

REPLICA: An Agentic Framework for Visually Faithful Document Reconstruction

Raghuveer R^{1*}[0009-0002-6321-1051], Anirudh Srinivasan^{1*}[0009-0008-1444-0222],
Venkata Kesav Venna^{1*}[0009-0000-1327-4607], Sreevatsa S²[0009-0006-7362-5759],
Aryan Jain²[0009-0004-8903-9119], Sahithi Kukkala²[0009-0003-5483-2232], and
Ravi Kiran Sarvadevabhatla^{1,2}[0000-0003-4134-1154]



¹BharatGen

{raghuveer.r, anirudh.srinivasan, venkat.kesav}@bharatgen.com



²IIIT Hyderabad

{tallapragada.s@research., aryan.j@research., sahithi.kukkala@research.,
ravi.kiran@}iiit.ac.in

* Equal Contribution

Project Page: <https://replica-agents.github.io/>

Abstract. PDF documents integrate poorly with the web. To improve accessibility, they are frequently converted into HTML. Existing approaches for document-to-HTML conversion largely focus on retaining text and logical structure. However, the document’s physical structure is not preserved. Visual cues such as color, font style are overlooked and design aspects such as content positioning are ignored. To address these shortcomings, we introduce REPLICA, an agentic document-to-HTML framework that generates visually faithful, layout-aware, and semantically enriched HTML which we refer as FID-HTML. Our approach generalizes across diverse document types and multiple languages. For systematic evaluation, we introduce VFDR-Bench spanning 15 languages, and 15+ document types. REPLICA achieves 93% visual fidelity, outperforming strong baselines including GPT-5 and Gemini 3 Pro. Notably, REPLICA with Qwen3-VL-8B achieves a 39% gain overall as compared to Qwen3-VL-30B, despite being $3.75\times$ smaller and operating at comparable inference time, highlighting the effectiveness of structured agentic design over brute-force scale. By preserving visual integrity and semantics in multilingual documents, REPLICA unlocks new possibilities for design-preserving translation, scalable access, and structured search.

Keywords: Document Reconstruction · Physical and Logical Layout Analysis · Document Agents

1 Introduction

PDFs preserve documents with precise control over layout, typography, and visual styling. Yet this fidelity often vanishes when converted for the web. Traditional OCR engines [50, 18, 9] process page images to recover plain text, discarding

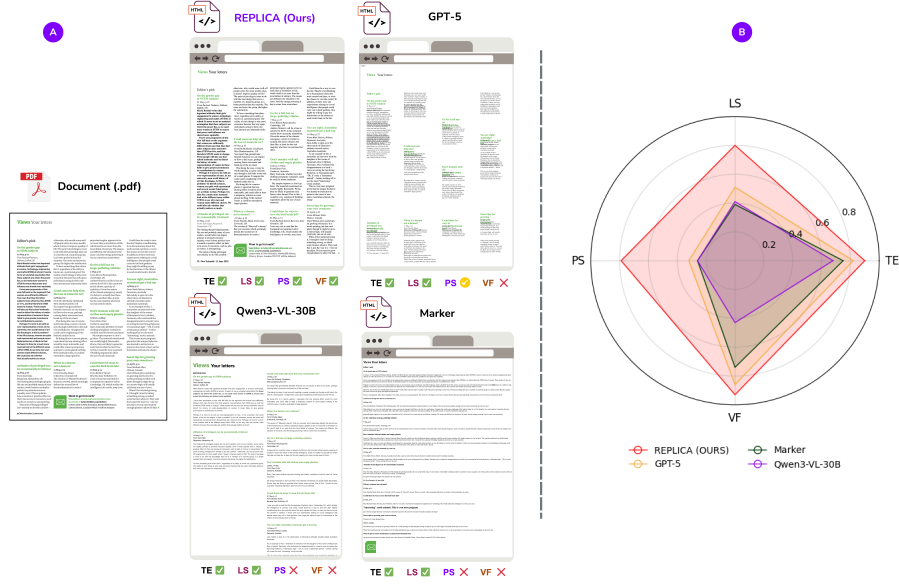


Fig. 1: **A** Comparison of visual fidelity across four approaches: our method REPLICA, an open-source Vision-Language Model (Qwen3-VL-30B [64]), a commercial Vision-Language Model (GPT-5 [38]), and a pipeline-based method (Marker [43]). Note that REPLICA faithfully preserves fine-grained layout cues - such as text color, paragraph indentation, font attributes (e.g., bold and italics), font size, and even global structural properties like the correct number of columns. **B** Radar plot of quantitative performance: showing the performance of REPLICA relative to baselines. Unlike other baselines, REPLICA reliably preserves all key document aspects after conversion - TE (Text Extraction), PS (Physical Structure), LS (Logical Structure), and VF (Visual Fidelity). Additional qualitative results can be found in fig. 2.

layout, typography, and styling. Other workflows convert PDFs to alternate formats such as Markdown [10]. PDF parsers [1,46,41] extract text or lightweight markup directly from the PDF’s internal structure but still lose structural hierarchy (e.g., multi-column layouts), fine-grained formatting, and much of the visual and spatial identity.

PDF-to-HTML conversion is a more common alternative in web publishing, digital archiving, and accessibility workflows. Plain HTML with minimal CSS [11] preserves basic layout elements but often simplifies typography, color, and positioning.

Existing HTML converters [32,43] often linearize multi-column layouts, distort tables, and misplace elements, compromising both structure and visual fidelity. Tools like LaTeXML [34] work well for specific sources (e.g., LaTeX) but lack cross-format generality. Recent document-focused VLMs [36,45,28,12] generate structured outputs from images, yet struggle with multilingual scripts, degraded scans, complex hierarchies, and fine-grained style preservation.

To overcome these limitations, we introduce REPLICA, an agentic framework that generates FID-HTML- semantically enriched, layout-aware HTML preserving text, logical structure, physical layout, and visual fidelity (see fig. 1). Beyond plain rendering, it embeds semantic layout classes and spatial cues, di-

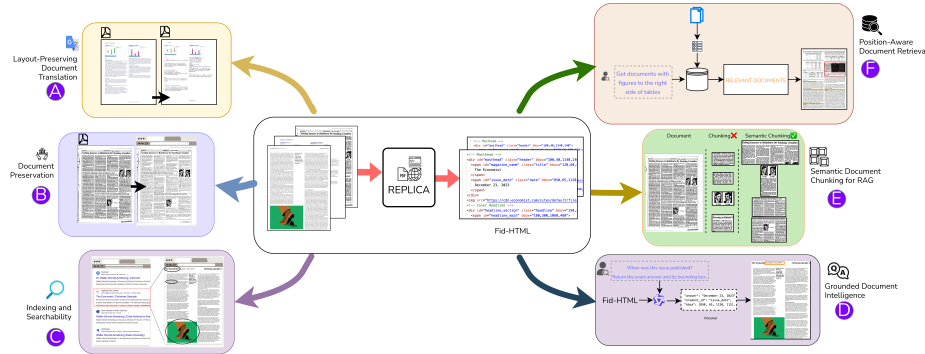


Fig. 2: Downstream Applications of REPLICA enabled by FID-HTML: **A** Layout-Preserving Document Translation, **B** Document Preservation, **C** Indexing & Searchability, **D** Grounded Document Intelligence, **E** Semantic Document Chunking for RAG, **F** Position-Aware Document Retrieval

rectly addressing the layout, typography, and styling losses of prior methods. Our contributions are:

1. FID-HTML, a new web-native format that represents documents in high-fidelity HTML, which is both semantically enriched and preserves visual and spatial fidelity.
2. REPLICA, an agentic framework for end-to-end page reconstruction that segments, localizes, assembles, and refines content to produce FID-HTML.
3. VFDR-Bench, a benchmark for evaluating fidelity and robustness in document-to-web conversion.
4. A suite of new evaluation metrics: Global Position Score (GPS), Local Position Score (LPS), and Visual Fidelity Score (VFS) - introduced alongside the benchmark for evaluating physical structure and visual fidelity in document-to-web conversion.

More broadly, REPLICA expands the utility of enriched HTML outputs beyond basic readability, potentially enabling applications like layout-preserving document translation, grounded document intelligence, Retrieval Augmented Generation (RAG), etc., (see fig. 2). The presence of semantic tags, preserved reading order, and enriched figure alt-text in FID-HTML ensures seamless compatibility with screen readers on the web-rendered document, further enhancing accessibility.

2 Related Work

Document reconstruction spans two complementary directions. Logical reconstruction recovers semantic structure - reading order, hierarchy (sections, lists, tables), and structured outputs such as HTML or LaTeX - often sacrificing precise layout. Physical reconstruction targets pixel-level fidelity, preserving page

geometry, multi-column structure, exact placement, styling, font attributes, spacing, and faithful reinsertion of figures and backgrounds. Together, these paradigms span the spectrum from semantic parsing to layout-preserving re-creation.

2.1 Logical Document Reconstruction

Pipeline Parsing: Modular systems split document understanding into layout detection, reading order, OCR, and table/math parsing, then merge outputs via rules or learned modules [72,17,59,18,23,56,42,15]. Tools like Marker [43] and MinerU [55] produce solid Markdown/JSON/HTML but typically linearize layouts and under-specify styling, limiting visual fidelity.[43,55,32]. **End-to-End VLMs:** General-purpose VLMs [66,3,8,64,68,6] support zero-shot document understanding and can generate descriptions or lightweight HTML/JSON , but often miss fine layout and styling details. Specialist models [20,4,17,60,16,27,31] improve hierarchy, tables, and formulas via document-centric training . Recent variants like Qwen3-VL and olmOCR generate HTML-like outputs [64,45], yet most still fall short on visual fidelity, including fonts, spacing, and multi-column placement.

2.2 Physical Document Reconstruction

Approaches for Document Images: Systems such as PDFMathTranslate and image-to-LaTeX regeneration target scientific documents, preserving formulas, tables, and two-column layouts [40,48,19,57]. More recent work like DREAM focuses on high-fidelity digital reconstruction, autoregressively modeling both content and layout to recover precise element positions [25].

Approaches for UI Screenshots: Screenshot-to-HTML methods target UI-centric, text-light layouts [53,61,13,30,14,70], often using iterative rendering and reflection [21,49], along with GUI detection and language-driven refinement [63,62,71]. However, they generalize poorly to complex, text-dense document layouts due to distribution shift.

Our approach (REPLICA) is a layout-aware method that explicitly targets *both* axes: (i) logical reconstruction (semantic tags, reading order, tables/math) and (ii) physical reconstruction (absolute positioning, fonts, spacing, multi-column layout). This yields web-native HTML/CSS that is semantically enriched and faithful to source appearance.

3 FID-HTML: A High-Fidelity Document Representation.

FID-HTML is a high-fidelity, semantically enriched HTML representation generated by , capturing both semantic structure and visual presentation to support faithful reconstruction and machine-readable downstream tasks.

Core Representation. FID-HTML encodes a document along six complementary dimensions to jointly preserve semantics and visual fidelity:  *Textual*

Table 1: **Comparison of document representation formats.** T.E. denotes Text Extraction; L.S. denotes Logical Structure; P.S. denotes Physical Structure; V.F. denotes Visual Fidelity. R.O. denotes Reading Order and L.H. denotes Logical Hierarchy. ‘P’ denotes partial support.

Format Type	Subtype	T.E.		L.S.				P.S.		V.F.
		Content	R.O.	L.H.	Tables	Lists	Figures	Formula	Hierarchy	BBoxes
Markdown	md	✓	P	✓	✗	✗	P	P	P	✗
	mmd	✓	P	✓	P	✗	P	P	P	✗
LaTeX	LaTeX	✓	P	✓	✓	✓	✓	✓	✓	✓
DocTags	DocTags [36]	✓	P	✓	✓	✓	✓	✓	✓	P
HTML	HTML	✓	P	✓	✓	✓	P	P	P	✗
	hOCR [50]	✓	P	✓	✓	✓	P	P	✓	✓
	Qwen-HTML [64]	✓	P	✓	✓	✓	P	P	✗	✓
	Unstructured HTML [52]	✓	P	✓	✓	✓	P	✗	✗	✓
Fid-HTML (Ours)		✓	✓	✓	✓	✓	✓	✓	✓	✓

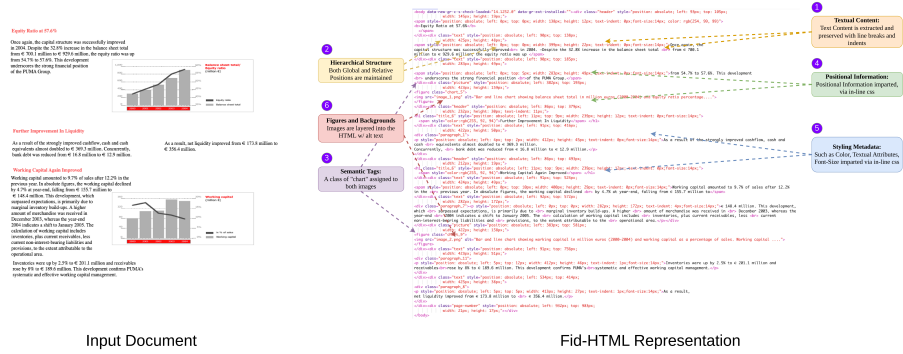


Fig. 3: **FID-HTML: High-Fidelity Document Representation.** FID-HTML is the enriched output format of REPLICA, designed to preserve both the semantic structure and visual presentation of a document. The representation encodes: ① *Textual content*; ② *Hierarchical structure*; ③ *Semantic tags*; ④ *Positional information*; ⑤ *Styling metadata*; and ⑥ *Figures and backgrounds*.

content, where complete text is retained at both word and block levels; ② *Hierarchical structure*, captured through nested `<div>` containers reflecting sections, subsections, and logical groupings; ③ *Semantic tags*, including structured elements such as `<li>`, `<table>`, `<tr>`, and `<td>`, along with meaningful class names for semantic grounding; ④ *Positional information*, encoding global placement via absolute positioning and local alignment via relative containers and `<p>` wrappers; ⑤ *Styling* and *metadata*, preserving font size, color, and text attributes (bold, italics, underline, strikethrough) through inline CSS; and ⑥ *Figures and backgrounds*, where images and page-level backgrounds are lay-

ered into the DOM with accessibility-oriented alt-text. Together, these components enable faithful reconstruction while retaining machine-readable semantic structure (see fig. 3).

Advantages: Unlike traditional parsers that discard layout and styling, FID-HTML preserves the full spectrum of document signals. Semantic tags and logical grouping support downstream LLM/VLM tasks (e.g., understanding, summarization, retrieval) (see Table 2); absolute and relative positioning maintain layout fidelity; styling metadata ensures faithful rendering; and VLM-generated alt-text enhances accessibility - capabilities typically absent in conventional systems (see Table 1).

4 Approach

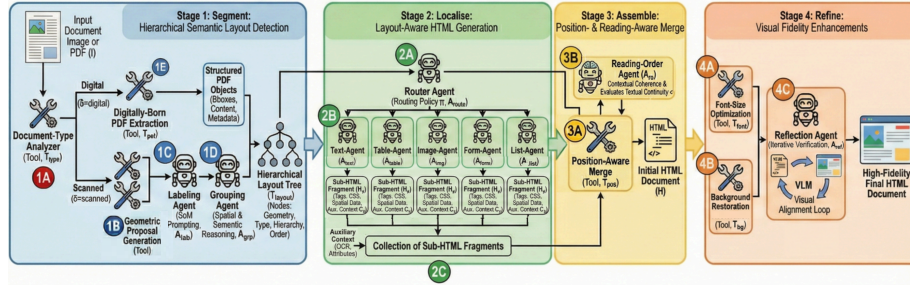


Fig. 4: **Overview of REPLICA: An agentic framework for Visually Faithful Document Reconstruction.** The agentic framework constructs a hierarchical semantic layout (1 Segment), generates sub-HTML fragments via routed specialist-agents (2 Localize), composes them with position- and reading-aware assembly (3 Assemble), and iteratively refines visual fidelity through font optimization, background restoration, and VLM-based reflection (4 Refine), producing a high-fidelity final HTML document.

REPLICA is an agentic document-to-web framework that treats documents as structured visual entities, reconstructing them in HTML with structural and visual fidelity.

REPLICA adopts an agentic divide-and-conquer strategy centered on four questions (Fig. 4):

1. *How to divide the document into meaningful components?* (**Segment**) ①
2. *How to convert each component into equivalent HTML?* (**Localize**) ②
3. *How to assemble components into a coherent structure?* (**Assemble**) ③
4. *How to refine structure for visual and structural fidelity?* (**Refine**) ④

We formalize the system as a coordination between reasoning agents and deterministic tool calls. Let $\mathcal{A}$ denote the set of agents and $\mathcal{T}$ the set of tools. Agents

perform semantic reasoning, routing, grouping, and generation, while tools execute structured and reproducible operations (e.g., layout detection, reading-order computation, font optimization).

Given a document page image or PDF I , the objective is to produce HTML H whose rendering $R(H)$ aligns with I across TE, LS, PS, and VF dimensions.

4.1 Segment: Hierarchical Semantic Layout Detection

The segment stage builds a semantically grounded layout tree $\mathcal{T}_{\text{layout}}$ encoding geometry, types, and hierarchy. While deterministic detectors localize boundaries well, they lack semantic reasoning and can fail on borderless tables, multi-column wraps, or nested structures. We therefore introduce reasoning agents to validate, refine, and hierarchically organize proposals before generation, preventing downstream error propagation.

Document-Type Analysis (Tool) 1A A document classifier tool $T_{\text{type}} \in \mathcal{T}$ determines whether the input I is digitally-born or scanned:

$$\delta = T_{\text{type}}(I), \quad \delta \in \{\text{digital}, \text{scanned}\}.$$

This decision governs whether structure is extracted directly from PDF primitives or inferred from visual geometry.

Geometric Proposal Generation (Tool) 1B If $\delta = \text{scanned}$, we invoke a layout detection ensemble $\mathcal{T}_{\text{det}} \subset \mathcal{T}$ to generate candidate regions:

$$\mathcal{B} = \{b_i\}_{i=1}^N \quad \text{where} \quad b_i = m(I), \quad m \in \mathcal{T}_{\text{det}}.$$

Each region b_i is defined by a bounding box and pixel support. These proposals are geometry-driven and may over-segment, merge incompatible content, or miss hierarchical relationships.

Labeling Agent 1C A labeling agent $A_{\text{lab}} \in \mathcal{A}$ assigns semantic attributes to each region:

$$(\ell_i, \tau_i) = A_{\text{lab}}(I, b_i),$$

where ℓ_i denotes the structural label (e.g., paragraph, table, figure) and τ_i captures auxiliary attributes (e.g., style or subtype). Using Set-of-Mark prompting [65], the agent jointly reasons over spatial grounding and textual content to also identify mis-segmented regions, detect mixed-content blocks, and reinterpret ambiguous proposals.

Grouping Agent 1D Given labeled regions, a grouping agent $A_{\text{grp}} \in \mathcal{A}$ constructs a directed layout graph

$$\mathcal{G} = (V, E), \quad V = \{v_i \mid v_i \leftrightarrow b_i\},$$

where nodes correspond to semantic regions. Edges E encode parent-child relations inferred from spatial containment, reading order, and semantic compatibility. During this stage, inconsistent or fragmented proposals can be merged,

reattached, or discarded to enforce coherent structure. The final layout tree is obtained by enforcing acyclicity and single-parent constraints:

$$\mathcal{T}_{\text{layout}} = \text{Tree}(\mathcal{G}).$$

This step resolves caption–figure bindings, list nesting, footnote attachment, and other hierarchical ambiguities beyond pure geometric detection.

Digitally-Born PDF Extraction (Tool) 1E If $\delta = \text{digital}$, we bypass geometric detection and instead invoke a structured PDF parser $T_{\text{pdf}} \in \mathcal{T}$:

$$\mathcal{B} = T_{\text{pdf}}(I),$$

which directly extracts bounding boxes, textual spans, and font metadata from PDF primitives. In this case, visual grounding and proposal aggregation are unnecessary.

4.2 Localize: Layout-Aware HTML Generation

Given the hierarchical layout tree $\mathcal{T}_{\text{layout}}$, the localize stage converts each node $v \in \mathcal{T}_{\text{layout}}$ into structured sub-HTML. Single-shot VLM generation per region is brittle and hallucination-prone, so we introduce a router agent for hypothesis validation before generation. The generation is carried out by specialist agents.

Router Agent 2A Each node v is associated with:

$$(I_v, b_v, \ell_v, \phi_v),$$

where b_v its bounding box, I_v is the visual crop of the input document restricted to b_v , ℓ_v the upstream semantic label from the Segment stage, and ϕ_v encodes structural constraints (e.g., parent type, ordering).

A routing agent $A_{\text{route}} \in \mathcal{A}$ selects an appropriate specialist agent:

$$a_v = A_{\text{route}}(I_v, \ell_v, \phi_v), \quad a_v \in \mathcal{A}_{\text{spec}}.$$

Rather than relying solely on the upstream label ℓ_v , the router jointly evaluates visual and contextual cues to detect modality mismatches. If the crop contradicts the predicted type (e.g., an equation labeled as text), it overrides the label and reroutes the node to the correct specialist, preventing cascading hallucinations during generation.

Specialist Agents 2B Different layout elements require distinct generation strategies. We therefore define a set of specialist agents:

$$\mathcal{A}_{\text{spec}} = \{A_{\text{text}}, A_{\text{table}}, A_{\text{img}}, A_{\text{form}}, A_{\text{list}}, A_{\text{eqn}}\} \subset \mathcal{A}.$$

After routing, each node v is assigned a specialist $a_v \in \mathcal{A}_{\text{spec}}$, which generates localized HTML:

$$H_v = a_v(I_v, b_v, \phi_v),$$

Conditional Tool Invocation. Specialist agents invoke auxiliary tools $\mathcal{T}_{\text{aux}} \subset \mathcal{T}$ only when necessary. For instance, the Text Agent calls OCR for low-resolution or script-complex crops but bypasses it for clean digital text, while the Table Agent queries a structure recognizer when detecting merged headers or irregular grids to obtain explicit row-column constraints. This selective orchestration improves fidelity without redundant computation, and since nodes are conditionally independent, sub-HTML generation is parallelizable.

4.3 Assemble: Position- and Reading-Aware Merge

The assemble stage merges independently generated sub-HTMLs $\{H_v\}_{v \in \mathcal{T}_{\text{layout}}}$ into a coherent document. While localization is node-level, reconstruction must preserve both hierarchical structure and logical reading flow. We therefore separate structural placement from semantic ordering.

Position-Aware Merge (Tool) 3A A composition tool $T_{\text{pos}} \in \mathcal{T}$ performs a bottom-up merge over $\mathcal{T}_{\text{layout}}$, preserving containment, nesting, and spatial alignment to produce a structurally valid document.

Reading-Order Agent 3B Spatial heuristics (top-to-bottom and left-to-right sorting) alone fails on complex layouts like multi-column newspapers (B in Figure 2) or floating elements. A reading-order agent $A_{\text{ro}} \in \mathcal{A}$ therefore computes a semantically grounded ordering relation by evaluating textual continuity and contextual coherence across nodes. The final HTML is linearized under $\prec$, ensuring a natural and logically consistent reading flow.

4.4 Refine: Visual Fidelity Enhancements

Even when segmentation, localization, and assembly are structurally correct, the rendered HTML may visually diverge from the source due to differences between PDF rendering and the browser’s CSS box model. To address this, we introduce a refinement stage that operates on the rendered output $R(H)$ and compares it against the original document image I .

Font-Size Optimization (Tool) 4A A deterministic tool $T_{\text{font}} \in \mathcal{T}$ selects maximal non-overflow font scales via bounded search to preserve typographic fidelity.

Background Restoration (Tool) 4B A tool $T_{\text{bg}} \in \mathcal{T}$ extracts page-level visual cues (e.g., background colors, shading, overlays) and restores them using CSS layering and absolute positioning to preserve global visual context.

Reflection Agent 4C A reflection agent $A_{\text{ref}} \in \mathcal{A}$ performs closed-loop visual comparison between $R(H)$ and I , identifying perceptual artifacts such as subtle overlaps, wrapping inconsistencies, or misalignment that emerge only after rendering. When discrepancies are detected, the agent injects targeted CSS corrections (e.g., layout adjustments or `shape-outside`) until visual alignment stabilizes. This refinement stage ensures that fidelity is evaluated at the level of appearance rather than structural compliance alone.

4.5 Implementation Details

Segment. For scanned documents, layout regions are detected using three deterministic tools: Doclayout-YOLO-v10 [73] for flat layouts, IndicDLP-YOLO-v10 [37] for hierarchical structures, and Hi-SAM [69] for fine-grained text segmentation. Their predictions are fused using IoU-based merging. Semantic labeling and hierarchical grouping are performed using Qwen3-VL-8B-Instruct, which assigns region types and constructs the nested layout tree.

Localize. Each layout node is routed to a specialized agent (text, table, image, form, list, or equation), all implemented using Qwen3-VL-8B [64]. Additional context is provided through deterministic tools: OCR via Gemma-3-4B-it [51], word localization via DocTR [35], and fine-grained style extraction via TextAR [22]. For tables, Surya’s Table Structure Recognition is used to explicitly recover row-column grids. Structured output validation and bounded retries ensure well-formed HTML.

Assemble. Fragments are merged using geometry and hierarchy signals from segmentation. Reading order is computed using coordinate-based heuristics (top-to-bottom, left-to-right with column handling).

Refine. Font sizes are optimized using `textFit.js` [47] with bounding box constraints from Hi-SAM. Backgrounds are reconstructed using OpenCV-based inpainting guided by region masks. Finally, a reflection agent (Qwen3-VL-8B) performs up to five iterative corrections (`max_iterations=5`) to visually align the rendered HTML with the source document.

5 Experimental Setup

We conduct comprehensive experiments to assess the performance of competing approaches on four key dimensions of document-to-web conversion. Beyond final output quality, we adopt an agentic evaluation protocol that measures intermediate agent decisions (segmentation, routing, refinement) and tool reliability (reading order, font fitting, background restoration), enabling component-level failure attribution in addition to system-level comparison.

VFDR-Bench We introduce VFDR-Bench, a benchmark for visually faithful document-to-web conversion. Existing efforts target sub-tasks such as text extraction (TE) via OCR benchmarks [67,45] or address logical structure (LS) only limitedly through table-to-HTML [24] and document-to-Markdown [29,7,39], but none capture all dimensions of the problem. VFDR-Bench fills this gap with 5,000 documents spanning single- and multi-column layouts, scanned and born-digital sources, 15+ languages, and 20+ document categories. All ground-truth annotations are fully expert human annotated to ensure structural, spatial, and visual correctness, providing a rigorous basis for both system-level fidelity evaluation and agent-level analysis.

5.1 System-Level Fidelity Metrics

Text Extraction (TE): Character Recognition Rate (CRR) and Word Recognition Rate (WRR).

Logical Structure (LS): Normalized Tree Edit Distance (NTED). We use TEDS [44] for tables and CDM [54] for formulas.

Physical Structure (PS): Global Position Score (GPS) and Local Position Score (LPS) measure layout- and line-level spatial fidelity. For layout blocks ($\mathcal{B}$) and lines ($\mathcal{L}$), we perform one-to-one matching between ground-truth and predicted elements using maximum IoU. We compute area-normalized spatial precision and recall based on overlap ratios, and report their harmonic mean (F1) as the final score. This formulation penalizes both missed elements (recall) and hallucinated bounding boxes (precision), yielding a balanced measure of physical structure fidelity.

Visual Fidelity (VF): Visual Fidelity Score (VFS) is computed via a VLM-as-a-judge comparing rendered HTML to human-annotated ground truth.

Overall Score: Overall score is calculated as an average across all the axes (TE, LS, PS, VF). To prevent pathological OCR failures from dominating multilingual aggregates, CRR and WRR are clipped at -1 per sample before averaging.

$$\text{Overall} = \frac{\text{TE} + \text{LS} + \text{PS} + \text{VF}}{4}, \quad \text{TE} = \frac{\text{CRR} + \text{WRR}}{2}, \quad \text{PS} = \frac{\text{GPS} + \text{LPS}}{2}.$$

5.2 Agent-Level Evaluation

For the *Segment* stage, we introduce Agnostic Text F1 (AT-F1) and Agnostic Visual F1 (AV-F1). Since downstream Set-of-Mark (SoM) prompting resolves semantic labels, evaluation isolates geometric quality only. We compute area-normalized spatial precision and recall for text and visual regions, and report their harmonic mean (F1). Unlike pure coverage metrics, this penalizes both missed regions (recall) and over-segmentation or bloated boxes (precision), ensuring tight and complete proposals that do not clutter SoM inputs. We additionally report Hierarchy Consistency (HC), the fraction of pages whose predicted layout trees satisfy structural constraints.

For the *Localize* stage, we introduce the List Structure Penalty (LSP) to assess proper list formatting, while established metrics such as Tree Edit Distance Similarity (TEDS) [44] for tables and Character Detection Matching (CDM) [54] for formulas are also employed.

We further report Routing Accuracy (RA) and Sub-HTML Validity Rate (SVR):

$$\text{RA} = \frac{\# \text{Correct Routes}}{\# \text{Total Regions}}, \quad \text{SVR} = \frac{\# \text{Valid DOM Fragments}}{\# \text{Generated Fragments}}.$$

Reflection behavior is quantified via Reflection Gain (RG):

$$\text{RG} = \frac{1}{K} \sum_{k=1}^K \left(S^{(k)} - S^{(k-1)} \right).$$

Table 2: **Quantitative comparison of document-to-HTML methods across four evaluation dimensions.** Text Extraction (TE: WRR, CRR), Logical Structure (LS: NTED), Physical Structure (PS: GPS, LPS), and Visual Fidelity (VF: VFS via VLM-as-a-judge) are reported. The overall score (OS), computed as the mean of TE, LS, PS, and VF (Section 5.1), is reported as OS_{all} (all languages), OS_{en} (English), and OS_{zh} (Chinese). Baselines grouped into four categories: Pipeline Tools, Expert VLMs, General VLMs, and Screenshot2HTML. REPLICA represents our agentic method.

Method	T.E.		L.S.	P.S.		V.F.	Overall		
	WRR (↑)	CRR (↑)	NTED (↓)	LPS (↑)	GPS (↑)	VFS (↑)	OS_{all} (↑)	OS_{en} (↑)	OS_{zh} (↑)
Marker [43]	0.68	0.82	0.61	0.19	0.34	0.74	0.53	0.60	0.45
Docling [32]	0.40	0.49	0.62	0.17	0.26	0.55	0.38	0.45	0.12
roImOCR [2]	0.54	0.67	0.95	0.15	0.07	0.56	0.34	0.42	0.35
smolDocling [36]	0.37	0.54	0.89	0.16	0.23	0.54	0.28	0.35	0.02
olmOCR [45]	0.49	0.64	0.93	0.21	0.09	0.33	0.31	0.38	0.34
Nanonets-OCR-s [33]	0.25	0.42	0.94	0.18	0.10	0.56	0.26	0.34	0.31
GOT-OCR-2.0 [60]	0.36	0.39	0.93	0.15	0.25	0.70	0.32	0.49	0.10
ChatDoc-OCRFlux [5]	0.30	0.38	0.88	0.15	0.24	0.44	0.27	0.39	0.34
Qwen3-VL [64]	0.66	0.79	0.59	0.22	0.30	0.43	0.47	0.48	0.31
Gemma3 [51]	0.58	0.59	0.85	0.19	0.25	0.38	0.36	0.42	0.37
InternVL3 [6]	0.06	0.30	0.82	0.19	0.29	0.31	0.22	0.29	0.27
Gemini-2.5-Pro [8]	0.67	0.42	0.56	0.14	0.31	0.82	0.47	0.47	0.30
GPT-5 [38]	0.65	0.79	0.49	0.21	0.43	0.86	0.58	0.62	0.50
Webcode2m [14]	0.33	0.45	0.82	0.15	0.29	0.39	0.30	0.42	0.13
Waffle VLM Websight [30]	0.14	0.42	0.84	0.11	0.29	0.19	0.22	0.27	0.11
REPLICA (Ours)	0.88	0.93	0.20	0.73	0.86	0.93	0.86	0.93	0.89

5.3 Tool-Level Reliability

$$TCSR = \frac{\# \text{Successful Tool Calls}}{\# \text{Total Tool Calls}}, \quad FFS = \frac{\# \text{No-Overflow Paragraphs}}{\# \text{Total Paragraphs}},$$

$$BSS = \frac{\# \text{Successful Background Restorations}}{\# \text{Eligible Pages}},$$

Reading Order Accuracy (ROA) is measured via Kendall- τ against ground-truth reading order.

Human Validation. Pearson correlation between seven expert annotators and metric-induced rankings exceeds 0.8 across 100 documents, confirming alignment with expert judgment.

6 Results

Overall Performance: REPLICA achieves the best overall results (table 2), with near-perfect text extraction, strong logical structure preservation, accurate physical structure recovery (over 85%), and the highest visual fidelity (93%). In contrast, all baselines trade off across axes — parsers sacrifice layout, VLMs sacrifice structure or positioning, and Screenshot2HTML models fail to generalize

Table 3: **REPLICA Framework Ablations.** Incremental impact of Segment, Agentic Localize (Router + Specialist Agents), and Refine components on Overall Score (OS), Visual Fidelity (VFS), and Global Position Score (GPS). Each row adds the specified component to the previous configuration.

Configuration	OS ($\uparrow$)	VFS ($\uparrow$)	GPS ($\uparrow$)
Flat Layout (DLN)	0.53	0.74	0.34
+ Hierarchical Layout (IDLDP-DLN)	0.60	0.79	0.48
+ Hi-SAM (Full Segment)	0.67	0.82	0.63
+ SoM in Segment	0.71	0.85	0.69
+ Monolithic Localize (w/o Router & Specialists)	0.76	0.88	0.72
+ Agentic Localize (Router + Specialists)	0.86	0.90	0.77
+ Font-Size Fix (fs)	0.89	0.92	0.79
+ Background Restore (bg)	0.90	0.93	0.80
+ Reflection (final)	0.93	0.97	0.83

to document settings — whereas REPLICA excels across all simultaneously. Overall, REPLICA sets a new state of the art by excelling across all axes simultaneously.

Robustness across languages and document types. REPLICA sustains high performance across 15 languages and 10+ document types, with minimal variance in text accuracy, structural fidelity, and visual alignment across scripts and layouts. In contrast, existing approaches degrade sharply beyond English and Chinese and remain confined to narrow domains or simple layouts, while REPLICA generalizes reliably across diverse formats.

Qualitative Examples. Figures 1, 2 present qualitative examples of REPLICA’s faithful visual reconstruction across a diverse range of document types - including historic newspapers, multi-column magazines, and financial documents. REPLICA consistently reproduces the original document’s appearance in FID-HTML, accurately capturing column boundaries, typography, and spatial layout.

6.1 Ablations

REPLICA Framework Ablations. As shown in table 3, performance improves consistently with each modular component, validating the agentic decomposition. Strengthening the Segment stage yields large gains in spatial alignment and overall score; introducing the agentic Router and Specialist formulation further produces a substantial jump over monolithic localization; and refinement steps (font correction, background stitching, reflection) deliver steady improvements, culminating in the strongest overall fidelity. The results demonstrate that high-quality reconstruction emerges from coordinated modular reasoning rather than any single stage.

Table 4: **Agentic and Tool Reliability Metrics.** Trace-based evaluation of intermediate agents and tool calls.

Category	Metric	Score
Segment Agent	Agnostic Text F1 (AT-F1)	0.98
	Agnostic Visual F1 (AV-F1)	0.99
	Hierarchy Consistency (HC)	0.98
Router Agent	Routing Accuracy (RA)	0.97
Localize Agent	Sub-HTML Validity Rate (SVR)	0.96
Reflection Agent	Reflection Gain (RG)	+0.06
Tools	Tool Call Success Rate (TCSR)	0.995
	Reading Order Accuracy (ROA)	0.98
	Font Fit Success (FFS)	0.97
	Background Stitch Success (BSS)	0.96

Agentic and Tool Reliability. Trace-based evaluation in table 4 confirms high reliability across coverage, routing, sub-HTML generation, and tool execution, with reflection delivering consistent gains — demonstrating that performance is driven by dependable structured reasoning rather than any single component.

6.2 Limitations

While REPLICA delivers high-fidelity single-page reconstruction, it does not yet support multi-page consistency, potentially fragmenting cross-page structures (e.g., spanning tables or figures). Text attributes (bold, italics, underline, color) are reliably recovered, but precise font identification remains unresolved. Existing methods such as DeepFont [58] and SwoRNet [26] offer limited, mostly English/Chinese coverage, and large-scale multilingual font recognition remains an open challenge.

7 Conclusion

We introduce REPLICA, an agentic framework for converting raw document image into high-quality document representation (HTML) while preserving both structure and visual appearance. Instead of relying on a single model to do everything, our approach breaks the problem into coordinated stages that employs various agents; detecting layout, generating content with specialized agents, assembling the document in the correct order, and refining the final output based on how it looks after rendering. By combining structured tools with reasoning agents, REPLICA can correct layout mistakes, maintain proper hierarchy, and resolve visual inconsistencies that arise during rendering. The resulting HTML is not only machine-readable but also visually faithful to the original document, enabling reliable use in accessibility, publishing, and document understanding applications.

8 Acknowledgments

Our sincere gratitude goes to Amruth Sagar K N and late B.V. Khadiravana for their invaluable support. The project was supported by BharatGen at IIIT Hyderabad.

References

1. A. S. Inc.: Pymupdf. <https://github.com/pymupdf/PyMuPDF> (2024) 2
2. AI, R.: Rolmocr: A faster, lighter open source ocr model (2025) 12
3. Anthropic: Claude 3.5 sonnet. <https://www.anthropic.com/news/claude-3-5-sonnet> (2024) 4
4. Blecher, L., et al.: Nougat: Neural optical understanding for academic documents. In: ICLR (2023) 4
5. chatdoc-com: OCRFlux: A multimodal toolkit for converting pdfs and images into markdown. <https://github.com/chatdoc-com/OCRFlux> (2025) 12
6. Chen, Z., et al.: Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In: CVPR (2024) 4, 12
7. Chen, Z., et al.: MDEval: Evaluating and enhancing markdown awareness in large language models. In: WWW (2025) 10
8. Comanici, G., et al.: Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next-generation agentic capabilities. arXiv **2507.06261** (2025) 4, 12
9. Cui, C., et al.: PaddleOCR 3.0 technical report. arXiv **2507.05595** (2025) 1
10. Developers, P.: Pandoc User’s Guide. Pandoc (2025), <https://pandoc.org/MANUAL.html> 2
11. Documentation, U.: pdf2htmlEX Manual (Ubuntu Xenial manpage) (2016), <https://manpages.ubuntu.com/manpages/xenial/man1/pdf2htmlEX.1.html> 2
12. Feng, H., et al.: Dolphin: Document image parsing via heterogeneous anchor prompting. arXiv **2505.14059** (2025) 2
13. Gui, Y., et al.: UICopilot: Automating ui synthesis via hierarchical code generation from webpage designs. In: WWW (2025) 4
14. Gui, Y., et al.: Webcode2m: A real-world dataset for code generation from webpage designs. In: WWW (2025) 4, 12
15. He, Y., et al.: Pingan-vcgroup’s solution for icdar 2021 competition on scientific table image recognition to latex. arXiv **2105.01846** (2021) 4
16. Hu, A., et al.: mPLUG-DocOwl2: High-resolution compressing for ocr-free multi-page document understanding. arXiv **2409.03420** (2024) 4
17. Huang, Y., et al.: Layoutlmv3: Pre-training for document ai with unified text and image masking. In: WWW (2022) 4
18. Jaided AI: Easyocr: Ready-to-use ocr with 80+ supported languages. <https://github.com/JaidedAI/EasyOCR> (2024) 1, 4
19. Kayal, P., et al.: Tables to latex: Structure and content extraction from scientific tables. IJDAR **26** (2022) 4
20. Kim, G., et al.: DONUT: Ocr-free document understanding transformer. In: ECCV (2022) 4
21. Kolthoff, K., et al.: Zero-shot prompting approaches for llm-based graphical user interface generation. arXiv **2412.11328** (2024) 4

22. Kumar, R., et al.: TexTAR: Textual attribute recognition in multi-domain and multi-lingual document images. In: ICDAR (2025) 10
23. Li, C., et al.: PP-OCrv3: More attempts for the improvement of ultra-lightweight ocr system. arXiv **2206.03001** (2022) 4
24. Li, M., et al.: TableBank: Table benchmark for image-based table detection and recognition. In: LREC (2020) 10
25. Li, X., et al.: DREAM: Document reconstruction via end-to-end autoregressive model. arXiv **2507.05805** (2025) 4
26. Li, X., et al.: SwordNet: Chinese character font style recognition network. IEEE Access **10** (2022) 14
27. Li, Z., et al.: Monkey: Image resolution and text labels are important for large multimodal models. In: CVPR (2024) 4
28. Li, Z., et al.: MonkeyOCR: Document parsing with a structure-recognition-relation triplet paradigm. arXiv **2506.05218** (2025) 2
29. Li, Z., et al.: Readoc: A unified benchmark for realistic document structured extraction. arXiv **2409.05137** (2024) 10
30. Liang, S., et al.: WAFFLE: Multi-modal model for automated front-end development. arXiv **2410.18362** (2024) 4, 12
31. Liu, Y., et al.: TextMonkey: An OCR-free large multimodal model for document understanding. arXiv **2403.04473** (2024) 4
32. Livathinos, N., et al.: Docling: An efficient open-source toolkit for ai-driven document conversion. arXiv **2501.17887** (2025) 2, 4, 12
33. Mandal, S., et al.: Nanonets-OCR-S: A model for transforming documents into structured markdown with intelligent content recognition and semantic tagging. arXiv (2025) 12
34. Miller, B.: Latexml the manual. Web document (2011) 2
35. Mindee: doctr: Document text recognition. <https://github.com/mindee/doctr> (2021) 10
36. Nassar, K., et al.: Smoldocling: Lightweight document understanding via efficient vision-language modeling. arXiv preprint arXiv:2501.04567 (2025) 2, 5, 12
37. Nath, O., et al.: IndicDLP: A foundational dataset for multi-lingual and multi-domain document layout parsing. In: ICDAR (2025) 10
38. OpenAI: Gpt-5 system card. Tech. rep., OpenAI (Aug 2025), <https://cdn.openai.com/gpt-5-system-card.pdf> 2, 12
39. Ouyang, L., et al.: OmniDocBench: Benchmarking diverse pdf document parsing with comprehensive annotations. In: CVPR (2025) 10
40. Ouyang, R., et al.: PDFMathTranslate: Scientific document translation preserving layouts. arXiv **2507.03009** (2025) 4
41. P. Team: Pypdfium2: Python bindings for pdfium. <https://github.com/pypdfium2-team/pypdfium2> (2024) 2
42. Padhi, I., et al.: TabFormer: Tabular transformers for modeling multivariate time series. In: ICASSP (2021) 4
43. Paruchuri, V.: Marker. <https://github.com/datalab-to/marker> (2025) 2, 4, 12
44. Pawlik, M., Augsten, N.: Tree edit distance: Robust and memory-efficient. Inf. Syst. **56** (2016) 11
45. Poznanski, A., et al.: olmocr: Open layout modeling for robust ocr-free document understanding. arXiv preprint arXiv:2502.01983 (2025) 2, 4, 10, 12
46. pypdf Maintainers: pypdf: A pure-python pdf library. <https://github.com/py-pdf/pypdf> (2024) 2
47. Reed, S.: textfit. <https://github.com/STRML/textFit> (2014) 10

48. Safnuk, B., Hu, G.: Reconstructing latex source files from generated pdfs: A neural network approach. In: INDIN. pp. 890–895 (2018) 4
49. Si, C., et al.: Design2Code: Benchmarking multimodal code generation for automated front-end engineering. arXiv **2403.03163** (2024) 4
50. Smith, R.: An overview of the Tesseract ocr engine. In: ICDAR (2007) 1, 5
51. Team, G., et al.: Gemma 3 technical report. arXiv **2503.19786** (2025) 10, 12
52. Unstructured-IO: Unstructured: Open-source tools for unstructured data processing. <https://github.com/Unstructured-IO> (2025) 5
53. Wan, Y., et al.: Divide-and-conquer: Generating ui code from screenshots. arXiv **2406.16386** (2025) 4
54. Wang, B., et al.: CDM: A reliable metric for fair and accurate formula recognition evaluation. arXiv **2409** (2024) 11
55. Wang, B., et al.: MinerU: An open-source solution for precise document content extraction. arXiv **2409.18839** (2024) 4
56. Wang, B., et al.: UniMERNet: A universal network for real-world mathematical expression recognition. arXiv **2404.15254** (2024) 4
57. Wang, Z., Liu, J.C.: Translating math formula images to latex sequences using deep neural networks with sequence-level training. IJDAR **24** (2021) 4
58. Wang, Z., et al.: DeepFont: Identify your font from an image. In: ACM MM (2015) 14
59. Wang, Z., et al.: LayoutReader: Pre-training of text and layout for reading order detection. In: EMNLP (2021) 4
60. Wei, H., et al.: General OCR theory: Towards OCR-2.0 via a unified end-to-end model. arXiv **2409.01704** (2024) 4, 12
61. Wu, F., et al.: Mllm-based ui2code automation guided by ui layout information. Proc. ACM Softw. Eng. **2** (2025) 4
62. Xiao, S., et al.: Ui semantic component group detection: Grouping ui elements with similar semantics in mobile graphical user interface. Displays **83** (2024) 4
63. Xie, M., et al.: UIED: A hybrid tool for gui element detection. In: ESEC/FSE (2020) 4
64. Yang, A., et al.: Qwen3 technical report. arXiv **2505.09388** (2025) 2, 4, 5, 10, 12
65. Yang, J., et al.: Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. arXiv **2310.11441** (2023) 7
66. Yang, Z., et al.: The dawn of LMMs: Preliminary explorations with GPT-4v. arXiv **2309.17421** (2023) 4
67. Yang, Z., et al.: CC-OCR: A comprehensive and challenging ocr benchmark for evaluating large multimodal models in literacy. arXiv **2412.02210** (2024) 10
68. Yao, Y., et al.: MiniCPM-V: A GPT-4v-level MLLM on your phone. arXiv **2408.01800** (2024) 4
69. Ye, M., et al.: Hi-SAM: Marrying segment anything model for hierarchical text segmentation. TPAMI (2024) 10
70. Yun, S., et al.: Web2Code: A large-scale webpage-to-code dataset and evaluation framework for multimodal llms. NeurIPS **37** (2024) 4
71. Zhang, T., et al.: NLDesign: A ui design tool for natural language interfaces. In: TACC-China (2024) 4
72. Zhao, Y., et al.: Detrs beat yolos on real-time object detection. In: CVPR (2024) 4
73. Zhao, Z., et al.: DocLayout-YOLO: Enhancing document layout analysis through diverse synthetic data and global-to-local adaptive perception. arXiv **2410.12628** (2024) 10