give a consistent picture, and it holds in every category: the methods separate for the first few trials and then flatten into a common band, reaching a plateau that is essentially indistinguishable within the confidence interval, with plateau values that agree across methods to within a few percent. TPE is the one method that starts noticeably lower and takes several trials to catch up, most visibly on HPS and CLIP. This delay reflects TPE's cold start. Before its model can guide the search, the estimator needs a handful of observations to work from, so its opening trials are effectively random. Those trials are also spread over a far larger space than random and grid search face, since TPE

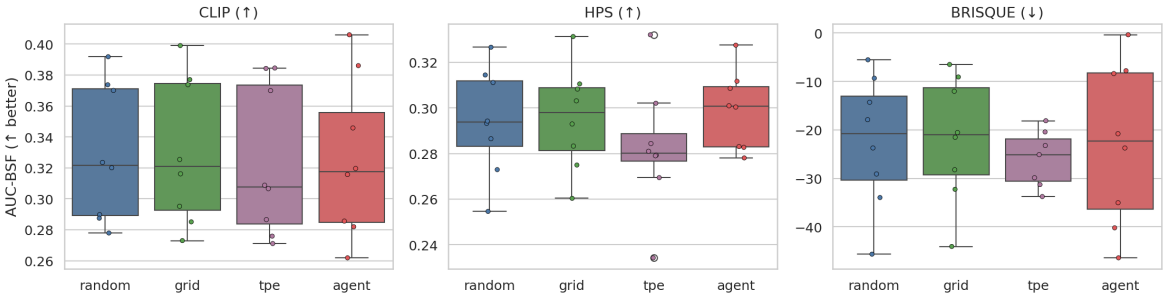


Figure 4.12: Direction-aligned AUC-BSF per prompt, one panel per quality metric.

treats guidance scale and step count as ranges rather than as three fixed levels each, so more of its budget is gone before its model becomes useful. Once the plateau is reached, the ranking between random, grid, and agent is essentially a statistical tie. Two category-level nuances are worth flagging. First, the agent's early-trial advantage on HPS is largest on multi-constraint and realism-oriented prompts, precisely the categories that RQ1 identified as the hardest to align. Second, on stylized prompts, all methods including TPE reach the plateau almost immediately, which suggests that when the good region of the space is broad, the choice of search strategy matters little. This is the same pattern as in RQ1, only viewed through the lens of search behavior rather than parameter effects.

Summary: All four methods converge to the same final quality, agreeing to within a few percent on every metric in every prompt category, so the ranking between them is about speed rather than ceiling. On that axis the agent is the strongest. It sits at or near its plateau on HPS from the first trial, with its largest early margin on multi-constraint and realism-oriented prompts. TPE is the weakest, losing several trials to its cold start before its model becomes informative.

Convergence speed and time to reach the baseline

The BSF curves show that all methods eventually converge, but they say little about how efficiently they use their budget along the way. We therefore compute the direction-aligned area under the BSF curve (AUC-BSF), which rewards methods that reach high scores earlier in the trial sequence rather than only at the end (Figure 4.12). The picture is again one of practical equivalence on CLIP: the four per-prompt distributions overlap heavily and their medians differ by less than a percentage point. On HPS the agent has the highest median AUC-BSF, with random and grid close behind and TPE clearly mirroring the cold-start delay visible in the BSF curves, and the tightest interquartile range. On BRISQUE the agent, random and grid are essentially tied, all slightly ahead of TPE.

Figure 4.13 compares the methods prompt by prompt, without averaging. It is built in two steps. First, for each method and prompt we take the final BSF value, that is, the best score that method reached on that prompt within its 20 trials. This gives eight numbers per method and metric, one per prompt. Second, we plot those numbers as a step function whose height at a given score is the fraction of prompts that ended at or below it. A method that does better on more prompts therefore sits further to the right. The empirical CDFs

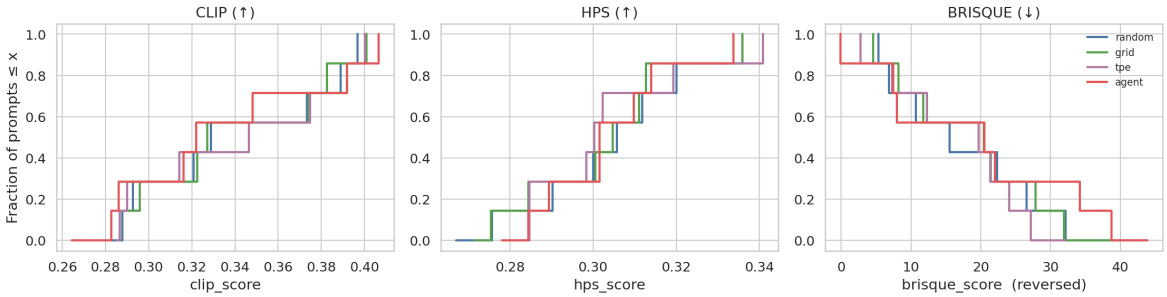


Figure 4.13: Empirical CDF of the final BSF score per prompt. Each method traces a step function whose height at a given value is the fraction of the eight prompts scoring at or below it.

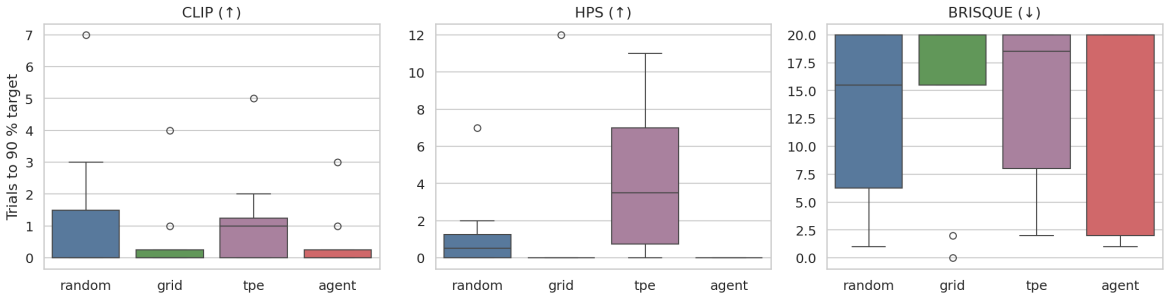


Figure 4.14: Trials required to reach 90 % of the best score any method achieves on the same prompt, per method and quality metric.

of random, grid, TPE, and the agent lie almost on top of each other on CLIP and BRISQUE, and are very close on HPS with the agent shifted slightly to the right (better) on the upper part of the distribution. This confirms that the ceiling of the search space is reachable by any of the four strategies given enough trials, but leaves the question of *how quickly* that ceiling is reached open. We answer this with the time-to-target statistic in Figure 4.14, which reports the number of trials each method needs to reach 90 % of the best score any method achieves on the same prompt. On CLIP and HPS the medians for random, grid, and agent are effectively zero, meaning the very first trial is already within 10 % of the eventual best for most prompts, whereas TPE's cold start pushes its median to 4–5 trials. BRISQUE is the outlier: the 90 % target proves hard for every method, with medians in the mid-teens and heavy censoring at the budget, indicating that BRISQUE has substantially more irreducible variance than CLIP and HPS.

A complementary and more directly practical question is how quickly each method surpasses the fixed *default* configuration, since the default is what a user would obtain without any search at all. Figure 4.15 presents the trials-to-beat-default distributions as violins. On HPS, all four methods beat the default within a handful of trials on almost every prompt: grid and the agent do so in roughly one to two trials on median, random in around two, and TPE in about four to five. On CLIP and BRISQUE the distributions are much more bimodal: several prompts are surpassed in a single trial, but a non-trivial fraction is never beaten within the budget, producing the mass piled up against the 20-trial censoring line. The agent shows a clear bottom-heavy shape on HPS and BRISQUE, indicating that when it beats the default at all, it typically does so within the first few trials.

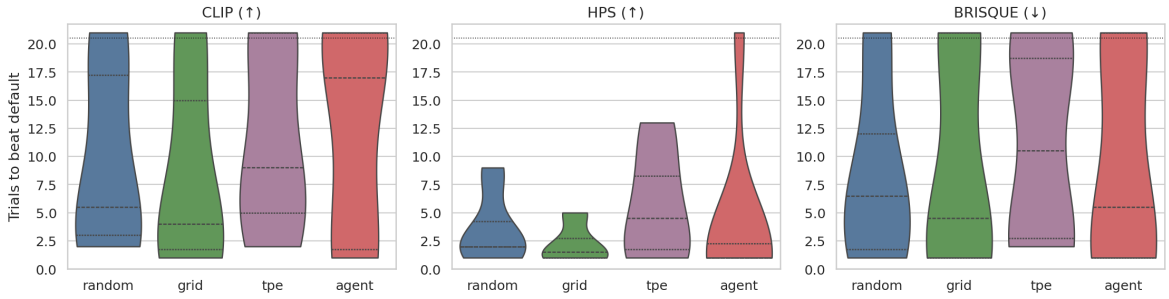


Figure 4.15: Trials to beat the *default* baseline configuration.

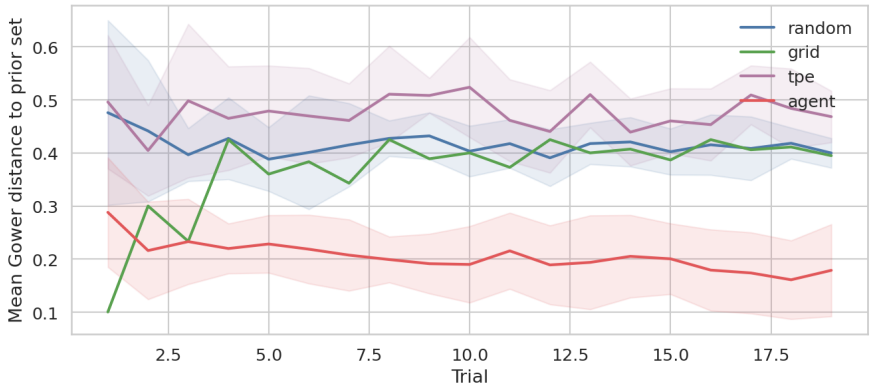


Figure 4.16: Mean Gower distance from each new trial to the set of previously tried configurations.

Summary: All non-TPE methods surpass the default configuration within one to two trials on HPS, and reach 90 % of the best achievable score essentially immediately on CLIP and HPS. TPE pays a consistent cold-start cost of a handful of trials on the categorical space. BRISQUE is the only metric where reaching a near-optimal score is uniformly hard for every method.

Search behavior inside the parameter space

The overall-performance and convergence-speed views tell us *what* the methods achieve, but not *how* they explore the configuration space. Figure 4.16 answers that with the Gower distance [19], a measure of how different two configurations are when the parameters are of mixed type. Two of our parameters are categorical (sampler and scheduler) and two are numeric (guidance scale and step count), so an ordinary Euclidean distance does not apply. Gower handles this by computing a per-parameter distance and averaging across parameters. For the two categorical parameters the distance is 0 if the values match and 1 if they differ, and for the two numeric parameters it is the absolute difference normalized by the parameter's range in the search space. The result is a single number between 0 and 1 that expresses how far apart two configurations are, where 0 means identical and 1 means maximally different across every parameter. For each trial we then compute the mean Gower distance from its configuration to every configuration already tried on the same prompt, and average across the eight prompts. A low value means the method is staying close to

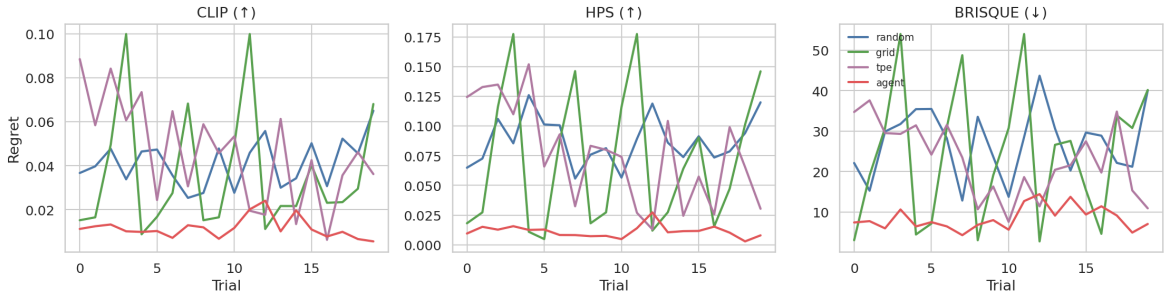


Figure 4.17: Mean regret over time per method.

what it has already seen, and a high value means it is sampling somewhere new. Random and TPE sit at approximately 0.4–0.5 throughout, indicating that each new trial is roughly as different from the history as two uniformly drawn configurations would be. Grid follows the same regime after its first two trials. The agent, by contrast, stabilizes around 0.18–0.22 from very early on and stays there for the remainder of the budget. In other words, the agent concentrates its budget in a smaller, more coherent region of the parameter space than the sampling-based methods do, which is consistent with the narrower HPS and BRISQUE distributions in Figure 4.10.

Focused exploitation is only useful if the region being exploited is a good one. Figure 4.17 shows the mean per-trial regret, i.e. the gap between the current trial's score and the best score seen for the same prompt across all methods. The agent's regret is close to zero throughout on all three quality metrics, whereas random and TPE produce visibly larger and more variable regret across the budget, and grid falls in between. The combined reading of Figures 4.16 and 4.17 is that the agent is not merely narrow in its sampling: it is narrow around the right part of the space. This pattern matches the RQ1 finding that a small number of configuration choices, chiefly the scheduler, controls most of the quality variance: once the agent identifies those choices, it has little reason to keep exploring.

Summary: The agent behaves substantially more exploitatively than random, grid, and TPE, revisiting a much smaller neighborhood of the parameter space, yet incurs consistently lower regret. Its narrower search does not cost quality because the good region of the space is itself narrow, as RQ1 already established.

A note on runtime

We omit wall-clock runtime from the method comparison because the runtime measurements are not directly comparable across methods in the current setup. TPE and, to a lesser extent, random both revisit configurations they have already tried; when they do, the underlying ComfyUI backend returns a cached image in under a second instead of rerunning diffusion, which artificially deflates their reported runtimes. The finding from RQ1 that runtime is a property of the sampler and step count alone holds independently of the search method. None of the four strategies has any mechanism to change how long a fixed configuration takes to run.

4.2.3 Discussion

All four methods end the 20-trial budget at comparable quality, so the agent's gain is limited to the trials it saves. Where a diffusion trial is expensive, that is, large models, high-resolution outputs, or many-step schedules, an approach that saves even a handful of trials per prompt can be worth the additional infrastructure and interpretive overhead that an agent-based system carries. Where a trial is inexpensive, the simpler baselines are the cheaper option once their own overhead is counted, and the agent should not be adopted on the promise of superior configurations, since none of the methods here found configurations the others could not reach. A trial that is taking only a few seconds in our own workflow is inexpensive in exactly this sense. The few trials that the agentic optimizer saves can be offset by the cost of running two language-model agents and an MCP server around the workflow. The choice between agentic and conventional search is therefore best framed as a budget question rather than a capability question, with the crossover point determined by how costly a single trial is in the specific deployment.

The mechanism behind the agent's trial efficiency reframes what an MCP-grounded optimizer of this shape actually contributes. What makes the agent efficient is not a smarter search procedure but the prior context it carries into every proposal, and the MCP layer functions as a warm start rather than as a reasoning component. This gives a concrete rule for when to adopt this kind of system and when to avoid it. It is the right tool when a reliable prior exists and the good region of the space is small enough that exploiting the prior is a valid move, and it is the wrong tool when either condition fails, because the same narrow, confident sampling will then land in a locally acceptable region and leave equally good regions unvisited. Both conditions concern the parameter space and the prior available for it, not the agent that consumes them. If a parameter space has not been characterized sufficiently to identify its high-performing configurations, a method without a prior is more honest than a method that will confidently confirm the wrong one, and this is the failure mode a practitioner should look for before extending the same architecture to a new workflow.

The cold start observed for TPE points in a related direction on the baseline side. TPE needs several observations before its surrogate becomes informative, and at a 20-trial budget those observations consume a substantial share of the evaluations, so a method with a working prior has a consistent advantage in this regime. Which method is preferable therefore depends on the budget and on whether a usable prior exists. A Bayesian sampler can absorb its cold start when the budget is larger or the parameters are continuous, whereas under a short budget over a mostly categorical space that has already been characterized, a warm start or a well-designed grid is the stronger default. The same prior could also be supplied to a conventional sampler instead of an agent. Meta-learning approaches to hyperparameter optimization do exactly this, initializing a Bayesian sampler with configurations that performed well on related tasks so that it does not begin from scratch [15]. Transferring the outcomes of previously optimized prompts in the same way would remove the cold start that costs TPE several trials here, and RQ1 suggests the conditions for such transfer hold, since the parameters that govern quality are stable across prompt categories. We did not test this, and it is the comparison most likely to separate the contribution of the prior from that of the agent carrying it.

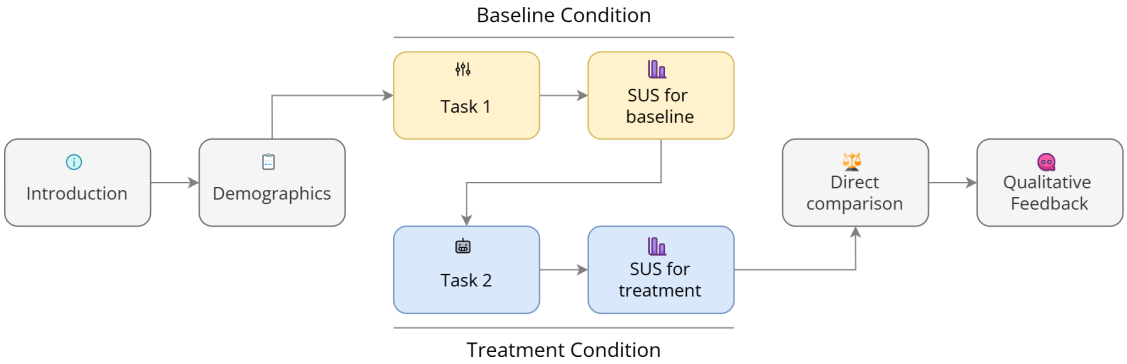


Figure 4.18: Overall procedure of the RQ3 user study.

4.3 RQ3

The trial-efficiency and inspectable-context contributions identified in RQ2 are only meaningful once a human is in the loop, since fewer machine-visible trials need not translate into less human effort, and structured artifacts can as easily be ignored as consulted. RQ3 therefore turns from the optimizer's behavior to its value for the people who use it, and examines through a within-subjects user study whether participants can configure a generative workflow with less effort when the agent assists them, and how helpful they find the traces, evaluation summaries, and explanations that accompany its proposals.

4.3.1 Operationalization

We investigate RQ3 through a within-subjects, task-based user study whose overall procedure is shown in Figure 4.18.

The procedure follows the six stages in a fixed order. After a short *introduction* that briefs the participant on the study, we collect *demographics* on their prior experience with generative image tools. The participant then completes the *baseline condition*, which consists of Task 1 followed immediately by the ten-item SUS questionnaire [6], whose five-point responses combine into a single 0–100 usability score (scores above 68 are commonly treated as acceptable), that captures the perceived usability of that condition. The *treatment condition* follows the same shape: Task 2 and then a second SUS. Afterwards, a *direct-comparison* questionnaire asks the participant to rate the two conditions side by side on a shared scale for mental workload, confidence, workflow understanding, and overall preference, together with a set of Likert items on the agent's explanations and evaluation summaries. A closing *qualitative-feedback* questionnaire captures free-form remarks on how the agent helped and what should be improved.

In the baseline condition, the participant edits the same KSampler parameters investigated in RQ1 and RQ2 manually inside ComfyUI and re-runs the workflow as they normally would. In the treatment condition, the same workflow is presented alongside the optimizer's chat panel, which surfaces the structured traces, evaluation summaries, and explanations

generated by the agent, and participants can send free-form corrective feedback between trials that the MCP-grounded configuration agent incorporates into its next proposal. Both conditions are hosted on the same browser-accessible ComfyUI instance and share the same underlying workflow and model checkpoint, and the two tasks use two different prompts of comparable difficulty. In both tasks, we intentionally set the KSampler parameters away from the workflow's defaults, since the default configuration already produces images of acceptable quality and would leave little headroom to observe where users struggle to reach a good result. At the end of the baseline condition, the final trial is treated as the participant's best attempt; at the end of the treatment condition, the participant is asked to select the best trial produced during the agent's session. These selected trials, along with all preceding trials from both conditions, are stored for subsequent analysis.

Data collection is organized around the two parts of RQ3: participant *effort* and perceived *usefulness*. For *effort*, the recorded sessions and the database record of every trial and message give completion time, the number of trials a participant runs before settling, and the final parameters chosen. For *usefulness*, the two per-condition SUS totals give a usability rating for each condition, and the direct-comparison and agent-evaluation items give condition-comparison and agent-specific ratings that can be paired against the objective log. The database keys every trial and every chat session to an anonymized participant identifier, so baseline and treatment trials from the same participant can be paired directly during analysis. The target sample consists of 13 participants recruited to span a range of prior experience with generative image tools. The demographics questionnaire captures this experience along three axes that includes years of hands-on use of generative image tools, self-rated familiarity with the diffusion parameters, and ComfyUI usage frequency.

4.3.2 Results

We recruited the target sample of 13 participants and report the outcomes below, moving from cohort demographics to what participants objectively produced in each condition, and then to how the process felt to them, first through perceived usability, then through direct condition comparisons, and finally through the items that target the agent's own behavior. Given the modest sample size, we read the direction and magnitude of each effect as the primary evidence, and treat significance values as a secondary check.

Participant demographics

Figure 4.19 summarizes the cohort. ComfyUI usage frequency spans the full spectrum, from a single participant who had never opened the tool to two daily users, with the plurality reporting only rare use. Self-rated familiarity with diffusion parameters (sampler, scheduler, guidance scale, inference steps) is broadly distributed around a central value of three on a zero-to-five scale (0 = none, 5 = expert), with a small number of participants at either extreme. Years of experience with generative image tools cluster between one and five years and are effectively bimodal, with a group of relative newcomers and a group of longer-term users. This spread matters for the analysis in two ways. It ensures the study is not confined to a single expertise stratum, and it lets us later relate individual differences in prior experience to how each participant evaluated the assistant.

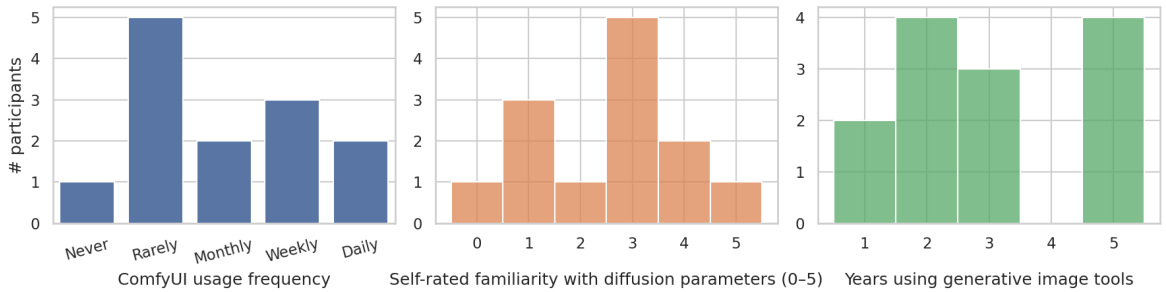


Figure 4.19: Distribution of prior experience across the 13 participants.

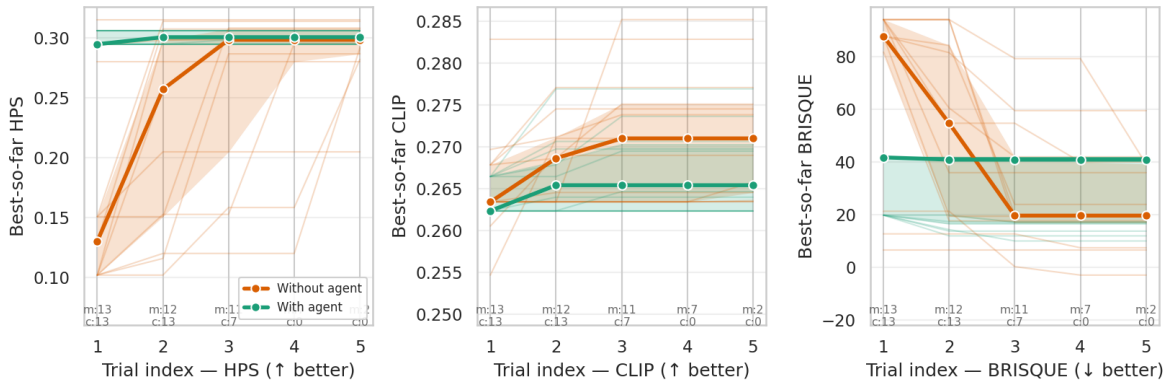


Figure 4.20: Best-so-far quality per participant over the trials of each condition, aggregated as median with interquartile band.

Objective performance: convergence and final configuration

We begin with what participants objectively produced in each condition, since the rest of the study is best read against a clear picture of the artifact-level differences. Figure 4.20 plots best-so-far quality per participant, aggregated across participants, for the three quality metrics used throughout the thesis. The treatment condition begins essentially at its plateau from the very first trial on all three metrics, whereas the baseline condition climbs across two to three trials before catching up, particularly on HPS and BRISQUE. By the third trial the two conditions reach comparable quality on all three metrics, and the small subset of participants who ran a fourth or fifth trial in either condition does so from an already-converged position. This is consistent with the search-behavior finding in RQ2. When the agent brings prior context to the task, its first proposal is already close to a good region of the configuration space, and additional trials serve mainly to fine-tune within that region rather than to search.

Convergence speed also shows up in how many trials participants used before settling. Median trials-to-settle drops from four in the baseline condition to three in the treatment condition. Per-trial runtime, by contrast, is longer in the treatment condition. Mean per-trial wall time rises from 38.9 s in the baseline condition to 64.7 s in the treatment condition, because the agent tends to propose configurations with heavier samplers and more inference steps than participants default to. The two effects roughly cancel on total time in front of the tool, which is comparable across conditions. Final quality metrics do not differ between

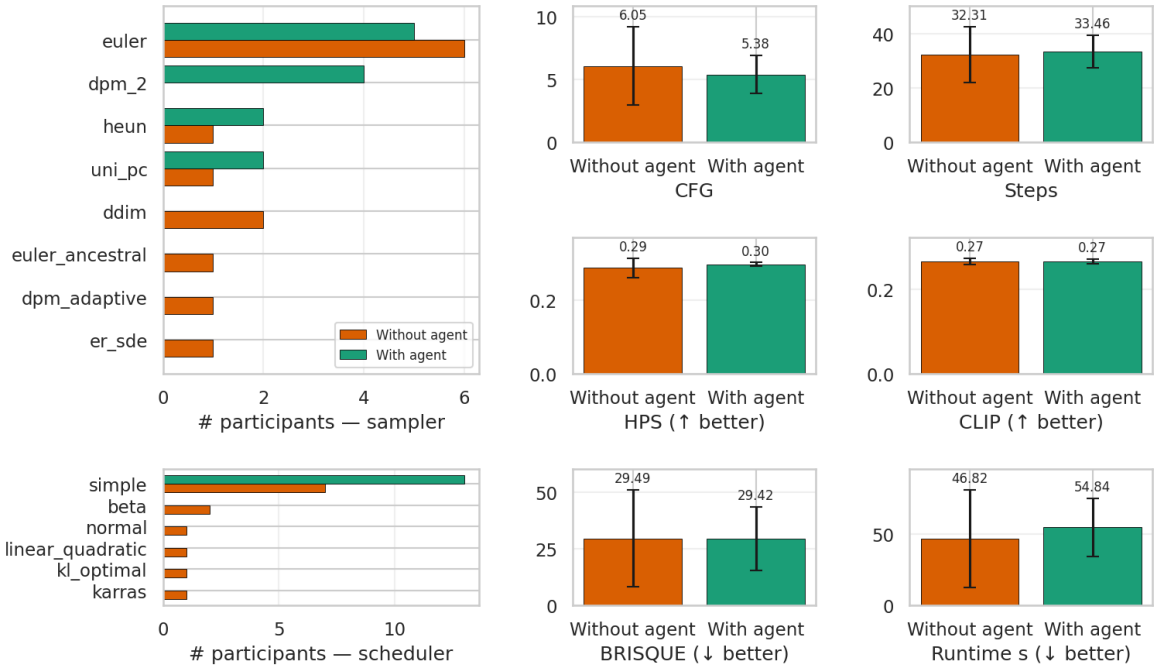
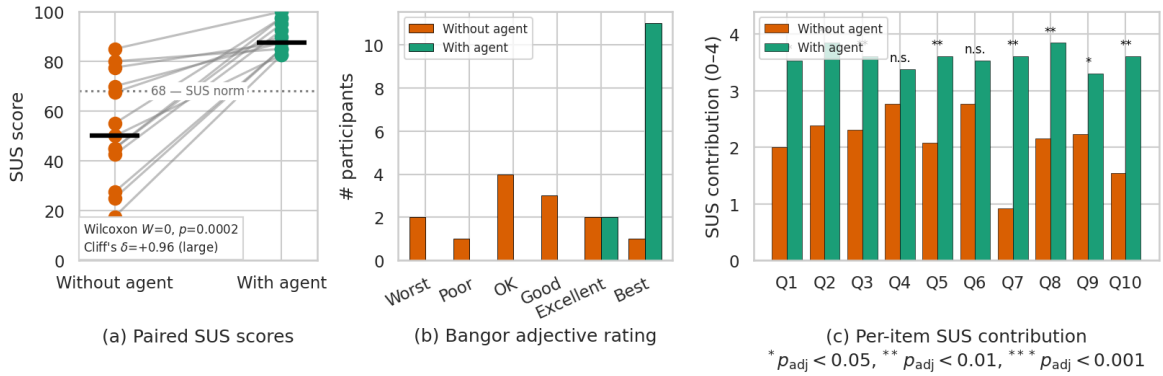


Figure 4.21: Final configuration selected by each participant, per condition.

conditions after correction, on any of HPS, CLIP, or BRISQUE, so the agent does not produce measurably better images at the metric level under the same trial budget.

Figure 4.21 inspects the configurations that participants ultimately selected as their final result. The most striking difference is on the scheduler. Every one of the thirteen with-agent trials converges to the `simple` scheduler, whereas the baseline condition scatters across six different schedulers (`simple`, `beta`, `normal`, `karras`, `kl_optimal`, `linear_quadratic`). Because RQ1 identified the scheduler as the single largest driver of HPS, an agent that internalizes that finding will concentrate its scheduler choice on the values known to be strong, while a participant reasoning from scratch will explore. The sampler distributions diverge in a related but weaker way. Manual runs cluster on `euler` with a long tail of one-off choices, while the with-agent runs distribute across `euler`, `dpm_2`, `heun`, and `uni_pc`. The continuous parameters (CFG and inference steps) converge to similar means but with markedly smaller variance in the treatment condition, indicating that the agent is not merely picking a different working point but is picking a more consistent one across participants. The quality metrics agree with the BSF plateau. HPS, CLIP, and BRISQUE are essentially equal between conditions on average, while runtime is systematically longer in the treatment condition for the reasons already noted.



(a) Paired SUS scores (b) Bangor adjective rating (c) Per-item SUS contribution

$^{*}p_{adj} < 0.05$, $^{**}p_{adj} < 0.01$, $^{***}p_{adj} < 0.001$

Figure 4.22: Perceived usability by condition: (a) paired SUS totals per participant with the Wilcoxon signed-rank and Cliff's δ annotated in the panel, (b) Bangor adjective ratings [3] selected per condition, and (c) per-item SUS contribution with Benjamini–Hochberg-corrected paired Wilcoxon significance marked above each item pair.

Summary: Participants reach comparable image quality in one to two fewer trials on median in the treatment condition, and the quality trajectory across trials is steeper on all three metrics. Final quality is statistically equal between conditions. The agent's advantage is in reaching a comparable image sooner and with a substantially more consistent configuration. Per-trial runtime is longer in the treatment condition because the configurations the agent proposes use heavier samplers and more steps.

Perceived usability (SUS)

Because the objective differences between conditions are limited to convergence speed rather than final quality, the interesting question is how the process felt to the participants who lived it. The paired SUS totals in Figure 4.22(a) show a large and unidirectional shift between the two conditions, significant under a paired Wilcoxon signed-rank test [57], a non-parametric alternative to the paired t-test that does not assume normal differences, with a large Cliff's δ [9], an ordinal effect size in $[-1, +1]$ that can be read as the probability that a random observation from one condition exceeds one from the other. In the baseline condition the median score is 50.0 and only four participants clear the widely used 68 acceptability threshold, whereas in the treatment condition the median rises to 87.5 and every participant lands well above the threshold. The per-participant deltas are strictly positive, with a median improvement of $+42.5$ points, and there is no participant for whom the treatment condition scored lower. Figure 4.22(b) confirms the same movement through the coarser adjective mapping of Bangor et al. [3], which associates SUS score bands with descriptors ranging from *Worst imaginable* to *Best imaginable*. In the baseline condition the adjectives are distributed across the negative half of the scale, with seven of the thirteen participants selecting one of *Worst imaginable*, *Poor*, or *OK*, whereas in the treatment condition eleven of the thirteen selected *Best imaginable* and the remaining two selected *Excellent*. The per-item breakdown in Figure 4.22(c) attributes the totals rather than concentrating them on one or two items. All ten SUS items shift in the same direction, and no item in the treatment condition falls below 3.3 out of 4. Eight of the ten items reach

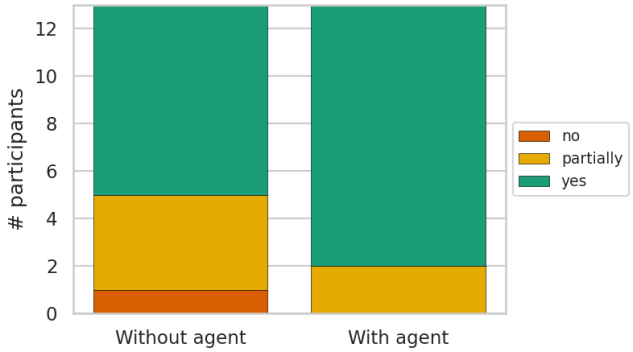


Figure 4.23: Post-task satisfaction with the produced image.

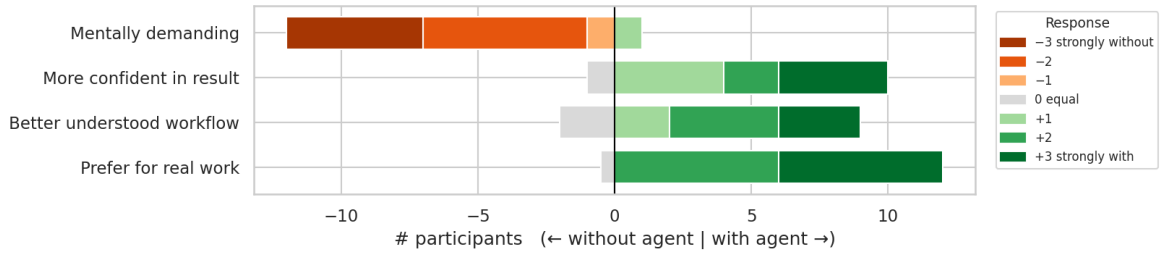


Figure 4.24: Direct choice comparison items. Negative values (orange) indicate the response leans toward the without-agent condition; positive values (green) toward the treatment condition; zero denotes no difference between conditions.

significance under a paired Wilcoxon signed-rank test with Benjamini–Hochberg correction [4], which controls the false discovery rate across the ten simultaneous tests; the two items that do not, on *needing technical support* and *inconsistency*, still shift in the expected direction. The largest gaps appear on the items about learning curve, that is, how quickly the tool can be learned and how much must be learned before getting going, which are precisely the items most sensitive to whether a tool provides scaffolding. The pattern is therefore consistent with the agent lifting the entire usability profile rather than affecting an isolated aspect of the interaction.

Summary: SUS scores rise from a median of 50.0 in the baseline condition to 87.5 in the treatment condition, all thirteen participants improve, and the improvement is spread across every item. All participants score in the top adjective grade in the treatment condition, compared with a majority in the bottom half in the baseline condition.

Task perception and forced-choice comparison

The post-task satisfaction results in Figure 4.23 agree with the SUS pattern. In the baseline condition, eight of the thirteen participants reported being satisfied with the image they had produced, four were partially satisfied, and one was not. In the treatment condition, eleven reported satisfaction and the remaining two were partially satisfied, with no dissatisfied responses. Satisfaction therefore rises but does not do so from a low base, which is consistent

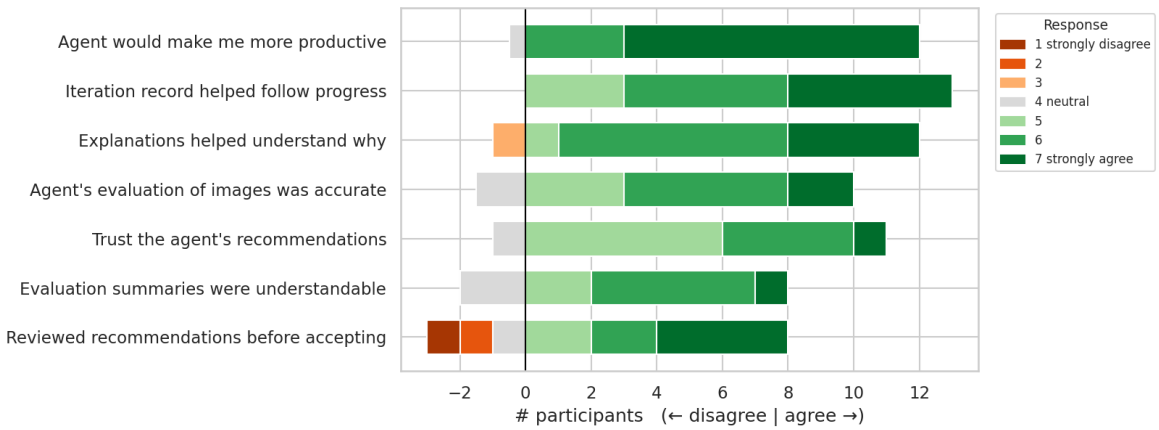


Figure 4.25: Response distribution across the seven agent-only Likert items, ordered by mean rating.

with the study prompts being difficult enough to expose the interaction differences without preventing participants from producing an acceptable image in either condition.

The items in Figure 4.24 probe the two conditions on a shared -3 to $+3$ scale, where a negative value favors the baseline condition and a positive value favors the treatment condition. All four items shift consistently in favor of the assistant. Perceived mental demand is the sharpest movement in the study. Twelve of the thirteen participants rated the baseline condition as more mentally demanding, with a median response of -2 and no participant rating the treatment condition as more demanding than the baseline. Confidence in the produced result and self-reported workflow understanding both shift toward the assistant with medians of $+1$ and $+2$ respectively, and preference for real work reaches a median of $+2$, with twelve of the thirteen participants preferring the assistant for their own work. The confidence and understanding items merit a small caution. Neither is a direct measure of what the participant actually knew or achieved, but is a self-report at the end of a short task. Read together with the SUS deltas, they still describe a coherent shift, that participants experienced the treatment condition as lower effort and more legible than the baseline condition and would prefer it for their own use.

Summary: Every item favors the treatment condition significantly and in the same direction, with the largest movement on perceived mental demand (median -2 toward the baseline condition). Post-task satisfaction also rises, from an eight-of-thirteen majority in the baseline condition to eleven of thirteen in the treatment condition, and no participant reports dissatisfaction with the agent-produced result.

Perceived quality of the agent's behavior

The seven items asked only after the treatment condition target the agent's own behavior. Figure 4.25 shows the item-level response distribution ordered by mean rating. Six of the seven items sit above the neutral midpoint of four. The clearest results are on productivity (mean 6.54, twelve of thirteen agreeing), the helpfulness of the iteration record (mean 6.15, all thirteen agreeing), and the helpfulness of the explanations (mean 6.00, twelve agreeing).

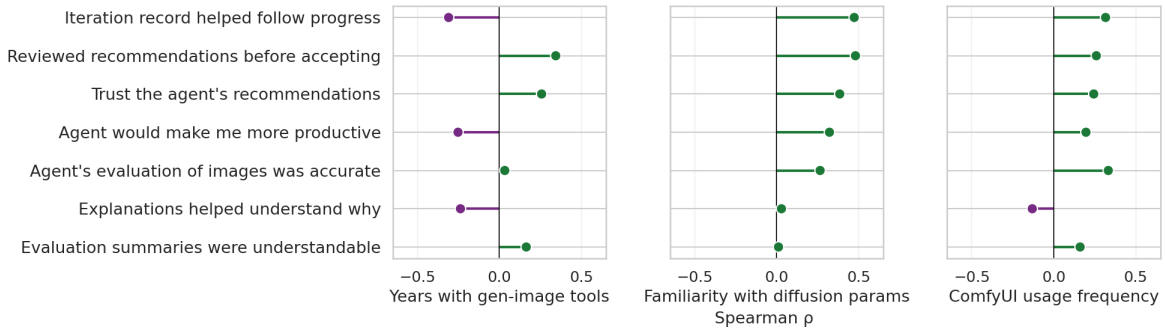


Figure 4.26: Spearman rank correlation between three measures of prior experience and each agent-only item.

Trust in the recommendations and perceived accuracy of the agent's own image evaluation are also clearly positive, with means above five. The only item that does not clear the neutral midpoint is whether participants reviewed the recommendations before accepting them (mean 4.92, three participants disagreeing outright). The ratings are bimodal, indicating that some participants engaged closely with the trace panel while others accepted proposals with limited inspection. The item about whether the evaluation summaries themselves were understandable is the weakest of the significantly positive items (mean 5.00), which reads as a mild readability concern about the numeric summaries rather than about the agent's behavior overall.

Figure 4.26 relates the agent ratings to the three measures of prior experience introduced in the demographics section. No cell reaches significance at $N = 13$, so we read the direction and magnitude of the correlations. The pattern is nevertheless coherent. Self-rated familiarity with diffusion parameters is uniformly positive across items and shows its two strongest Spearman rank correlations on the two items that concern engagement with the artifacts produced by the assistant, that is, following progress via the iteration record ($\rho \approx +0.47$) and reviewing recommendations before accepting them ($\rho \approx +0.48$). Both values sit in the moderate-positive range of the Spearman -1 to $+1$ scale, so participants with higher self-rated familiarity tended to give these two items higher ratings. Participants who already understood the parameter space were also the ones most likely to use the trace panel and vet the agent's proposals. Years of experience with generative image tools shows a small negative correlation on the productivity and explanations items, which reads as senior generative-tool users finding the assistant's guidance mildly less indispensable. ComfyUI usage frequency shows the weakest signal of the three measures, suggesting that what a participant knows about diffusion parameters predicts their engagement with the agent's outputs more reliably than how often they use the surrounding tool.

Summary: Six of seven agent-only items sit significantly above the neutral midpoint after correction, with productivity, the iteration record, and the explanations receiving the strongest endorsement. Participants with higher self-rated familiarity with diffusion parameters engaged most with the trace and evaluation artifacts. Participants with the most years of prior experience found the assistant's guidance least indispensable.

4.3.3 Discussion

The metrics of the selected image do not differ between conditions and the plateau is reached within two to three trials either way, but the perceived cost of getting there is much lower with the agent, and the pattern is unambiguous across every usability item. Studies of these systems that measure only final image quality will underestimate their contribution, because the effect surfaces primarily as a perceived-effort reduction rather than as a metric shift, so workload and usability measures deserve to be treated as primary evidence rather than as supplementary self-report. The adoption question is likewise reframed as not whether the model is a stronger optimizer than the user, but whether the effort saved outweighs the per-trial cost and the additional infrastructure the agent brings with it. In this study, that balance clearly favors the agent because manual tuning is tedious and the per-session inference cost of running both agents is small compared to any plausible labor rate for the same time window (Appendix A.3). It can nevertheless reverse for simple workflows tuned occasionally by an expert, or for one-off configurations whose trials are inexpensive. The clearest implication is therefore that an MCP-grounded agent optimizer of this shape should be evaluated as an effort-reduction tool rather than as a quality-improvement tool.

Some participants inspected the agent's proposals before accepting them and others did not, yet both groups reached comparable image quality. The design implication is that inspection should be made available rather than required, and the two-tier layout used here, a short visible summary with an expandable technical detail, is at least compatible with both groups, since neither had to adopt the other's workflow to reach comparable outcomes. Whether this layout is preferable to alternatives, such as showing the full detail by default or hiding it behind a separate view, is a question the study does not answer. This setting also differs from ones in which an unreviewed AI output can cause real harm, since the worst outcome of an unreviewed proposal here is an image the user can regenerate. A risk nevertheless sits with the non-inspecting group, since they cannot recognize a poor recommendation on their own and must rely on the system to flag when a proposal may be wrong, which RQ3 does not measure. Finally, participants reported better workflow understanding in the treatment condition even though many of them did not inspect the agent's traces before accepting its proposals. The reported understanding is therefore better read as a judgement about interface clarity than as an increase in knowledge about the workflow itself. Future studies should treat these two things as separate: whether the interface feels understandable to a user is not the same as whether the user actually knows more about what the workflow did, and a single self-report item cannot tell them apart.

The usability improvement reported here comes from a single setting: one workflow, one pair of prompts, thirteen participants, and a fixed condition order that mildly favors the treatment. The direction of the effect can be trusted, but its size should not be assumed to transfer to other workflows or user populations without a broader evaluation across workflows, prompt sets, and larger participant samples.

4.4 Discussion

The baseline methods, i.e. random search, grid search, and TPE, make different assumptions about the parameter space. The first two make no assumptions, which makes them robust defaults but each trial is drawn independently of the outcomes already observed. TPE fits a probabilistic surrogate during the run and therefore needs several startup observations before its proposals become informative. The configuration space characterized in RQ1 rewards neither strategy under a short trial budget, since the optimal region is narrow, controlled chiefly by a few parameters, and the shape of the surface is already known once the factorial has been run. The agent exploits exactly this situation. The characterization, the parameter space, and the per-prompt trial history, together with the generated images, are exposed as tools and resources through the MCP layer and can be read by the configuration agent before each proposal and by the evaluator agent before each judgement. The agent's advantage is therefore not a smarter search procedure built into the language model but the context that the protocol places in front of it, and the role of MCP in this system is that of an inspectable context store that warm-starts the agent. Because every proposal is grounded in artifacts that are persisted through the same layer, the inputs the agent saw at any point in a run can be reviewed and reconstructed after the fact, so reproducibility in this system applies to the context presented to the agent rather than to the specific proposal returned. The RQ2 and RQ3 results are consistent with this framing. The agent revisits a markedly smaller neighborhood of the parameter space than any baseline yet incurs lower regret, which is the behavior of a method exploiting a reliable prior rather than discovering structure online, and the same mechanism translates into measurably fewer trials-to-settle and lower perceived workload for participants in the treatment condition. The same mechanism predicts the method's limits. When the parameter space is richer, for example when the workflow adds a refiner stage or control adapters that introduce interactions the factorial cannot observe, or when no factorial characterization is available at all, the agent proposes with the same confidence but on evidence that does not apply, and a prior-free method is the more appropriate choice. The practitioner implication is that the factorial characterization should be treated as the first step rather than an optional baseline, since it supplies both the prior that the agent consumes and the structural facts that decide whether the agent is appropriate at all. Meta-learning initialization [15] would supply the same prior to a conventional sampler, and comparing that setup against the agent would isolate how much of the gain comes from the prior rather than from the agent itself. Finally, because the underlying language model does not expose deterministic decoding, replication should rely on aggregate statistics across prompts and seeds rather than on per-run comparisons.

4.5 Threats to Validity

This thesis faces threats to validity both from the proposed optimization approach itself and from the way we evaluate it. We group them by whether they affect the internal attribution of effects, the constructs we claim to measure, or the generalizability of the findings.

Internal validity. Internal validity concerns whether we can credibly attribute the observed effects to the proposed method rather than to alternative explanations within the experimental setup.

A first threat is that uncontrolled factors such as seed effects or prompt selection may drive the observed differences rather than the agentic optimization method itself. RQ1 averages seed variance into its main-effect estimates by evaluating each configuration under two random seeds. RQ2 and RQ3 use a single seed per trial, so seed variance is not averaged at the per-trial level, but RQ2 aggregates over 20 trials across 8 prompts per method and RQ3 aggregates over 13 participants, so the reported statistics absorb seed noise into their aggregates. The per-prompt **ECDF!** (ECDF!)s in Figure 4.13 further show that the RQ2 ranking holds prompt by prompt rather than only in aggregate, so seed noise would have to flip the ordering across most prompts to overturn the conclusion. Prompt selection is a related concern, mitigated by sampling the eight prompts across the four difficulty categories identified in the factorial study. The main RQ2 findings hold category by category, so a different draw of prompts from the same categories is unlikely to reverse the ranking.

A further threat concerns the automatic metrics themselves, and is sharpest for the agent. The agent's system prompt targets HPS v2 as the primary quality objective, so if it optimizes strongly for that evaluator it may exploit weaknesses of the metric rather than genuinely improve image quality. Two observations limit how far this can go in our data. First, CLIP and BRISQUE are not part of the agent's reward signal, and the agent's final-trial scores on both agree with the prior-free baselines to within a few percent in every prompt category (Section 4.2.2), so a strategy that pushed HPS v2 at the cost of the other two metrics would have shown up and did not. Second, RQ3 participants select the final image themselves and reported being satisfied or partially satisfied with it in every treatment-condition trial, which is a weak but non-zero human signal on perceptual quality. The residual threat is that all three automatic metrics could share the same blind spots, and we did not run an independent human-only quality rating on the RQ2 final images, so an artifact that happens to correlate across HPS v2, CLIP, and BRISQUE would remain invisible here.

A related concern involves the fairness of the RQ2 comparison across methods. As shown in Table 4.3, random and grid search draw guidance scale and step count from the same discrete levels used in RQ1, whereas TPE and the agent draw them from continuous ranges. The agent therefore operates over a strictly larger candidate space than the sampling-based baselines, so any observed advantage in trial efficiency reflects both the method and the space it is free to propose from. The wall-clock runtime measurements collected during the same comparison are similarly confounded, because the ComfyUI backend serves cached images for repeated configurations, as discussed in the runtime note at the end of Section 4.2.2.

Finally, the RQ3 user study administers the baseline condition before the treatment condition in a fixed order, so learning effects may favor the assistant. Participants encounter the workflow, the prompt style, and the interaction surface in the baseline condition first, which means part of the effort reduction attributed to the assistant may instead reflect familiarity acquired in the preceding task.

Construct validity. Construct validity concerns whether the study measures what it claims to measure. The RQ3 direct-comparison items on confidence in the produced result and on workflow understanding are self-reports collected immediately after a short task, so they capture perceived rather than demonstrated understanding and can be influenced by satisfaction with the produced image or by preference for the more polished interface. The objective trial log complements these items with observable outcomes such as best-so-far quality, trials-to-settle, and the final selected configuration, but the interpretation of confidence and understanding as user-facing constructs should be read together with that caveat.

External validity. External validity concerns whether the findings generalize beyond the specific experimental setup used in this thesis.

The reported results depend substantially on the chosen workflow, model checkpoint, and prompt set. All experiments use a single ComfyUI workflow and the Qwen-Image checkpoint with four inference-time parameters, and the prompt set consists of eight prompts across four categories. Different workflows may expose substantially more parameters and different interactions between them, different checkpoints may respond differently to the same optimization strategy, and different prompt sets may favor some forms of improvement more than others.

A related external-validity concern is the size of the RQ3 participant sample. Thirteen participants is modest, and the cohort may not represent broader usage scenarios even though it spans a range of prior experience with generative image tools. Two factors nevertheless strengthen the conclusions drawn from this sample. First, the direction of the effect is unanimous. Every participant's SUS total rises in the treatment condition and no participant reports dissatisfaction with the agent-produced result, so a unidirectional effect of this size is unlikely to be a sampling artifact. Second, the inferential analysis uses paired Wilcoxon signed-rank tests with Cliff's δ and Benjamini–Hochberg correction, which are appropriate for small ordinal samples and account for both the sample size and the number of simultaneous comparisons. Under those tests, eight of the ten SUS items and six of the seven agent-specific items remain significant.

4.6 Future Work

Several directions naturally extend this work. The most direct is to broaden the scope beyond a single ComfyUI workflow and the four inference-time parameters studied here, particularly to workflows with coupled components such as refiner stages, region-specific conditioning, or control adapters, where parameter interactions become first-class rather than negligible. A second direction is to complement the custom MCP server used in this thesis with the official Comfy MCP server that the ComfyUI project itself now ships [13], together with the recently released Comfy Agent that consumes it [12]. Comfy MCP exposes template discovery, node introspection, model-catalog queries, and batch execution through the same protocol layer, so an agent could reason about workflow topology, available custom nodes, and local hardware constraints alongside the parameter and trial context already exposed in our system, instead of operating only over the parameter values inside a fixed

workflow. This would let the same optimization loop generalize across workflows without a per-workflow schema and would align the system with the interface the broader ComfyUI ecosystem is standardizing on. Further work could also extend the evaluation surface with preference signals beyond HPS v2 to mitigate the metric-overfitting concern raised in the threats to validity, and repeat the user study with a larger, counterbalanced cohort to isolate the effort effect from learning and order effects. Together, these extensions would test how far the exploit-a-prior behavior observed here generalizes once the workflow, the evaluator, and the participant population are less tightly constrained than in the present setup.

5

Related Work

This thesis lies at the intersection of agentic systems, configuration optimization, and automatic evaluation. Existing work has not addressed this exact combination directly, but several related lines of work provide important foundations.

Agentic assistance in generative systems. ComfyUI-Copilot [62] presents an LLM-powered assistant for automated workflow development. It combines a planner-style assistant with specialized worker agents for subtasks such as recommendation, debugging, and tutorial support. ComfyGPT [28] similarly adopts a multi-agent design for ComfyUI workflow generation and highlights the importance of handling complex node dependencies and modular graph structures. ComfyUI-R1 [61] further shows that automated workflow generation is not only a generation problem but also a reasoning and correctness problem, since hallucinated nodes, invalid structures, or broken connections can undermine the resulting workflows. More recent work on knowledge-centric agents for ComfyUI [33] develops this direction further by extracting hierarchical representations from real-world workflows and reasoning from task descriptions through intermediate strategies to executable structures, which argues that effective workflow generation requires explicit knowledge grounding rather than direct text-to-JSON conversion. Beyond ComfyUI-specific assistants, PromptSculptor [60] decomposes prompt refinement into specialized agent roles and shows that iterative, role-based improvement can improve text-to-image prompting efficiency. Additionally, VisionCreator [30] is relevant as broader agentic visual-generation work. Although it does not study workflow configuration optimization directly, it shows that planning and structured multi-step reasoning are becoming central capabilities in visual-generation systems. Similarly, AgenticIR [66] studies complex image restoration through a multi-stage agentic system operating over specialized tools. More broadly, recent work on LLM-based agentic systems across domains such as healthcare emphasizes planning, tool use, reflection, memory, and collaboration as core architectural features of agentic behavior [11, 43, 54]. Researchers also explore role-specialized and multi-agent designs in other complex application domains, including recommender systems and coding assistants, studying explicit task decomposition, inter-agent coordination, reflection, and execution-grounded validation [14, 26, 27, 29, 39, 40, 42]. Taken together, these works motivate the use of specialized agents in generative pipelines, while also showing that agentic systems require grounding, validation, and explicit coordination mechanisms.

Tool-augmented agents and structured interaction protocols. A related but distinct line of work studies how LLM-based agents interact with external tools and data sources in a structured way. Early approaches such as ReAct [63] interleave free-form reasoning

steps with tool invocations described in natural language, demonstrating that explicit action–observation cycles can substantially improve agent performance on knowledge-intensive tasks. Toolformer [49] extends this direction by teaching a model to insert tool calls inline during generation through self-supervised fine-tuning, allowing tool use to be acquired without task-specific demonstrations. Subsequent work formalizes the interface further by requiring tools to be declared with explicit input and output schemas, and by surveying the design space of function-calling and tool-use across modern LLM systems [44]. The MCP [1] continues this trajectory by defining a transport-level protocol that separates interaction into typed primitives and by making each of these primitives discoverable and inspectable at runtime. It provides the same uniform grounding mechanism for read-only context for the agents' own system instructions.

Parameter optimization. The closest existing work to this thesis is ComfyGI [50], which applies genetic improvement to ComfyUI workflows and directly mutates parameters such as seed, step count, CFG, sampler, scheduler, denoise level, and model checkpoint. The results are especially relevant because they frame image-generation quality as a search problem over workflow configurations rather than as a fixed property of a model. ComfyGen [17] addresses a related but slightly different problem by predicting prompt-adaptive workflows whose construction can vary model choice, samplers, step counts, and guidance scales. Although it does not perform iterative parameter optimization in the same sense as ComfyGI, it supports the broader idea that generation quality depends strongly on selecting an appropriate workflow and parameterization for a given prompt. Related evidence also appears outside workflow systems. Dynamic Classifier-Free Diffusion Guidance via Online Feedback [38] shows that online feedback can adapt CFG dynamically during sampling, while Align Your Steps [47] shows that sampling schedules themselves can be optimized systematically rather than chosen through fixed heuristics. Along the same lines, Zhang et al. [65] treat timestep selection as a black-box problem and directly optimize target quality metrics through Bayesian optimization, which positions sampling-time hyperparameters as a search problem over actual evaluation outcomes rather than theoretical surrogates and is the closest existing counterpart to the sample-efficient search studied in RQ2. Hyperparameter optimization and AutoML improve model performance through structured search over large parameter spaces under practical resource constraints [20]. In the broader context, this problem connects to research on automated parameter tuning, algorithm configuration, and self-adaptive software systems, where structured search over configurable parameters improves system behavior rather than manual trial-and-error [2, 8, 25, 41].

Multi-dimensional automatic evaluation. Existing work suggests that no single metric is sufficient for text-to-image evaluation, because generated images must often be judged along several dimensions at once, including perceptual quality, semantic alignment, and human preference. The canonical instance of CLIP-based alignment evaluation is CLIP Score [21], which measures the cosine similarity between CLIP embeddings of a generated image and its textual prompt as a reference-free signal for image–text alignment, and is the form of CLIP-derived signal used in this thesis. Building on this idea, CLIP-IQA [55], CLIP Artificial Intelligence-Generated Image Quality Assessment (AGIQA) [52], and HPS v2 [59] show the value of CLIP-based evaluators for capturing semantic, perceptual, and preference-related

signals, while Automatic Evaluation for Text-to-image Generation [53] further emphasizes that evaluation is inherently multi-faceted and benefits from decomposition into multiple sub-tasks. WeGenBench [34] extends this direction with a diagnostic benchmark that annotates each prompt along multiple dimensions and pairs evaluation scores with reasoning trajectories, which supports the view that multi-dimensional evaluation is now treated as a prerequisite for text-to-image optimization rather than an auxiliary check. BRISQUE [37] provides a complementary no-reference IQA perspective by focusing on distortion and naturalness in the image itself rather than prompt alignment or preference. The challenge of automatic evaluation is not unique to image generation. Similar challenges also arise in other generative domains such as open-domain dialogue and text generation, where researchers likewise treat automatic evaluation as a multi-dimensional problem involving several complementary quality criteria rather than a single sufficient metric [31, 32, 35].

6

Concluding Remarks

This thesis set out to ask whether an MCP-grounded multi-agent system can make the configuration of text-to-image diffusion workflows more effective, and less demanding to work with. Modern generative pipelines expose a growing number of controllable inference-time parameters whose combined effects are hard to reason about, and workflow-based environments such as ComfyUI make this complexity explicit without on their own resolving it. To study this question, we designed a system in which a configuration agent and an evaluator agent communicate through a shared MCP server that exposes the parameter space, the reference findings from a prior factorial study, the history of completed trials, and their generated images as typed protocol objects. We integrated the system directly into ComfyUI as a chat-based sidebar and evaluated it through three research questions that separated the structure of the problem, the search behavior of the optimizer, and the effect of the resulting traces on human users.

The three studies combine into a coherent picture. RQ1 showed that quality and runtime are driven by largely disjoint subsets of the four parameters, so the configuration space reduces to two nearly independent problems. Taking that structure as a prior, RQ2 found that the agentic optimizer reaches the same ceiling as the sampling-based baselines in substantially fewer trials, concentrating its budget in a smaller region of the space while incurring lower regret. RQ3 confirmed that the assistant changes the experience of configuring a workflow more than it changes the final image, with usability rising for every participant and perceived mental demand falling sharply while the final quality metrics remained statistically indistinguishable between conditions. The effect is carried by the context the agent reads rather than by MCP itself. The approach is therefore well matched to problems that already admit a small, exploitable prior, and the same architecture may concentrate confidently on the wrong region when the good region is broader or when a factorial prior is unavailable.

A

Appendix

a.1 Evaluation prompt set

The prompt set used in the RQ1 factorial study and the RQ2 optimizer comparison consists of eight prompts divided across four categories. Realism-oriented prompts target natural scenes without strong stylistic constraints. Stylized prompts explicitly request a non-photorealistic visual style. Composition-heavy prompts emphasize spatial arrangement, symmetry, or layered scene structure. Multi-constraint prompts combine many simultaneous requirements, including camera parameters, lighting conditions, and compositional rules, within a single instruction.

Realism-oriented

1. *HD camera, emotional masterpiece, sunset, close-up portrait.*
2. *Ultra wide-angle lens, island under blue sea and sky, sunlight filtering through tree leaves, casting dappled light and shadow.*

Stylized

1. *Tennis female athlete, short hair, white tennis outfit, black shorts, side view returning ball, 3D cartoon style.*
2. *Light watercolor, outside cafe, bright white background, less detail, dreamy, Studio Ghibli.*

Composition-heavy

1. *A grand hallway with perfect symmetry, repeating columns, a person centered at the vanishing point, balanced lighting, architectural photography style.*
2. *Bird's-eye view of a crowded city square, hundreds of people in varied poses, detailed clothing and interactions, strong leading lines from streets, layered foreground–midground–background, balanced composition, soft daylight shadows, cinematic top-down, ultra-detailed realism.*

Multi-constraint

1. *Photorealistic low-angle street scene in New York City at night, slight rain creating glossy reflective pavement, neon signs and warm streetlights mixing color temperatures (3200K–5600K), dense pedestrian traffic with varied clothing and umbrellas, one central subject frozen in sharp focus while surrounding people show motion blur from slow shutter (1/15s), shot on full-frame camera with 50mm lens, f/1.8 aperture, shallow depth of field with creamy bokeh, visible lens distortion and light bloom, ISO 800 subtle grain, cinematic teal-orange color grading, volumetric steam rising from street vents, strong leading lines from crosswalk stripes guiding to subject, rule-of-thirds placement, foreground puddles reflecting lights, midground crowd density, background skyscrapers fading into atmospheric haze, high micro-contrast and realistic skin/fabric textures, HDR dynamic range preserving highlights and shadows, documentary photography style, 8K ultra-detailed.*

2. *Photorealistic close-up scene of a barista preparing latte art in a modern café, stainless steel espresso machine with visible steam pressure, milk pouring captured mid-motion, fine crema texture on coffee surface, warm tungsten lighting (3000K) mixed with soft daylight from window (5500K), realistic shadows and reflections on metal surfaces, shot with 90mm macro lens, f/2.8 aperture, shallow depth of field isolating hands and cup, background softly blurred with identifiable café elements, slight motion blur in pouring milk stream (1/60s), ISO 400 with minimal grain, accurate white balance blending warm and cool tones, rule-of-thirds composition placing cup off-center, foreground includes blurred coffee beans, midground focus on action, background depth with customers and furniture, high micro-detail on skin texture and liquid surface, HDR preserving highlights in steam and shadows under machine, cinematic color grading with natural tones, 8K ultra-detailed, commercial product photography style.*

A.2 Per-category main-effect sizes (RQ1)

Figure A.1 reports the one-way ANOVA η^2 effect sizes computed independently within each prompt category, confirming that the parameter ranking discussed in Section 4.1.2 is stable across categories rather than an artifacts of averaging over them.

A.3 Agent inference cost

To ground the cost discussion in Section 4.3.3, we instrumented per-call token usage separately on the configuration agent and the evaluator agent across one complete twenty-trial RQ2 run. Over the full session the two agents jointly consumed 314,489 tokens (252,124 input, 62,365 output), costing approximately $3.10 at the AWS Bedrock EU regional on-demand rate for Claude Opus 4.6 ($5.50 per million input tokens, $27.50 per million output tokens, as listed under Geo and In-region Cross-region Inference in Europe (Stockholm) on 15 October 2026). The per-trial cost grows with trial index because every proposal replays the per-prompt history through the context window. The treatment condition in RQ3 ran for a median of three trials per participant (Section 4.3.2), placing the inference cost of a



Figure A.1: One-way ANOVA η^2 per parameter and per prompt category, computed independently for each of the four outcome metrics.

typical agent-assisted session near \$0.34, and projecting the measured rate across all eight prompts and twenty trials yields approximately \$25 for the full RQ2 comparison.

Statement on the Usage of Generative Digital Assistants

For this thesis, the following generative digital assistants have been used: We have used Google Gemini for understanding complex terminologies and Claude for brainstorming approaches to the solution and text rewriting. We are aware of the potential dangers of using these tools and have used them sensibly with caution and with critical thinking.

Bibliography

[1] Anthropic PBC. *Model Context Protocol (MCP) Specification*. Version 2025-11-25. Anthropic. 2025. URL: https://modelcontextprotocol.io/specification/2025-11-25.

[2] Rodrigo Araújo and Reid Holmes. "Lightweight self-adaptive configuration using machine learning." In: *Proceedings of the 31st Annual International Conference on Computer Science and Software Engineering*. 2021, pp. 133–142.

[3] Aaron Bangor, Philip Kortum, and James Miller. "Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale." In: *Journal of Usability Studies* 4.3 (2009), pp. 114–123.

[4] Yoav Benjamini and Yosef Hochberg. "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing." In: *Journal of the Royal Statistical Society: Series B (Methodological)* 57.1 (1995), pp. 289–300. DOI: 10.1111/j.2517-6161.1995.tb02031.x.

[5] James Bergstra and Yoshua Bengio. "Random Search for Hyper-Parameter Optimization." In: *Journal of Machine Learning Research* 13.10 (2012), pp. 281–305. URL: https://jmlr.org/papers/v13/bergstra12a.html.

[6] John Brooke. "SUS: A "quick and dirty" usability scale." In: *Usability Evaluation in Industry*. Ed. by P. W. Jordan, B. Thomas, B. A. Weerdmeester, and I. L. McClelland. London: Taylor & Francis, 1996, pp. 189–194.

[7] Ting Chen. *On the Importance of Noise Scheduling for Diffusion Models*. 2023. arXiv: 2301.10972 [cs.CV]. URL: https://arxiv.org/abs/2301.10972.

[8] Betty H.C. Cheng et al. "08031 – Software Engineering for Self-Adaptive Systems: A Research Road Map." In: *Software Engineering for Self-Adaptive Systems*. Ed. by Betty H. C. Cheng, Rogerio de Lemos, Holger Giese, Paola Inverardi, and Jeff Magee. Vol. 8031. Dagstuhl Seminar Proceedings (DagSemProc). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2008, pp. 1–13. DOI: 10.4230/DagSemProc.08031.2. URL: https://drops.dagstuhl.de/entities/document/10.4230/DagSemProc.08031.2.

[9] Norman Cliff. "Dominance statistics: Ordinal analyses to answer ordinal questions." In: *Psychological Bulletin* 114.3 (1993), pp. 494–509. DOI: 10.1037/0033-2909.114.3.494.

[10] Jacob Cohen. *Statistical Power Analysis for the Behavioral Sciences*. 2nd. Hillsdale, NJ: Lawrence Erlbaum Associates, 1988.

[11] Bernardo G. Collaco, Syed Ali Haider, Srinivasagam Prabha, Cesar A. Gomez-Cabello, Ariana Genovese, Nadia G. Wood, Sanjay P. Bagaria, Narayanan Gopala, Cui Tao, and Antonio Jorge Forte. "The role of agentic artificial intelligence in healthcare: a scoping review." In: *npj Digital Medicine* (2026). ISSN: 2398-6352. DOI: 10.1038/s41746-026-02517-5. URL: https://doi.org/10.1038/s41746-026-02517-5.

[12] Comfy Org. *Comfy Agent: Build ComfyUI Workflows With AI on Your Desktop*. Accessed: 2026-10-09. Comfy Org, 2026. URL: https://comfyui-wiki.com/en/news/2026-10-01-comfy-agent.

[13] Comfy Org. *Comfy MCP: Agent-Driven ComfyUI Workflow Integration*. Accessed: 2026-09-16. Comfy Org. 2026. URL: https://docs.comfy.org/agent-tools/mcp.

[14] Jiabao Fang, Shen Gao, Pengjie Ren, Xiuying Chen, Suzan Verberne, and Zhaochun Ren. *A Multi-Agent Conversational Recommender System*. 2024. arXiv: 2402.01135 [cs.IR]. URL: https://arxiv.org/abs/2402.01135.

[15] Matthias Feurer, Jost Tobias Springenberg, and Frank Hutter. "Initializing Bayesian Hyperparameter Optimization via Meta-Learning." In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 29. 1. 2015. DOI: 10.1609/aaai.v29i1.9354.

[16] Ronald A. Fisher. *Statistical Methods for Research Workers*. Edinburgh: Oliver and Boyd, 1925.

[17] Rinon Gal, Adi Haviv, Yuval Alaluf, Amit H. Bermano, Daniel Cohen-Or, and Gal Chechik. *ComfyGen: Prompt-Adaptive Workflows for Text-to-Image Generation*. 2024. arXiv: 2410.01731 [cs.CV]. URL: https://arxiv.org/abs/2410.01731.

[18] Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. "GenEval: An Object-Focused Framework for Evaluating Text-to-Image Alignment." In: *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*. 2023.

[19] John C. Gower. "A General Coefficient of Similarity and Some of Its Properties." In: *Biometrics* 27.4 (1971), pp. 857–871. DOI: 10.2307/2528823.

[20] Xin He, Kaiyong Zhao, and Xiaowen Chu. "AutoML: A survey of the state-of-the-art." In: *Knowledge-Based Systems* 212 (2021), p. 106622. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2020.106622. URL: https://www.sciencedirect.com/science/article/pii/S0950705120307516.

[21] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. "CLIPScore: A Reference-free Evaluation Metric for Image Captioning." In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Ed. by Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 7514–7528. DOI: 10.18653/v1/2021.emnlp-main.595. URL: https://aclanthology.org/2021.emnlp-main.595/.

[22] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. *GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium*. 2017. arXiv: 1706.08500 [cs.LG]. URL: https://arxiv.org/abs/1706.08500.

[23] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising Diffusion Probabilistic Models." In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 6840–6851. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

[24] Jonathan Ho and Tim Salimans. *Classifier-Free Diffusion Guidance*. 2022. arXiv: 2207.12598 [cs.LG]. URL: https://arxiv.org/abs/2207.12598.

[25] Holger H Hoos. "Automated algorithm configuration and parameter tuning." In: *Autonomous search*. Springer, 2012, pp. 37–71.

[26] Chengkai Huang, Junda Wu, Yu Xia, Zixu Yu, Ruhan Wang, Tong Yu, Ruiyi Zhang, Ryan A. Rossi, Branislav Kveton, Dongruo Zhou, Julian McAuley, and Lina Yao. *Towards Agentic Recommender Systems in the Era of Multimodal Large Language Models*. 2025. arXiv: 2503.16734 [cs.AI]. URL: https://arxiv.org/abs/2503.16734.

[27] Dong Huang, Jie M. Zhang, Michael Luck, Qingwen Bu, Yuhao Qing, and Heming Cui. *AgentCoder: Multi-Agent-based Code Generation with Iterative Testing and Optimisation*. 2024. arXiv: 2312.13010 [cs.CL]. URL: https://arxiv.org/abs/2312.13010.

[28] Oucheng Huang, Yuhang Ma, Zeng Zhao, Mingrui Wu, Jiayi Ji, Rongsheng Zhang, Zhipeng Hu, Xiaoshuai Sun, and Rongrong Ji. *ComfyGPT: A Self-Optimizing Multi-Agent System for Comprehensive ComfyUI Workflow Generation*. 2025. arXiv: 2503.17671 [cs.MA]. URL: https://arxiv.org/abs/2503.17671.

[29] Md. Ashraful Islam, Mohammed Eunus Ali, and Md Rizwan Parvez. "MapCoder: Multi-Agent Code Generation for Competitive Problem Solving." In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 4912–4944. DOI: 10.18653/v1/2024.acl-long.269. URL: https://aclanthology.org/2024.acl-long.269/.

[30] Jinxiang Lai, Zexin Lu, Jiajun He, Rongwei Quan, Wenzhe Zhao, Qinyu Yang, Qi Chen, Qin Lin, Chuyue Li, Tao Gao, Yuhao Shan, Shuai Shao, Song Guo, and Qinglin Lu. *VisionCreator: A Native Visual-Generation Agentic Model with Understanding, Thinking, Planning and Creation*. 2026. arXiv: 2603.02681 [cs.CV]. URL: https://arxiv.org/abs/2603.02681.

[31] Tian Lan, Xian-Ling Mao, Wei Wei, Xiaoyan Gao, and Heyan Huang. "PONE: A Novel Automatic Evaluation Metric for Open-domain Generative Dialogue Systems." In: *ACM Trans. Inf. Syst.* 39.1 (Nov. 2020). ISSN: 1046-8188. DOI: 10.1145/3423168. URL: https://doi.org/10.1145/3423168.

[32] Tian Lan, Yang-Hao Zhou, Zi-Ao Ma, Fanshu Sun, Rui-Qing Sun, Junyu Luo, Rong-Cheng Tu, Heyan Huang, Chen Xu, Zhijing Wu, and Xian-Ling Mao. *A Survey of Automatic Evaluation Methods on Text, Visual and Speech Generations*. 2025. arXiv: 2506.10019 [cs.CL]. URL: https://arxiv.org/abs/2506.10019.

[33] Zhendong Li, Lei Sun, Ruibo Ming, He Zhang, Danda Pani Paudel, Luc Van Gool, and Jinjin Gu. *Knowledge-Centric Agents for Workflow Generation in ComfyUI*. 2026. arXiv: 2607.15845 [cs.CV]. URL: https://arxiv.org/abs/2607.15845.

[34] Qian Liang, Xiaomin Li, Ying Zhang, Jia Xu, Lihao Ni, Hongrui Li, Jingjing Li, Jing Lyu, and Chen Li. *WeGenBench: A Multidimensional Diagnostic Benchmark towards Text-to-Image Model Optimization*. 2026. arXiv: 2606.20100 [cs.CV]. URL: https://arxiv.org/abs/2606.20100.

[35] Yen-Ting Lin and Yun-Nung Chen. "LLM-Eval: Unified Multi-Dimensional Automatic Evaluation for Open-Domain Conversations with Large Language Models." In: *Proceedings of the 5th Workshop on NLP for Conversational AI (NLP4ConvAI 2023)*. Ed. by Yun-Nung Chen and Abhinav Rastogi. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 47–58. DOI: 10.18653/v1/2023.nlp4convai-1.5. URL: https://aclanthology.org/2023.nlp4convai-1.5/.

[36] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan LI, and Jun Zhu. "DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps." In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh. Vol. 35. Curran Associates, Inc., 2022, pp. 5775–5787. URL: https://proceedings.neurips.cc/paper_files/paper/2022/file/260a14acce2a89dad36adc8eefe7c59e-Paper-Conference.pdf.

[37] Anish Mittal, Anush Krishna Moorthy, and Alan Conrad Bovik. "No-Reference Image Quality Assessment in the Spatial Domain." In: *IEEE Transactions on Image Processing* 21.12 (2012), pp. 4695–4708. DOI: 10.1109/TIP.2012.2214050.

[38] Pinelopi Papalampidi, Olivia Wiles, Ira Ktena, Aleksandar Shtedritski, Emanuele Bugliarello, Ivana Kajic, Isabela Albuquerque, and Aida Nematzadeh. *Dynamic Classifier-Free Diffusion Guidance via Online Feedback*. 2025. arXiv: 2509.16131 [cs.LG]. URL: https://arxiv.org/abs/2509.16131.

[39] Qiyao Peng, Hongtao Liu, Hua Huang, Qing Yang, and Minglai Shao. "A survey on llm-powered agents for recommender systems." In: *arXiv preprint arXiv:2502.10050* (2025).

[40] Ivens Da Silva Portugal, Paulo Alencar, and Donald Cowan. "An Agentic AI-based Multi-Agent Framework for Recommender Systems." In: *2024 IEEE International Conference on Big Data (BigData)*. 2024, pp. 5375–5382. DOI: 10.1109/BigData62323.2024.10825765.

[41] Dmytro Pukhkaiev and Uwe Aßmann. *Parameter Tuning for Self-optimizing Software at Scale*. 2019. arXiv: 1909.03814 [cs.LG]. URL: https://arxiv.org/abs/1909.03814.

[42] Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. "ChatDev: Communicative Agents for Software Development." In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 15174–15186. DOI: 10.18653/v1/2024.acl-long.810. URL: https://aclanthology.org/2024.acl-long.810/.

[43] Jianing Qiu, Kyle Lam, Guohao Li, Amish Acharya, Tien Yin Wong, Ara Darzi, Wu Yuan, and Eric J. Topol. "LLM-based agentic systems in medicine and healthcare." In: *Nature Machine Intelligence* 6.12 (2024), pp. 1418–1420. ISSN: 2522-5839. DOI: 10.1038/s42256-024-00944-1. URL: https://doi.org/10.1038/s42256-024-00944-1.

[44] Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-rong Wen. "Tool learning with large language models: a survey." In: *Frontiers of Computer Science* 19.8 (Jan. 2025). ISSN: 2095-2236. DOI: 10.1007/s11704-024-40678-2. URL: http://dx.doi.org/10.1007/s11704-024-40678-2.

[45] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. "Learning Transferable Visual Models From Natural Language Supervision." In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 8748–8763. URL: https://proceedings.mlr.press/v139/radford21a.html.

[46] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. "High-Resolution Image Synthesis With Latent Diffusion Models." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 10684–10695.

[47] Amirmojtaba Sabour, Sanja Fidler, and Karsten Kreis. *Align Your Steps: Optimizing Sampling Schedules in Diffusion Models*. 2024. arXiv: 2404.14507 [cs.CV]. URL: https://arxiv.org/abs/2404.14507.

[48] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. *Improved Techniques for Training GANs*. 2016. arXiv: 1606.03498 [cs.LG]. URL: https://arxiv.org/abs/1606.03498.

[49] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. *Toolformer: Language Models Can Teach Themselves to Use Tools*. 2023. arXiv: 2302.04761 [cs.CL]. URL: https://arxiv.org/abs/2302.04761.

[50] Dominik Sobania, Martin Briesch, and Franz Rothlauf. *ComfyGI: Automatic Improvement of Image Generation Workflows*. 2024. arXiv: 2411.14193 [cs.CV]. URL: https://arxiv.org/abs/2411.14193.

[51] Charles Spearman. "The Proof and Measurement of Association between Two Things." In: *American Journal of Psychology* 15.1 (1904), pp. 72–101. DOI: 10.2307/1412159.

[52] Zhenchen Tang, Zichuan Wang, Bo Peng, and Jing Dong. "CLIP-AGIQA: Boosting the Performance of AI-Generated Image Quality Assessment with CLIP." In: *Pattern Recognition*. Ed. by Apostolos Antonacopoulos, Subhasis Chaudhuri, Rama Chellappa, Cheng-Lin Liu, Saumik Bhattacharya, and Umapada Pal. Cham: Springer Nature Switzerland, 2025, pp. 48–61. ISBN: 978-3-031-78125-4.

[53] Rong-Cheng Tu, Zi-Ao Ma, Tian Lan, Yuehao Zhao, Heyan Huang, and Xian-Ling Mao. "Automatic Evaluation for Text-to-image Generation: Task-decomposed Framework, Distilled Training, and Meta-evaluation Benchmark." In: *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar. Vienna, Austria: Association for Computational Linguistics, July 2025, pp. 22340–22361. ISBN: 979-8-89176-251-0. DOI: 10.18653/v1/2025.acl-long.1088. URL: https://aclanthology.org/2025.acl-long.1088/.

[54] Shubham Vatsal, Harsh Dubey, and Aditi Singh. "Agentic AI in Healthcare and Medicine: A Seven-Dimensional Taxonomy for Empirical Evaluation of LLM-Based Agents." In: *IEEE Access* 14 (2026), pp. 4840–4863. DOI: 10.1109/ACCESS.2026.3651218.

[55] Jianyi Wang, Kelvin C.K. Chan, and Chen Change Loy. "Exploring CLIP for Assessing the Look and Feel of Images." In: *Proceedings of the AAAI Conference on Artificial Intelligence* 37.2 (2023), pp. 2555–2563. DOI: 10.1609/aaai.v37i2.25353. URL: https://ojs.aaai.org/index.php/AAAI/article/view/25353.

[56] Shuhei Watanabe. *Tree-Structured Parzen Estimator: Understanding Its Algorithm Components and Their Roles for Better Empirical Performance*. 2026. arXiv: 2304.11127 [cs.LG]. URL: https://arxiv.org/abs/2304.11127.

[57] Frank Wilcoxon. "Individual Comparisons by Ranking Methods." In: *Biometrics Bulletin* 1.6 (1945), pp. 80–83. DOI: 10.2307/3001968.

[58] Chenfei Wu et al. *Qwen-Image Technical Report*. 2025. arXiv: 2508.02324 [cs.CV]. URL: https://arxiv.org/abs/2508.02324.

[59] Xiaoshi Wu, Yiming Hao, Keqiang Sun, Yixiong Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. *Human Preference Score v2: A Solid Benchmark for Evaluating Human Preferences of Text-to-Image Synthesis*. 2023. arXiv: 2306.09341 [cs.CV]. URL: https://arxiv.org/abs/2306.09341.

[60] Dawei Xiang, Wenyan Xu, Kexin Chu, Tianqi Ding, Zixu Shen, Yiming Zeng, Jianchang Su, and Wei Zhang. "PromptSculptor: Multi-Agent Based Text-to-Image Prompt Optimization." In: *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Ed. by Ivan Habernal, Peter Schulam, and Jörg Tiedemann. Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 774–786. ISBN: 979-8-89176-334-0. DOI: 10.18653/v1/2025.emnlp-demos.59. URL: https://aclanthology.org/2025.emnlp-demos.59/.

[61] Zhenran Xu, Yiyu Wang, Xue Yang, Longyue Wang, Weihua Luo, Kaifu Zhang, Baotian Hu, and Min Zhang. *ComfyUI-R1: Exploring Reasoning Models for Workflow Generation*. 2025. arXiv: 2506.09790 [cs.CL]. URL: https://arxiv.org/abs/2506.09790.

[62] Zhenran Xu, Xue Yang, Yiyu Wang, Qingli Hu, Zijiao Wu, Longyue Wang, Weihua Luo, Kaifu Zhang, Baotian Hu, and Min Zhang. "ComfyUI-Copilot: An Intelligent Assistant for Automated Workflow Development." In: *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*. Ed. by Pushkar Mishra, Smaranda Muresan, and Tao Yu. Vienna, Austria: Association for Computational Linguistics, July 2025, pp. 632–643. ISBN: 979-8-89176-253-4. DOI:

10.18653/v1/2025.acl-demo.61. URL: https://aclanthology.org/2025.acl-demo.61/.

[63] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. *ReAct: Synergizing Reasoning and Acting in Language Models*. 2023. arXiv: 2210.03629 [cs.CL]. URL: https://arxiv.org/abs/2210.03629.

[64] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 586–595.

[65] Travis Zhang, Christian Belardi, Justin Lovelace, Jin Peng Zhou, Saebyeol Shin, Carla P. Gomes, and Kilian Q. Weinberger. *Optimize Your Sampling: Tuned Diffusion Sampling with Bayesian Optimization*. 2026. arXiv: 2608.18040 [cs.CV]. URL: https://arxiv.org/abs/2608.18040.

[66] Kaiwen Zhu, Jinjin Gu, Zhiyuan You, Yu Qiao, and Chao Dong. *An Intelligent Agentic System for Complex Image Restoration Problems*. 2025. arXiv: 2410.17809 [cs.CV]. URL: https://arxiv.org/abs/2410.17809.