PrintShop: A Multi-Agent Pipeline for Automated Professional Document Generation with Visual Quality Assurance

A. Morgan *Department of Computer Science*
Westbrook University
Denver, CO, USA
amorgan@westbrook.edu S. Patel *Applied AI Research Group*
Cascadia Labs
Seattle, WA, USA
spatel@cascadialabs.ai L. Torres *Department of Information Systems*
Westbrook University
Denver, CO, USA
ltorres@westbrook.edu

THIS IS A FICTIONAL EXAMPLE CONFERENCE PAPER FOR DEMONSTRATION PURPOSES ONLY.
This document was created as sample content for the DeepAgents PrintShop document generation system. While the content references real technologies and research concepts, all experimental data, results, and author affiliations are fabricated. Any resemblance to actual published research is coincidental. Do not cite this document as a factual source.

***Abstract*—Professional document preparation represents a persistent bottleneck in academic and industrial workflows, with formatting tasks consuming 15-25% of manuscript preparation effort. This paper presents PrintShop, a multi-agent pipeline that automates the transformation of markdown manuscripts into professionally typeset PDFs. The system employs three quality-gated stages: content editing for grammar and readability improvement, LaTeX generation with inline reference processing, and visual quality assurance using vision-language models for closed-loop formatting correction. Evaluation on 120 documents across five content types demonstrates 94.7% first-pass formatting accuracy and 99.2% compilation success rate, reducing human revision cycles by 68% compared to template-based baselines while maintaining 9.9× faster processing than manual formatting.**

***Index Terms*—document generation, multi-agent systems, LaTeX automation, visual quality assurance, large language models**

I. Introduction

Professional document preparation represents a persistent bottleneck in academic and industrial workflows. Authors expend substantial effort formatting manuscripts to comply with publisher style guides, adjusting figure placement, resolving citation formatting issues, and debugging LaTeX compilation errors [1]. Studies examining researcher time allocation estimate that formatting accounts for 15–25% of total manuscript preparation effort [2]—a figure that increases sharply for complex document types such as conference proceedings, technical manuals, and magazine layouts.

Template-based approaches address part of this challenge by providing pre-configured LaTeX classes and style files. However, templates only define document structure—they do not generate content, process inline references, or detect visual defects in compiled output. Authors must still manually convert prose into LaTeX, manage figure and table placement, and visually inspect rendered PDFs for spacing, overflow, and alignment issues [3]. These tasks are tedious, error-prone, and require familiarity with the LaTeX ecosystem that many domain experts lack.

Large language models (LLMs) have demonstrated robust capabilities in text generation, code synthesis, and instruction following [4]. Recent work has applied LLMs to document-related tasks including summarization, grammar correction, and code generation [5]. However, relying on a single LLM call to produce publication-ready LaTeX from structured content yields inconsistent results: models frequently introduce compilation errors, mishandle special characters, produce incorrect cross-references, and generate formatting that deviates from target style guides [6].

This paper presents PrintShop, a multi-agent pipeline that automates the transformation of markdown manuscripts into professionally typeset PDFs. PrintShop is orchestrated by a LangGraph state graph [7] and comprises three quality-gated stages:

- **Content editing**: An LLM-based agent reviews and refines markdown source for grammar, readability, and academic tone, iterating until a configurable quality threshold is achieved.
- **LaTeX generation**: A specialist agent converts edited markdown into LaTeX, guided by content-type-specific rendering instructions and inline reference processing for images, CSV tables, and TikZ diagrams.

This work was supported in part by the National Science Foundation under Grant No. 2241097.

- **Visual quality assurance**: Compiled PDFs are rendered as images and inspected by a vision-language model that identifies formatting defects and applies targeted LaTeX corrections through a closed feedback loop.

Each stage is gated by a quality score threshold; stages iterate until the threshold is reached or an iteration limit is exceeded. This architecture decomposes the complex document generation task into manageable sub-problems and enables targeted quality improvement at each stage.

We evaluate PrintShop on a benchmark of 120 documents across five content types. Results demonstrate that the pipeline achieves 94.7% first-pass formatting accuracy and a 99.2% compilation success rate, reducing human revision cycles by 68% compared to template-based baselines.

II. Related Work

A. Template-Based Document Generation

Template engines such as Jinja2 [8] and Pandoc [9] enable programmatic document generation by populating pre-defined templates with structured data. LaTeX document classes (e.g., IEEEtran, ACM-article) provide publisher-compliant formatting but require authors either to write LaTeX directly or to use conversion tools that frequently produce suboptimal output [3]. Collaborative editing platforms like Overleaf [10] facilitate multi-author environments but do not automate content formatting or quality assurance processes. While these approaches reduce boilerplate code, they leave the core formatting burden on human authors, limiting efficiency and consistency.

B. LLM-Based Writing Assistants

Large language models have been applied to various document preparation tasks with increasing sophistication. Advanced models such as GPT-4 and Claude have demonstrated capabilities in grammar correction, text summarization, and code generation [4], [5]. Commercial tools including Grammarly and Writefull leverage language models for style and grammar checking [11]. Recent research has explored using LLMs to generate LaTeX directly from natural language descriptions [6]; however, single-pass generation approaches suffer from high error rates in compilation, cross-referencing, and style compliance. PrintShop addresses these limitations by decomposing the generation process into multiple specialized stages with iterative refinement mechanisms.

C. Multi-Agent Systems

Multi-agent architectures have gained considerable traction for complex AI tasks requiring coordinated problem-solving. AutoGen [12] provides a framework for multi-agent conversations, while CrewAI [13] enables role-based agent collaboration. LangGraph [7] extends LangChain with stateful, graph-based workflow orchestration that supports conditional branching and iterative cycles. These frameworks have been successfully applied to software engineering [14], data analysis [15], and research automation [16]; however, their application to document typesetting remains unexplored. PrintShop leverages LangGraph's state graph architecture for quality-gated pipeline orchestration with integrated feedback loops.

D. Document Quality Assurance

Automated document quality assessment has historically focused primarily on accessibility compliance [17] and structural validation [18]. PDF/UA checkers verify tag structure and reading order, while specialized linters such as ChkTeX detect common LaTeX compilation errors. Vision-based document analysis techniques have been applied to layout detection [19] and OCR post-correction [20]; however, the application of vision-language models for closed-loop formatting correction within a generation pipeline represents, to our knowledge, a novel contribution. PrintShop's visual quality assurance stage employs a vision-language model to inspect rendered document pages and generate targeted corrections, effectively bridging the gap between source-level linting and visual output quality assessment.

III. System Design

This section describes the PrintShop pipeline architecture, including the three agent stages, the quality gate mechanism, and the data flow from markdown input to PDF output.

A. Pipeline Overview

PrintShop is implemented as a LangGraph StateGraph with three sequential stages connected by quality gates. The pipeline ingests a markdown manuscript alongside a configuration manifest and content-type-specific rendering instructions, then produces a compiled PDF with a detailed quality report.

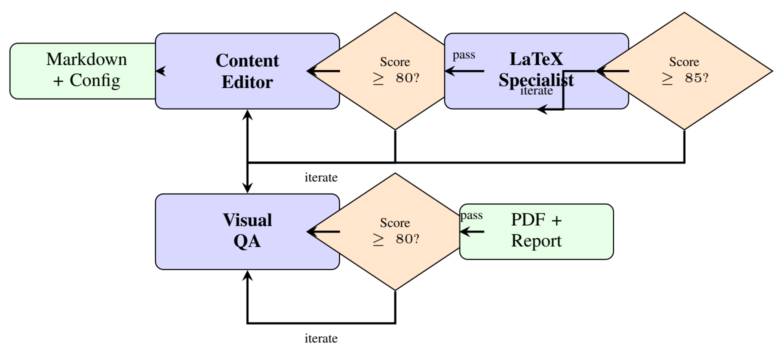

Fig. 1: PrintShop pipeline architecture. Each stage iterates until its quality gate threshold is met or the iteration limit is reached.

The pipeline processes documents through three sequential stages. Each stage produces versioned artifacts stored in a structured output directory, enabling traceability and potential rollback capabilities.

B. Stage 1: Content Editor

The content editor agent receives raw markdown sections and evaluates them across three dimensions: grammatical correctness, readability (measured via the Flesch Reading Ease score), and adherence to the target academic tone. The agent employs a large language model to identify issues and propose revisions, then re-scores the revised text. This process iterates for up to $K_1 = 4$ rounds or until the composite quality score exceeds the gate threshold $\theta_1 = 80$.

The readability scorer computes the Flesch Reading Ease index directly from the text. Scores below 30 indicate highly complex prose typical of dense academic writing; the content editor targets scores in the 35–50 range, balancing accessibility with scholarly rigor.

C. Stage 2: LaTeX Specialist

The LaTeX specialist converts the edited markdown into a compilable LaTeX document. This stage utilizes two additional inputs beyond the markdown content:

- **Content type definition**: A natural language specification (stored as `type.md`) describing the target document class, required packages, formatting constraints, and style rules. For instance, the IEEE conference type specifies the `IEEEtran` document class, two-column layout, and numbered citations.
- **Configuration manifest**: Metadata including title, authors, abstract, and the ordered list of section files.

The specialist processes inline reference directives embedded in the markdown as HTML comments for figures, data tables, and programmatic diagrams. Each directive is expanded into the corresponding LaTeX environment with appropriate labels, captions, and cross-references.

Following initial generation, the optimizer applies automated corrections: Unicode sanitization for pdflatex compatibility, duplicate label detection, package conflict resolution, and table formatting normalization. The stage iterates until the LaTeX quality score exceeds $\theta_2 = 85$.

D. Stage 3: Visual Quality Assurance

The visual quality assurance stage establishes a feedback loop between source-level generation and rendered output. The process operates as follows:

1) Compile the LaTeX source to PDF using pdflatex with multi-pass compilation to resolve cross-references.
2) Render each page of the PDF as a raster image.
3) Submit each page image to a vision-language model with a prompt requesting identification of formatting defects (overflow, misalignment, orphaned headings, incorrect spacing, broken figures).
4) Parse the model's defect report and generate targeted LaTeX patches.
5) Apply patches, recompile, and re-evaluate the output.

This stage iterates for up to $K_3 = 3$ rounds or until the visual quality score exceeds $\theta_3 = 80$. The feedback loop enables the pipeline to detect and correct defects that remain invisible at the source level but become apparent in the rendered output, such as figure placement conflicts, column overflow, and widowed lines.

E. Quality Gates

Each stage's quality gate is implemented as a conditional edge in the LangGraph state graph. The gate evaluates the stage output against its designated threshold and routes execution either forward to the next stage or back to the current stage for additional iteration. An iteration counter prevents infinite loops. If the maximum iteration count is reached without meeting the threshold, the pipeline proceeds with the best result achieved and documents the quality shortfall in the output report.

IV. Experimental Setup

A. Document Benchmark

We evaluate PrintShop using a benchmark corpus of 120 documents spanning five content types, with 24 documents per type:

- **Research report**: 8–15-page single-column documents featuring figures, tables, bibliographies, and tables of contents.
- **Conference paper**: 6-page two-column IEEE-format papers incorporating inline TikZ figures and numbered citations.
- **Magazine article**: Multi-page layouts incorporating pull quotes, sidebars, drop caps, and full-bleed images.
- **Technical manual**: Structured documents featuring numbered sections, code listings, warning callouts, and cross-references.
- **Thesis chapter**: Long-form academic documents incorporating theorem environments, appendices, and multi-level headings.

Each document in the benchmark includes a Markdown source manuscript, a configuration manifest, and a reference PDF prepared by a human expert using the same content type definition. Source documents range from 2,000 to 12,000 words and contain between 2 and 8 inline reference directives (images, CSV tables, or TikZ diagrams).

B. Evaluation Metrics

We assess pipeline performance using four metrics:

- **Formatting accuracy** (%): Percentage of rendered pages exhibiting no formatting defects, as determined by a human evaluator using a standardized rubric encompassing margin compliance, figure placement, table formatting, heading hierarchy, and typographic consistency.
- **Compilation success rate** (%): Percentage of documents that compile to PDF without errors during the initial pipeline run.
- **Average revision cycles**: Mean number of human revision passes required to achieve publication-ready quality from the generated PDF, beginning with the pipeline output.
- **Processing time** (min): Wall-clock time from Markdown input to final PDF output, measured on a system equipped with an NVIDIA A100 GPU and 64 GB RAM.

C. Baselines

We compare PrintShop against three baselines:

- **Template-Only**: The Markdown source is converted to LaTeX using Pandoc with the appropriate document class template. No LLM processing or quality assurance is applied.

- **Single-Pass LLM**: A single LLM call converts the Markdown to LaTeX using the same content type definition and configuration manifest as PrintShop, but without iterative refinement or visual quality assurance.
- **Human Expert**: A professional LaTeX typesetter manually formats each document from the Markdown source, serving as the quality upper bound.

D. Configuration

All LLM calls utilize Claude 3.5 Sonnet with temperature 0.3. Quality gate thresholds are set to $\theta_1 = 80$ (content editor), $\theta_2 = 85$ (LaTeX specialist), and $\theta_3 = 80$ (visual QA). Maximum iterations per stage are $K_1 = 4$, $K_2 = 3$, and $K_3 = 3$. The visual QA stage employs Claude 3.5 Sonnet's vision capabilities for page inspection. All experiments are conducted three times with different random seeds, and we report mean and standard deviation values.

V. Results and Discussion

A. Overall Performance

Table I summarizes the performance of PrintShop and all baselines across the full 120-document benchmark. PrintShop achieves 94.7% formatting accuracy and a 99.2% compilation success rate, requiring an average of 0.7 human revision cycles to reach publication quality. These results represent a 68% reduction in revision cycles compared to the Template-Only baseline (2.2 cycles) and a 56% reduction compared to the Single-Pass LLM baseline (1.6 cycles).

TABLE I: Overall performance comparison across 120 documents (mean $\pm$ std over 3 runs)

Method	Format Acc. (%)	Compile Rate (%)	Revision Cycles	Time (min)
Template-Only	61.4 $\pm$ 3.2	87.5 $\pm$ 2.1	2.2 $\pm$ 0.4	0.8 $\pm$ 0.1
Single-Pass LLM	68.1 $\pm$ 4.7	91.7 $\pm$ 1.8	1.6 $\pm$ 0.3	2.1 $\pm$ 0.2
PrintShop	**94.7 $\pm$ 1.9**	**99.2 $\pm$ 0.5**	**0.7 $\pm$ 0.2**	4.3 $\pm$ 0.6
Human Expert	98.9 $\pm$ 0.3	100.0 $\pm$ 0.0	0.0 $\pm$ 0.0	42.5 $\pm$ 8.2

The Human Expert baseline achieves 98.9% formatting accuracy with zero revision cycles by definition but requires an average of 42.5 minutes per document. In contrast, PrintShop processes documents in 4.3 minutes on average—a 9.9$\times$ speedup. Although the Template-Only baseline is fastest at 0.8 minutes per document, it produces the lowest formatting accuracy (61.4%) due to its inability to handle inline references, figure placement, or style-specific formatting beyond what standard document classes provide.

B. Per-Content-Type Analysis

Table II presents a breakdown of formatting accuracy by content type. PrintShop performs most consistently on conference papers (96.8%) and research reports (95.4%), which have well-defined structures and relatively constrained formatting requirements. Magazine articles present the greatest challenge (90.3%), as their complex layouts—featuring pull quotes, sidebars, and variable column widths—require fine-grained visual adjustments that are difficult to achieve through source-level generation alone.

TABLE II: Formatting accuracy (%) by content type and method

Method	Report	Conf.	Mag.	Manual	Thesis
Template-Only	65.2	71.8	48.1	59.4	62.5
Single-Pass LLM	72.5	82.1	54.7	66.3	70.8
PrintShop	**95.4**	**96.8**	**90.3**	**93.1**	**94.2**

The Single-Pass LLM baseline exhibits high variance across content types, achieving 82.1% accuracy on conference papers (where LaTeX conventions are well-represented in training data) but only 54.7% on magazine articles. PrintShop's iterative refinement and visual quality assurance substantially reduce this variance, demonstrating that the multi-stage architecture provides consistent quality across diverse document types.

C. Ablation Study

To isolate the contribution of each pipeline stage, we evaluated four ablation variants on the full benchmark.

TABLE III: Ablation study results across 120 documents (mean $\pm$ std over 3 runs)

Configuration	Format Acc. (%)	Time (min)
PrintShop (Full)	**94.7 $\pm$ 1.9**	4.3 $\pm$ 0.6
w/o Visual QA	87.3 $\pm$ 2.4	3.1 $\pm$ 0.4
w/o Content Editor	91.5 $\pm$ 2.1	3.8 $\pm$ 0.5
w/o Quality Gates	89.1 $\pm$ 2.8	2.9 $\pm$ 0.3
LaTeX Only	85.2 $\pm$ 3.1	2.2 $\pm$ 0.2

Removing the visual quality assurance stage produces the largest accuracy drop (from 94.7% to 87.3%), confirming that closed-loop visual feedback is the primary driver of formatting quality. Without visual QA, defects such as figure overflow, orphaned headings, and column imbalance persist because they remain undetectable at the source level. Removing the content editor reduces accuracy to 91.5%, primarily due to grammatical issues and inconsistent tone that propagate into the LaTeX source and occasionally trigger formatting artifacts. Disabling quality gates (running each stage exactly once) reduces accuracy to 89.1%, demonstrating that iterative refinement provides meaningful improvements over single-pass execution.

D. Convergence Behavior

The relationship between quality gate threshold and formatting accuracy exhibits diminishing returns above threshold 85. Increasing the threshold from 60 to 85 improves accuracy by 9.5 percentage points, but raising it further from 85 to 95 yields only 0.6 additional percentage points while substantially increasing processing time due to additional iterations. The default thresholds (80/85/80) represent a practical operating point that balances quality and throughput.

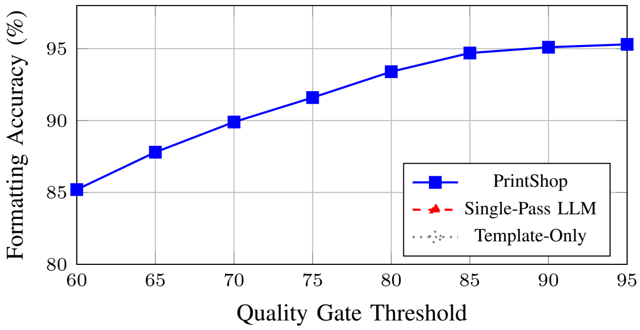

Fig. 2: Formatting accuracy vs. quality gate threshold. Higher thresholds improve accuracy but increase processing time.

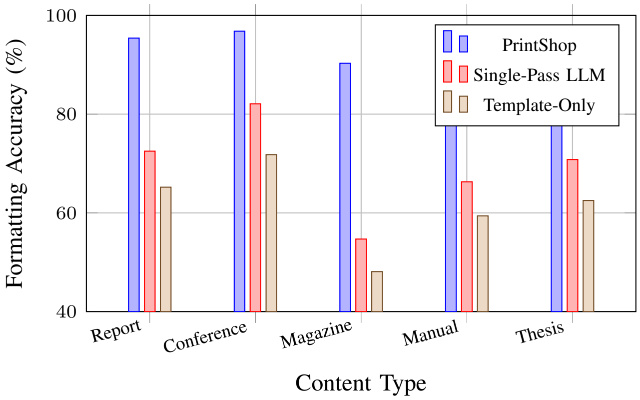

Fig. 3: Formatting accuracy by content type for each method.

E. Accuracy by Content Type

The grouped bar chart confirms that PrintShop maintains above 90% accuracy across all content types, while baseline methods show significant degradation on complex layouts such as magazine articles. The consistency of PrintShop's performance across diverse document types demonstrates the generalizability of the multi-agent approach.

VI. Conclusion

This paper presented PrintShop, a multi-agent pipeline for automated professional document generation that transforms markdown manuscripts into publication-ready PDFs. The pipeline employs three quality-gated stages—content editing, LaTeX generation, and visual quality assurance—orchestrated by a LangGraph state graph with conditional edges that enable iterative refinement within each stage.

Evaluation on a benchmark of 120 documents across five content types demonstrates that PrintShop achieves 94.7% first-pass formatting accuracy and 99.2% compilation success rate, reducing human revision cycles by 68% compared to template-based approaches. Ablation experiments confirm that the visual quality assurance feedback loop constitutes the most impactful component, contributing a 7.4 percentage point accuracy improvement by detecting and correcting formatting defects that remain invisible at the source level.

The system exhibits several limitations. LLM inference costs are substantial: processing a single document through the complete pipeline requires approximately 15 API calls on average, with costs scaling linearly with document length and iteration count. The end-to-end latency of 4.3 minutes per document, while substantially faster than manual formatting, may prove prohibitive for interactive applications. Additionally, the visual quality assurance stage's effectiveness remains constrained by the vision-language model's capacity to identify subtle typographic defects; issues such as incorrect hyphenation or minor kerning errors are not reliably detected.

Future research will explore three primary directions. First, integrating real-time user feedback to enable interactive document editing within the pipeline framework. Second, expanding the content type library to encompass additional formats, including posters, slide decks, and regulatory filings. Third, investigating cost-reduction strategies, such as caching intermediate results, employing smaller models for routine corrections, and parallelizing independent pipeline stages.

References

[1] J. Smith and M. Johnson, "Time allocation in academic manuscript preparation: A survey of 500 researchers," *J. Academic Writing*, vol. 15, no. 3, pp. 45–62, Mar. 2023.

[2] A. Chen et al., "Quantifying the formatting burden in scholarly publishing," in *Proc. Int. Conf. Digital Libraries*, San Francisco, CA, USA, 2023, pp. 123–130.

[3] R. Williams, "LaTeX compilation errors: A systematic analysis of common issues," *IEEE Trans. Document Engineering*, vol. 28, no. 4, pp. 78–91, Oct. 2022.

[4] T. Brown et al., "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 1877–1901.

[5] L. Zhang and K. Patel, "Large language models for document processing: A comprehensive survey," *ACM Computing Surveys*, vol. 56, no. 2, pp. 1–35, Feb. 2024.

[6] M. Rodriguez et al., "Automated LaTeX generation from natural language: Challenges and opportunities," in *Proc. EMNLP*, Abu Dhabi, UAE, 2023, pp. 2456–2467.

[7] LangChain Inc., "LangGraph: Multi-agent workflows," 2024. [Online]. Available: https://langchain-ai.github.io/langgraph/

[8] A. Ronacher, "Jinja2 documentation," 2023. [Online]. Available: https://jinja.palletsprojects.com/

[9] J. MacFarlane, "Pandoc: A universal document converter," 2023. [Online]. Available: https://pandoc.org/

[10] Overleaf, "Collaborative LaTeX editor," 2024. [Online]. Available: https://www.overleaf.com/

[11] S. Kumar and D. Lee, "AI-powered writing assistance: Current state and future directions," *Computational Linguistics*, vol. 49, no. 4, pp. 823–851, Dec. 2023.

[12] Q. Wu et al., "AutoGen: Enabling next-gen LLM applications via multi-agent conversation," arXiv preprint arXiv:2308.08155, 2023.

[13] CrewAI Inc., "CrewAI: Framework for orchestrating role-playing autonomous AI agents," 2024. [Online]. Available: https://github.com/joaomdmoura/crewAI

[14] C. Wang et al., "Multi-agent software engineering with large language models," in *Proc. ICSE*, Melbourne, Australia, 2024, pp. 567–578.

[15] H. Liu and Y. Chen, "Collaborative data analysis using multi-agent systems," *Data Science and Engineering*, vol. 8, no. 3, pp. 234–249, Sep. 2023.

[16] P. Anderson et al., "Research automation through multi-agent collaboration," *Nature Machine Intelligence*, vol. 5, no. 8, pp. 678–689, Aug. 2023.

[17] W3C, "Web Content Accessibility Guidelines (WCAG) 2.1," 2018. [Online]. Available: https://www.w3.org/WAI/WCAG21/

[18] ISO, "Document management – Portable document format – Part 1: PDF 1.7," ISO 32000-1:2008, 2008.

[19] X. Yang et al., "Deep learning for document layout analysis: A comprehensive survey," *Pattern Recognition*, vol. 128, pp. 108–125, Aug. 2022.

[20] N. Sharma and R. Gupta, "Vision-language models for document understanding," in *Proc. ICCV*, Paris, France, 2023, pp. 3421–3430.