FormGym: Doing Paperwork with Agents

Matthew Toles[1], Isaac Song[2,3], Rattandeep Singh[1], Zhou Yu[1,3],

[1]Columbia University, [2]Georgia Institute of Technology, [3]Arklex.ai,

Correspondence: mt3639@columbia.edu

Abstract

End-to-end form filling refers to automatically populating fields in a document-style form with the appropriate information derived from external data. Although prevalent and useful, no formal benchmark exists for evaluating systems' form completion accuracy. Existing datasets focus on parsing, extraction and web form interaction, rather than end-to-end completion of document-style forms. We propose FormGym, a benchmark formulation of the end-to-end form filling task that evaluates form completion and accuracy. We construct FormGym by repurposing three existing datasets and add one new dataset to achieve more challenging, diverse, and realistic test cases. Our studies show baseline vision language agents (VLAs) perform poorly on FormGym in every scenario, primarily due to poor field localization. GUI agents perform better but suffer from high latency and costs. Therefore we also introduce FieldFinder, a field localization tool that enables zero-shot VLAs to find and accurately place text in input fields. We find that VLAs augmented with FieldFinder achieve better performance compared to baselines in all models.

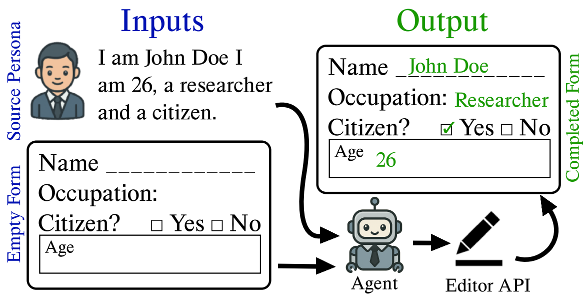


Figure 1: In the FormGym task, agents are provided with an unfilled source form and a user persona containing answers to fields in the form. The agent must use an editor API to produce the completed source form. Diverse layout semantics indicate suggested fields, such as underlines, colons, check boxes, and table cells.

1 Introduction

Filling out paperwork is a pervasive and tedious task. The US government estimates that federal agencies collectively generate nearly 10 billion hours of mandatory paperwork each year (USOMB, 2016). Ad-hoc tools and advances among digital agents in related fields have shown potential to substantially reduce this burden (Ghosh et al., 2024). However, we are unaware of any formal benchmark to evaluate automatic form filling. This results in uncertainty and risk for users, especially on forms carrying legal or financial weight.

Existing literature addresses components of end-to-end (E2E) form filling, including document understanding, question answering, tool use, and image manipulation. Work on GUI agent systems and benchmarks has primarily targeted browser and other desktop UIs. However, we are concerned with printable, PDF-style images of target documents. Lacking the same appearance and affordances as GUIs, these documents will challenge existing strategies, especially systems reliant on underlying structures such as the DOM or metadata.

In form filling tasks, each target document contains one or more empty fields, each associated with a field name (e.g. Occupation) in the document indicating the desired field value (e.g. Researcher) through layout semantics (e.g. proximity) (Figure 1). Field values are derived from an external source persona, such as free text or an image of a related document. We formalize the E2E form filling task mapping unfilled forms and source persona to filled forms as follows: for each field in the unfilled form, place an appropriate string inside its bounding box.

We introduce FormGym, a novel benchmark dataset and evaluation framework for end-to-end form filling. To construct FormGym, we leverage three existing datasets originally constructed for document understanding. From these, we constructing ground truth field (name, bounding box,

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics
Volume 1: Long Papers, pages 3771–3785
March 24-29, 2026

value) tuples for empty forms by redacting values. However, because of their original annotation schemes, it is difficult to construct realistic source personas without the field values appearing verbatim. We therefore extend our benchmark by manually annotating the Auto Loans dataset with a more sophisticated schema. This allows FormGym to address longer forms, incorporate secondary source documents into source personas, and include fields that require composing values from multiple facts rather than copying text directly.

Noting near-zero accuracy by baseline vision language agents (VLAs), we identify field localization as the main bottleneck in end-to-end form filling, even in frontier models. Rather than train purpose-built agents, we would prefer to provide flexible infrastructure adaptable to any new LLM. Hence, we develop FieldFinder, an open vocabulary model trained to output the bounding box of the field associated with an input field name. As a lightweight tool, FieldFinder allows VLAs to specify the target field by name rather than cartesian coordinates. We demonstrate that FieldFinder improves VLA form filling accuracy across all scenarios and models with minimal latency and memory overhead (Figure 1). Our contributions are twofold:

- **A benchmark for evaluating agents on end-to-end form completion**, showing that current VLAs and Claude Computer Use struggle to accurately identify field placements.
- **An open-vocabulary field localization model**, showing that it helps VLAs overcome spatial reasoning limitations.

Our code and dataset are available at https://github.com/mtoles/formgym

2 Related Work

Several benchmarks exist for evaluating document layout (Zhong et al., 2019; Pfitzmann et al., 2022; Li et al., 2020, 2019; Harley et al., 2015; Li et al., 2019). Numerous frameworks (Xu et al., 2020; Li et al., 2021; Bao et al., 2020; Appalaraju et al., 2021; Lee et al., 2022; Lu et al., 2024) have been proposed for navigating these types of tasks (Table 3). Despite deep exploration of document understanding, prior work has not addressed document elements that are suggested only by negative space in the document (e.g., Occupation in Figure 1). Some challenging examples include non-underlined fields, table cells or those indicated merely by a colon (e.g., "Name: "). Existing commercial software, such as Mac OS Preview and Amazon Textract, can localize some but not all suggested elements.

Unlike traditional question answer-style (QA) benchmarks, vision language and graphical user interface (GUI) agent evaluations generally measure a path-independent end-state, such as in Zhou et al. (2023), Zheng et al. (2022), Liu et al. (2023), Yao et al. (2024), and He et al. (2024), which often include elements of form-filling. GUI agents including (Qin et al., 2025a; Gou et al., 2025) generally focus training on GUIs rather than flat document images and interact with documents through browser UIs rather than editor APIs. In contrast, our work explores end-to-end, real-world completion of PDF-style image domain forms.

Computer use agents including Claude Computer Use (Anthropic, 2024) and Openai Operator (OpenAI, 2025) show potential to address these types of tasks. Open source frameworks have also been proposed, including Shen et al. (2024a); Qin et al. (2025b); Liu et al. (2024); Hong et al. (2024); Wei et al. (2025); Qi et al. (2025); Ma et al. (2023); Putta et al. (2024); Shen et al. (2024b). However, many of these solutions exploit structured elements such as the DOM, or are limited in their domain or interaction methods.

3 FormGym Benchmark

The FormGym dataset is composed of documents from three existing datasets plus one novel dataset. While existing datasets represent a large quantity of diverse, multilingual documents, our Auto Loans dataset introduces some of the most challenging scenarios in the form filling domain.

3.1 Conversion from Existing Datasets

We draw examples from the FUNSD (Jaume et al., 2019), XFUND (Xu et al., 2022), and Form-NLU (Ding et al., 2023) document relation datasets, representing scanned English, scanned multilingual, and digital-born English financial documents, respectively (Figure 2). We choose these datasets because they contain a large quantity of relationship annotations between document field names and values, as well their bounding boxes (Table 1). FUNSD is composed scanned documents from the Truth Tobacco Industry Document archive (University of California, San Francisco). XFUND is composed of scanned documents from Common

Crawl in seven languages (Common Crawl, 2023). Form-NLU is composed of Australian financial filings. For each key-value relation, we create one (document image, field name, field value, value bounding box) example. We extract names, values, and bounding boxes from document annotations. On these documents, the correctness of predicted field values is determined by exact match. Because these datasets include only completed documents, we then create empty fillable documents by deleting value text from filled fields with horizontal inward content-aware fill.[1]

3.2 Creating A New Auto Loans Dataset

While the above documents present a diversity of forms, the automatic nature of example creation does not explore the full complexity of real-world form filling. Specifically, field names and values may be phrased differently from the source persona (e.g. "previous address" vs. "last place of residence") and may require generation based on multiple facts, such as a street, city, and postal code. In addition, forms in existing datasets contain relatively few fields per form, substantially reducing the challenge of field localization. Finally, users may wish to have forms completed based on the content of other document images to avoid transcribing relevant information. In this case, we will require multiple forms containing partially overlapping field values. Therefore, we construct the additional Auto Loans dataset.

We searched the web for American auto loan application PDF files, choosing the single page with the most fields. One author then manually annotated the bounding box of each empty field. Another author reviewed these annotations and together they discussed and corrected any disagreements. We present additional annotation details in (Appendix A). We defined multiple source personas ensuring that all information necessary for every form be present, such as the applicant's first, middle, and last name. This increases the diversity of source personas and enables better coverage of certain form fields, including check boxes, which are are sparsely filled. Observing that most enumerated options in these fields have four or fewer choices, we construct four personas, each spanning all documents. Finally, for each field, we define a function that determines prediction correctness based on one or more source persona elements. For example, the Full Name field will be marked as correct iff `input == source_data.firstname + " " + source_data.lastname`.

Defining source personas independently from form fields allows several forms of flexibility not available in XFUND, FUNSD, and Form-NLU. Specifically, it allows us to accept a broader range of inputs, for example, different phone number formats. It also permits automatically propagating other documents to serve as ground truth source persona inputs for the Auto Loans (Image) task (Section 3.2). By manually annotating empty forms instead of automatically redacting completed ones, we avoid artifacts from content-aware fill and biases from original text placement. This yields a subset with maximal evaluation precision.

3.3 FormGym Dataset Statistics

In total, FormGym contains 25,466 train and 3,889 test field examples spread across 1,709 train and 215 test forms (Table 1). The dataset contains between 7.2 and 88.6 fields per form with an average of 18.1. 46% of test documents are in one of seven non-English languages. 65% of documents are scanned versus digital-born.

4 FieldFinder Field Localizer

We find that baseline VLAs perform poorly, primarily due to poor field localization. We therefore design a field localization tool called FieldFinder that augments any VLA by taking on the localization task. FieldFinder is trained to take a form image and text description of the name of the target field as input and predicts the bounding box around the valid input space (Figure 3). Functionally, FieldFinder is an open-vocabulary object detection model, where the target objects are suggested text input fields associated with nearby text that matches the input string.

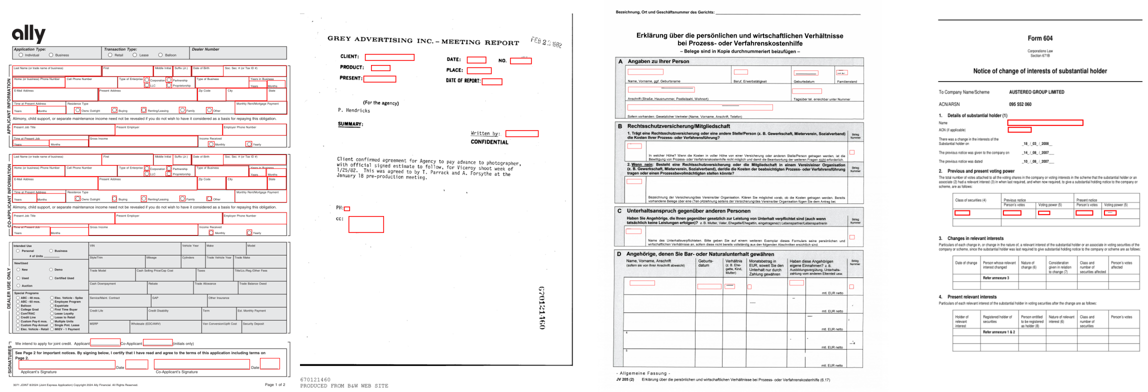
4.1 FieldFinder Dataset

We draw examples from the FUNSD, XFUND, and Form-NLU train datasets (Figure 2). In some cases, field names can only be described precisely by including additional hierarchical information, such as section headers. For example, in several Auto Loans forms, personal information for the applicant and the coapplicant are only indicated based on a distant section header. Similarly, specific table cells must be described by row, column, and table headers. We therefore prepend descriptions

[1] github.com/light-and-ray/resynthesizer-python-lib

	Forms		Fields		Fields/Form		Sources	Lang.	Domain	Quality	Comp.
	Train	Test	Train	Test	Train	Test					
Auto Loans	-	10	-	886		88.6	4	1	Auto Loans	Digital	✓
FUNSD	155	39	2,246	577	14.5	14.8	1	1	US Government	Scanned	
XFUND	1,112	100	19,559	1,950	17.6	19.5	1	7	Common Crawl	Scanned	
Form-NLU	442	66	3,661	476	8.3	7.2	1	1	AU Stock Exchange	Digital	
Total	1,709	215	25,466	3,889	14.9	18.1					

Table 1: FormGym Dataset statistics. Sources indicates the number of source personas for field value and prompt generation. Lang. indicates the number of languages. Comp. indicates whether field values must be generated by composing from multiple facts rather than merely copying a span.



(a) Auto Loans (b) FUNSD (c) XFUND (d) Form-NLU

Figure 2: Example forms and field bounding boxes in the FormGym dataset.

of all hierarchically superior elements to the input text (i.e., `Section 1 | Members Table | Names Column | Row 1` → `John Doe`). Examples consist of (field name, field bounding box) pairs such that FieldFinder can learn to localize the bounding box of a field's value given its name.

4.2 FieldFinder Training

We fine-tune a Florence 2 Large (Xiao et al., 2024) vision foundation model to predict the field bounding box given the target field name string and form image. We choose Florence 2 because its pretraining contains both open-vocabulary object detection and tasks requiring OCR, minimizing the distribution shift between pretraining and fine-tuning. Florence 2 Large has only 0.77B parameters, contributing minimal latency and memory overhead when augmenting with much larger VLAs. We train the FieldFinder for 6 epochs, batch size 8, learning rate 1e-6 with cosine profile on 1x NVIDIA A100 GPU for 30 hours. We selected these parameters using grid search across 2-4 options per parameter.

4.3 FieldFinder Results

We choose accuracy as our primary metric, defined as whether the center of the predicted bounding box falls within the ground truth bounding box. FieldFinder achives 54.3% accuracy on average across the the test sets of all FUNSD, XFUND, and Form-NLU, compared to near-zero accuracy by baseline VLAs. However, we note a wide disparity in performance depending on the dataset (Figure 4). We see the strongest performance in Form-NLU at 80.5% accuracy, which we attribute to its few fields per form and lack of scan artifacts. Accuracy decreases on FUNSD to 57.4%, which contain low quality scans. XFUND presents an even greater challenge due to its multilingual scans, with FieldFinder reaching 24.9% accuracy. Finally, we find weak performance on Auto Loans at only 6.9% accuracy, which did not appear in the training data and contains by far the most fields per form, at 88.6 fields per form. We conclude that FieldFinder can perform well on high quality English documents with few fields per form, but struggles in other contexts and out-of-distribution data.

Although distribution shift likely affects quality

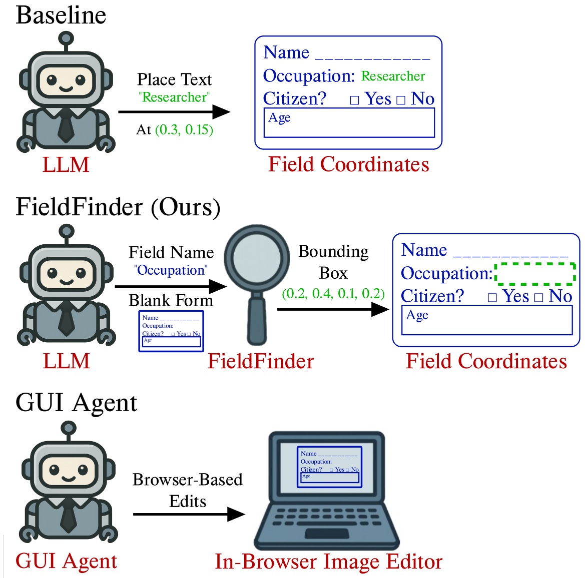


Figure 3: In the baseline case, the LLM receives an unfilled source form and persona information in its context and attempts to complete the form through a text placement API based on (x, y) coordinates. In the FieldFinder (ours) case, the text placement API is replaced with the FieldFinder tool. Instead of using coordinates as input, the FieldFinder tool takes the name of a field as input, then uses an open vocabulary object detection model to detect the corresponding field boudning box. In the GUI Agent case, the GUI agent uses an in-browser image editing tool (designed for humans) to place text on the PDf.

on Auto Loans, we draw attention to a confounding factor; Auto Loans documents are so densely covered with fillable fields (88.6 per page) that there is a high probability for an errant placement to land in an unrelated field, resulting in two errors—an empty filed and a doubly filled field. When accuracy is plotted against fields per form on a log scale, English datasets Form-NLU, FUNSD, and Auto Loans shows a clear negative linear trend (Figure 4). This further indicates that FieldFinder accuracy highly dependent on the number of fields per page.

5 End-to-End Form Filling Pipelines

Noting the strong performance of VLAs on related visual reasoning and QA tasks, we instantiate an E2E form filling VLA pipeline for their evaluation. While many methods have been proposed for related tasks, only VLAs possess the necessary capabilities to fully perform end-to-end form filling (Appendix C). We construct parallel pipelines for VLAs augmented with our FieldFinder tool and Claude Comptuer Use GUI agent.

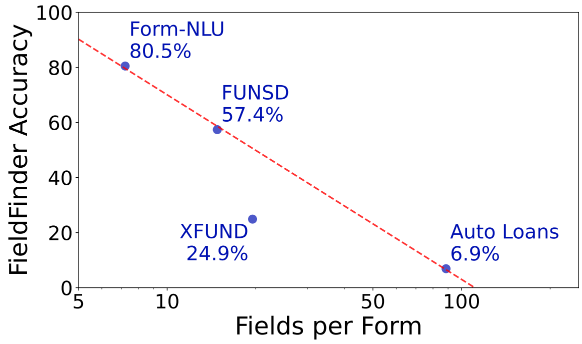


Figure 4: FieldFinder accuracy vs. fields per form (log scale). Trend line shown for English datasets only. The trend suggests that high numbers of fields per form and multi-lingual forms are the greatest challenges for FieldFinder.

5.1 VLA Pipeline

We choose leading open- and closed-source VLAs as our baselines because they represent the only end-to-end solutions that the necessary criteria in table 3. We select Llava 7B, Molmo 7B, and Aria 25B as our open-source; and Claude 4 and GPT-5 as our frontier VLAs. We prompt VLAs with the source image, source persona and form editing API documentation. We also evaluate Claude 4 + Set of Marks (Yang et al., 2023), overlaying a grid with coordinate references at vertices.

To leverage VLAs existing tool use capabilities, we construct a minimal image editing API by which they can edit the form. The image editing API permits placing text or signatures/initials on the image by specifying the string and (x, y) coordinates. It also allows for deleting all input text intersecting an (x, y) coordinate and terminating a session early.

5.2 FieldFinder Pipeline

We evaluate VLAs augmented with FieldFinder in a pipeline consistent with the previously described VLA pipeline. However, we replace the text and signature placement APIs with a FieldFinder API. Rather than taking Cartesian coordinates as input along with the field value string, the FieldFinder API takes a field name and value. It then places the value at the location detected by FieldFinder based on the field name.

5.3 GUI Agent Pipeline

We instantiate the Claude Computer use Use GUI agent with the free in-browser photo editing application Photopea[2], whose interface is nearly identi-

[2]photopea.com

cal to Adobe Photoshop. We prompt Claude Computer Use with the natural language source persona and instructions to complete the form. Prompts include detailed instructions on how to use the Photopea interface, without which GUI agents fail completely. We provide additional implementation details in Appendix E.1. For accessibility and cost reasons, we limit operators to five minutes per page and downsample test sets to 10 documents per source.

6 Experiments and Results

6.1 Pipeline Settings

Because Auto Loans documents contain numerous fields per page, we evaluate VLAs under two settings:

One-Shot - The agent must place all text at once.

Iterative - The agent may take multiple sets of actions over the course of up to 5 rounds, allowing it to correct mistakes. Agents receive feedback on the success or failure of each action.

The Iterative setting allows agents to adapt to tooling and to realize and correct mistakes using the deletion API endpoint. The GUI agent pipeline is inherently iterative. We present source personas in two forms:

Text - The source persona is presented as plain text.

Image - The source persona is presented as an image of another auto loan document. Persona data not appearing in the image is presented in plain text.

We evaluate VLAs in our E2E form filling pipeline with and without FieldFinder on the FormGym dataset. We find FieldFinder improves accuracy in all cases.

6.2 Metrics

We evaluate field completion correctness based on whether each field contains the correct value, as defined in Sections 3.1 and 3.2. If a field contains multiple text inputs, we concatenate them with a space. We choose field accuracy as our primary evaluation metric, ignoring those that should be empty according to the ground truth label to avoid inflating accuracy. A text input is considered to be inside a field if the center point of the text bounding box is within the field. A side effect of this rule is the increased difficulty of forms with numerous fields. On such forms, inaccurate placements are more likely to fall into unrelated fields, resulting in a double error.

We choose center point correctness instead of other metrics including intersect-over-union or full enclosure because it does not rely on *a priori* knowledge of text size for models to accurately place text. It also avoids ambiguity in how to judge text that slightly overhangs out of a field or into another field when, to a human, the intention of the form filler would be obvious.

6.3 Performance Comparison

Overall, VLAs struggle on FormGym, with models performing best on Form-NLU and worst on Auto Loans (Table 2). Baseline VLAs generally score $\leq 3\%$. Set of Marks VLA augmentation offers no improvement over baseline VLA setup. Claude Comptuer Use, however, achieves an average accuracy of 21.1% Our method using Claude 4 + FieldFinder, achieves the highest accuracy at 23.0%.

One outlier is baseline VLA Claude 4's performance on FUNSD, achieving 21.0%. It appears that, in some cases, Claude can performing vertical but not horizontal localization, frequently generating $x = 0.5$. Because FUNSD fields are very wide, this results in higher accuracy. We observe a similar effect in Claude Computer Use, which achieves 60.2% accuracy. However, when introducing FieldFinder, we observe equal or better performance in all cases (Figure 2). Augmenting with FieldFinder, Claude 4 improves from 4.5 to 23.0% average accuracy. In the best case, Claude 4's performance on Form-NLU increases from 0% to 54%. We observe smaller gains, up to 9.2% percentage points on Auto Loans (Text) (Figure 5). Notably, Claude 4 with FieldFinder surpasses Claude Computer Use on both digital-born datasets, Auto Loans and Form-NLU. All open-source models also achieve performance improvements with FieldFinder, with Aria leveraging the tool better than smaller models. GPT-5 and Claude appear to struggle to chained reasoning in the more complex Auto Loans (Image) task, where accuracy falls by around 50% compared to when source information is supplied in text form.

We illustrate the most common FieldFinder errors in Figure 6. Primary failure modes include placing text slightly outside the field and placing text in the wrong field with a lexically similar name. The later case can generate secondary failure by adding additional text into the incorrect field, sometimes superimposing it on the otherwise correct answer.

	AL (Text)		AL (Image)		FUNSD	XFUND	Form-NLU	Average	Cost
	OS	IT	OS	IT	OS	OS	OS		
Aria 25B	0.0	0.0	0.0	0.0	1.0	1.0	0.0	0.4	-
Claude 4	0.0	0.0	0.3	0.7	21.0	1.0	0.0	4.5	0.76
GPT-5	1.0	1.5	0.5	0.7	2.0	2.0	0.0	1.2	**0.30**
Llava 7B	0.0	0.0	0.0	0.0	1.0	1.0	0.0	0.4	-
Molmo 7B	0.0	0.0	-	-	1.0	1.0	0.0	0.5	-
Claude 4 + SoM	0.0	1.0	1.0	1.0	9.0	1.0	0.0	2.3	0.69
Aria 25B + FF (ours)	3.3	4.2	1.5	2.0	20.0	9.0	29.0	12.7	-
Claude 4 + FF (ours)	8.3	9.2	**4.8**	**5.3**	32.0	15.0	**54.0**	**23.0**	0.43
GPT-5 + FF (ours)	**8.5**	**9.8**	3.0	**5.3**	29.0	14.0	50.0	21.3	0.37
Llava 7B + FF (ours)	1.0	1.0	0.3	0.3	4.0	3.0	4.0	2.5	-
Molmo 7B + FF (ours)	0.5	1.3	-	-	9.0	3.0	7.0	5.0	-
Claude Computer Use	2.7	-	1.4	-	**46.2**	**26.7**	32.4	21.1	54.00
Claude 4 (GT Coords)	74.0	90.0	49.2	56.2	83.0	82.0	77.0	75.3	-

Table 2: Percentage of fields containing a correct value. AL: Auto Loans dataset. IT: Iterative. OS: One-Shot. FF: FieldFinder. SoM: Set of Marks. GT: VLA was supplied with ground truth field coordinates in the prompt. Macro average is taken across each dataset. Cost indicates API fees in USD per thousand fields. We do not include Molmo in the Auto Loans (Image) task because it is not pretrained with multiple images.

Our FieldFinder localization tool demonstrates a method for equipping zero-shot VLAs with field localization abilities. Without FieldFinder, VLAs struggle to even approach this task due to a lack of training data involving pixel-level predictions, with frontier models scoring below 1% accuracy in multiple domains. As expected, FieldFinder performs strongest on high quality English document images with relatively few fields. FieldFinder uses only 0.77B parameters, requiring negligible memory and latency overhead when augmenting both closed- and open-source models with no need for additional training.

We note that XFUND accuracy falls substantially below the English dataset trend line, despite constituting the largest portion of training data. We attribute this to the multi-lingual nature of the XFUND dataset, a well studied weakness in models pre-trained primarily on English data (Geigle et al., 2025).

7 Error Analysis

Surprisingly, we see little improvement between One-Shot and Iterative flows, except in the case of Aria, which shows a 1.27-fold increase in accuracy (Table 2). Analyzing VLA trajectories, we attribute this to frontier models calling the `Terminate` action after the first turn in most (or in the case of Claude, all) trajectories. Although these models are prompted with the number of remaining opportunities to edit the form, they appear to lack the self-awareness to doubt their own accuracy and utilize future turns for error correction.

GUI agents present one path to end-to-end form filling, with Claude Computer Use achieving 21.1% accuracy. Qualitatively, we observe that it is particularly challenged by long documents, often failing to finish within the allotted time. It also typically fails on forms containing tables, resulting in only 2.7% accuracy on Auto Loans (Text). Overall, Claude Computer Use falls short of our Claude 4 + FieldFinder method (23.0%). Moreover, Claude Computer Use costs $5.40 per 1000 fields, compared to $0.043 for Claude 4 + FF. Furthermore, Claude Computer Use requires 28 minutes per 100 fields, compared to approximately 20 seconds seconds for Claude 4. Ultimately, we find that Claude Computer Use requires over 100x the cost and 84x the time to complete forms while providing worse overall accuracy.

We conduct studes to attribute errors to either the document understanding or document reasoning steps of the VLA pipeline using Claude 4. Document understanding errors arise from VLAs failing to understand the input document image. Document reasoning errors, in contrast, arise from erroneous generations by the VLA even when provided with ground truth free text descriptions of field names and locations. Compared to the localization errors discussed in Section 4.3, these errors are somewhat less common (Figure 7) overall, but more common in the high quality, low field density Form-NLU dataset.

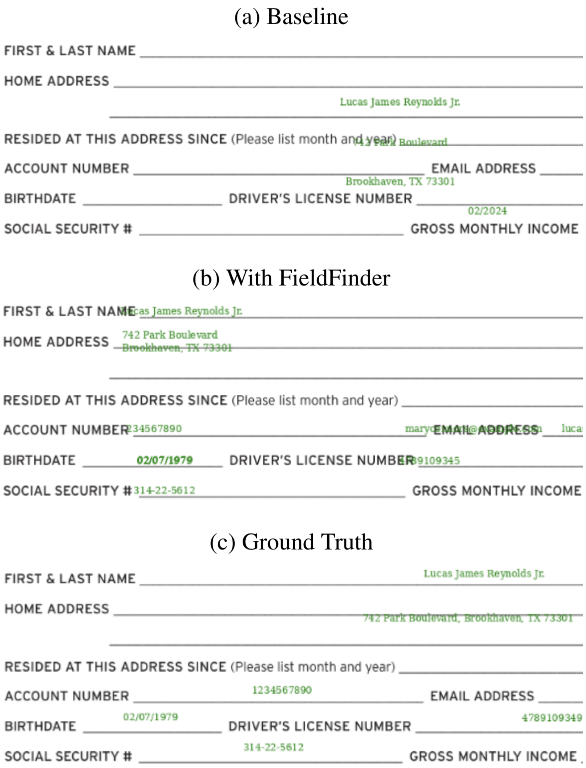


Figure 5: Example output by Claude 4 baseline, with FieldFinder (ours), and ground truth in the Auto Loans (Text) One-Shot task. We attribute FieldFinder's leftward bias to supervision artifacts: training labels mark left-biased value text rather than full fields. Without FieldFinder, Claude appears to struggle more with horizontal spacing that with vertical spacing, assigning most placements an x coordinate of exactly 0.5 (not centered due to figure cropping).

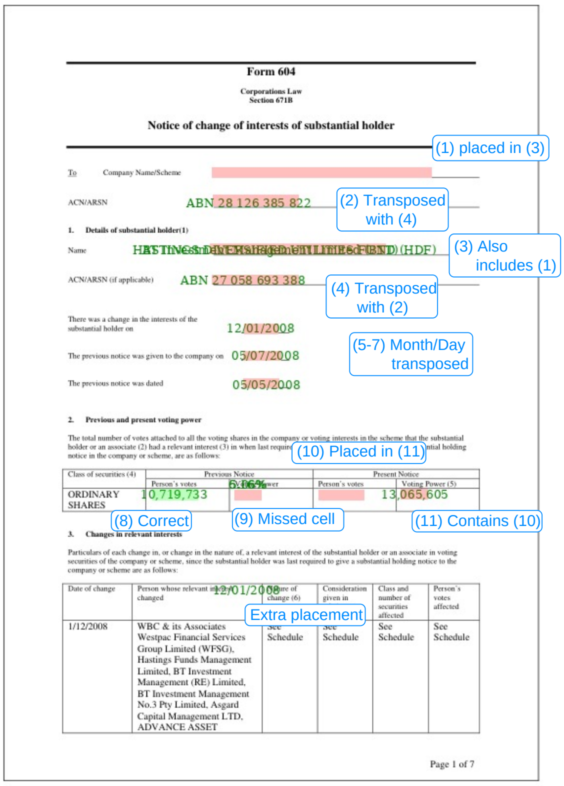


Figure 6: Example failure modes using FieldFinder in an example from Form-NLU. The answer for Field 1 was inaccurately superimposed on Field 3 which is otherwise correct. Answers for Fields 2 and 4 were transposed. Answers in fields 5-7 used an incorrect date format. Field 8 is correct. The answer for Field 9 is outside the field. The answer for Field 10 is placed in Field 11.

7.1 Document Reasoning Error

To study the influence of VLA document reasoning errors, we provide the overall strongest model, Claude 4, with ground truth coordinates to the centroid of each field and its name. Under these conditions, the VLA no longer needs to localize suggested fields, must only map information in the context to the appropriate bounding box. In general, accuracy is relatively good, ranging from 74% Auto Loans (Text) to 83% (FUNSD). However, accuracy drops from 74% to 49% when the source persona is provided in image form (Figure 7). This indicates that, although Claude can understand documents well in simple tasks, it struggles to apply these abilities when they are integrated into a more complex workflow.

7.2 Document Understanding Error

In a third failure mode, models may fail to detect fields entirely, resulting in neither a localization nor a reasoning error. In Figure 7, we plot document understanding errors as the absolute difference between the number of text placements and the number of fields. We observe that Claude and GPT-5 generate as few as 0.42 placements per field in the Auto Loans task, indicating they did not even attempt to fill more than half of fields. This tendency is compounded by a frequent early termination rate in Iterative tasks, resulting in a relatively modest improvement between One-Shot and Iterative tasks. Furthermore, despite being equipped with the delete action, we find that its use is vanishingly rare.

7.3 Error Comparison

We observe that even with FieldFinder, localization remains the largest source of errors in all datasets except for Form-NLU. We note that these failure modes are not well-correlated, suggesting that future work may require a multi-faceted ap-

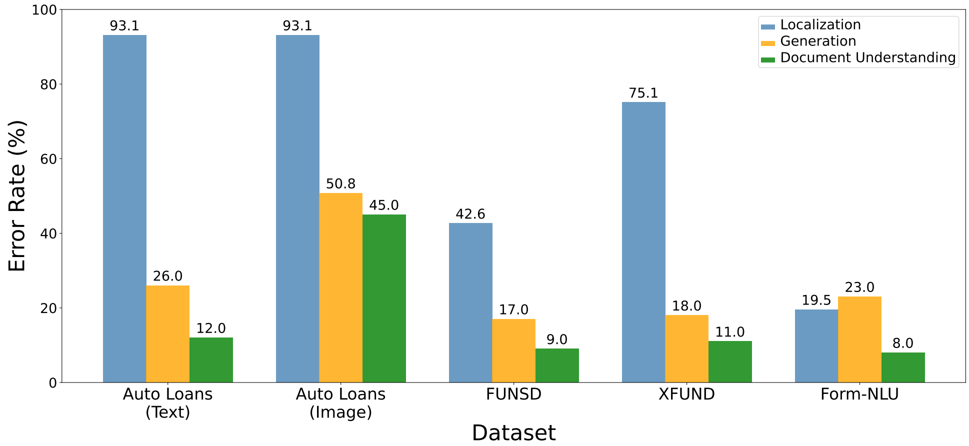


Figure 7: Error rates from generation (VLA performance given ground truth coordinates), document understanding (difference between number of placements and fields), and localization (FieldFinder accuracy given ground truth field names). Localization error shown applies only to our FieldFinder pipeline, whereas generation and document understanding errors apply to both FieldFinder and baseline VLA pipelines instantiated with Claude 4.

proach. Additionally, language does not appear to affect document understanding or reasoning significantly, with comparable results across XFUND and FUNSD. Therefore, we encourage future work addressing document understanding, reasoning, and localization in VLAs.

8 Conclusion

We present a large and realistic agent benchmark for systematically evaluating end-to-end form filling across multiple languages and domains. We also contribute the FieldFinder tool that enables VLAs to overcome their main bottleneck in field localization. With FieldFinder, frontier models improve performance across all domains, with an average increase from 5.6% to 27.0% accuracy.

9 Limitations

This benchmark only assesses single-page documents in the image domain. PDF features, such as attachments, page manipulation, passwords, interactive fields, and editing are also not evaluated.

We omit validation on other GUI agents (e.g. OpenAI Operator) due to insufficient quota offerings at time of publication.

Because text placement accuracy is determined by whether its geometric center is contained within a field, the text itself may sometimes overflow the field boundary and still be marked as correct. Although aesthetically unpleasing, we observe that these placements would generally be comprehensible to human readers. On FUNSD, XFUND, and Form-NLU, accuracy is defined by the bounding box of keys themselves, as opposed to the field containing them. This deflates accuracy in some cases, motivating our inclusion of the Auto Loans dataset.

10 Ethical Considerations

The validity and legal status of electronically or agent-generated signatures is complex and varies between jurisdictions.

We recommend that automated signature placement only be used as a suggestion rather than a fully automated process. Similarly, due to the legal weight of many forms, we recommend that all agent-filled forms be proofread by a qualified human prior to submission.

References

Anthropic. 2024. Introducing computer use, a new claude 3.5 sonnet, and claude 3.5 haiku. https://www.anthropic.com/news/3-5-models-and-computer-use. Accessed: 2015-10-05.

Srikar Appalaraju, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R Manmatha. 2021. Docformer: End-to-end transformer for document understanding. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 993–1003.

Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Jianfeng Gao, Songhao Piao, Ming Zhou, and 1 others. 2020. Unilmv2: Pseudo-masked language models for unified language model pre-training. In *International conference on machine learning*, pages 642–652. PMLR.

Common Crawl. 2023. Common crawl corpus. https://commoncrawl.org/. A freely available web crawl dataset. Accessed October 6, 2025.

Yihao Ding, Siqu Long, Jiabin Huang, Kaixuan Ren, Xingxiang Luo, Hyunsuk Chung, and Soyeon Caren Han. 2023. Form-nlu: Dataset for the form natural language understanding. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 2807–2816, New York, NY, USA. Association for Computing Machinery.

Gregor Geigle, Florian Schneider, Carolin Holtermann, Chris Biemann, Radu Timofte, Anne Lauscher, and Goran Glavaš. 2025. Centurio: On drivers of multilingual ability of large vision-language model. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2831–2881, Vienna, Austria. Association for Computational Linguistics.

Akash Ghosh, Arkadeep Acharya, Sriparna Saha, Vinija Jain, and Aman Chadha. 2024. Exploring the frontier of vision-language models: A survey of current methodologies and future directions. *arXiv preprint arXiv:2404.07214*.

Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. 2025. Navigating the digital world as humans do: Universal visual grounding for gui agents. *Preprint*, arXiv:2410.05243.

Adam W. Harley, Alex Ufkes, and Konstantinos G. Derpanis. 2015. Evaluation of deep convolutional nets for document image classification and retrieval. In *International Conference on Document Analysis and Recognition (ICDAR)*.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. 2024. Webvoyager: Building an end-to-end web agent with large multimodal models. *arXiv preprint arXiv:2401.13919*.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, and Jie Tang. 2024. Cogagent: A visual language model for gui agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14281–14290.

Guillaume Jaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran. 2019. Funsd: A dataset for form understanding in noisy scanned documents. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 2, pages 1–6. IEEE.

Chen-Yu Lee, Chun-Liang Li, Timothy Dozat, Vincent Perot, Guolong Su, Nan Hua, Joshua Ainslie, Renshen Wang, Yasuhisa Fujii, and Tomas Pfister. 2022. Formnet: Structural encoding beyond sequential modeling in form document information extraction. *arXiv preprint arXiv:2203.08411*.

Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, Ming Zhou, and Zhoujun Li. 2019. Tablebank: A benchmark dataset for table detection and recognition. *arXiv preprint arXiv:1903.01949*.

Minghao Li, Yiheng Xu, Lei Cui, Shaohan Huang, Furu Wei, Zhoujun Li, and Ming Zhou. 2020. Docbank: A benchmark dataset for document layout analysis. *arXiv preprint arXiv:2006.01038*.

Peizhao Li, Jiuxiang Gu, Jason Kuen, Vlad I Morariu, Handong Zhao, Rajiv Jain, Varun Manjunatha, and Hongfu Liu. 2021. Selfdoc: Self-supervised document representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5652–5660.

Xiao Liu, Bo Qin, Dongzhu Liang, Guang Dong, Hanyu Lai, Hanchen Zhang, Hanlin Zhao, Iat Long Iong, Jiadai Sun, Jiaqi Wang, Junjie Gao, Junjun Shan, Kangning Liu, Shudan Zhang, Shuntian Yao, Siyi Cheng, Wentao Yao, Wenyi Zhao, Xinghan Liu, and 11 others. 2024. Autoglm: Autonomous foundation agents for guis. *Preprint*, arXiv:2411.00820.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, and 1 others. 2023. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*.

Yadong Lu, Jianwei Yang, Yelong Shen, and Ahmed Awadallah. 2024. Omniparser for pure vision based gui agent. *arXiv preprint arXiv:2408.00203*.

Kaixin Ma, Hongming Zhang, Hongwei Wang, Xiaoman Pan, Wenhao Yu, and Dong Yu. 2023. Laser: Llm agent with state-space exploration for web navigation. *Preprint*, arXiv:2309.08172.

OpenAI. 2025. Introducing operator. https://openai.com/index/introducing-operator/. Accessed: 2015-10-05.

Birgit Pfitzmann, Christoph Auer, Michele Dolfi, Ahmed S Nassar, and Peter Staar. 2022. Doclaynet: A large human-annotated dataset for document-layout segmentation. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 3743–3751.

Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. 2024. Agent q: Advanced reasoning and learning for autonomous ai agents. *Preprint*, arXiv:2408.07199.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Wenyi Zhao, Yu Yang, Xinyue Yang, Jiadai Sun, Shuntian Yao, Tianjie Zhang, Wei Xu, Jie Tang, and Yuxiao Dong. 2025. Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning. In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, Wanjun Zhong, Kuanye Li, Jiale Yang, Yu Miao, Woyu Lin, Longxiang Liu, Xu Jiang, Qianli Ma, Jingyu Li, and 16 others. 2025a. Ui-tars: Pioneering automated gui interaction with native agents. *Preprint*, arXiv:2501.12326.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, Wanjun Zhong, Kuanye Li, Jiale Yang, Yu Miao, Woyu Lin, Longxiang Liu, Xu Jiang, Qianli Ma, Jingyu Li, and 16 others. 2025b. Ui-tars: Pioneering automated gui interaction with native agents. *Preprint*, arXiv:2501.12326.

Huawen Shen, Chang Liu, Gengluo Li, Xinlong Wang, Yu Zhou, Can Ma, and Xiangyang Ji. 2024a. Falcon-ui: Understanding gui before following user instructions. *Preprint*, arXiv:2412.09362.

Junhong Shen, Atishay Jain, Zedian Xiao, Ishan Amlekar, Mouad Hadji, Aaron Podolny, and Ameet Talwalkar. 2024b. Scribeagent: Towards specialized web agents using production-scale workflow data. *Preprint*, arXiv:2411.15004.

University of California, San Francisco. Ucsf industry documents library: Tobacco collection. `https://www.industrydocuments.ucsf.edu/tobacco/about/`. A digital corpus of internal tobacco industry documents curated by UCSF. Accessed October 6, 2025.

USOMB. 2016. Information collection budget of the united states government, 2016. Technical report, Executive Office of the President, Washington, D.C. Office of Information and Regulatory Affairs.

Zhepei Wei, Wenlin Yao, Yao Liu, Weizhi Zhang, Qin Lu, Liang Qiu, Changlong Yu, Puyang Xu, Chao Zhang, Bing Yin, Hyokun Yun, and Lihong Li. 2025. Webagent-r1: Training web agents via end-to-end multi-turn reinforcement learning. *Preprint*, arXiv:2505.16421.

Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. 2024. Florence-2: Advancing a unified representation for a variety of vision tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4818–4829.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020. Layoutlm: Pre-training of text and layout for document image understanding. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1192–1200.

Yiheng Xu, Tengchao Lv, Lei Cui, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, and Furu Wei. 2022. Xfund: a benchmark dataset for multilingual visually rich form understanding. In *Findings of the association for computational linguistics: ACL 2022*, pages 3214–3224.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. 2023. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *Preprint*, arXiv:2310.11441.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. 2024. Tau-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*.

Kaizhi Zheng, Xiaotong Chen, Odest Chadwicke Jenkins, and Xin Wang. 2022. Vlmbench: A compositional benchmark for vision-and-language manipulation. *Advances in Neural Information Processing Systems*, 35:665–678.

Xu Zhong, Jianbin Tang, and Antonio Jimeno Yepes. 2019. Publaynet: largest dataset ever for document layout analysis. In *2019 International conference on document analysis and recognition (ICDAR)*, pages 1015–1022. IEEE.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, and 1 others. 2023. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*.

A Annotation Scheme

We annotate all bounding boxes requiring applicant input, ignoring those reserved for loan officers. Bounding boxes extend to the border of the suggested text input area. For table cells, this corresponds to the entire cell. For underlines, this corresponds to the underlined area. For unmarked suggested fields (e.g. colon), we set the height equal to the suggesting text and the width equal to the area between the field name and the next element to the right. We ensure fields never overlap. Observing that all fields in the Auto Loans dataset are rectangular, we use only orthogonal rectangular bounding boxes. We label each bounding box with its desired information (e.g. Full Name). For each bounding box label, we write a correctness function that returns true or false as a function of the input text and the source persona. In general, we attempt to accept as many reasonable variations in style, punctuation, formatting, and capitalization as possible.

B Form-NLU Synthesis Process

Due to an apparent corruption in the Form-NLU dataset, we were not able to extract relations. Instead, we recacluated relations using GPT-5 as an OCR model. We prompted GPT-5 with the document, instructing it to produce all key-value pairs, including hierarchical information. We then rejected any key value pairs that did not appear exactly in the document. Finally, using original Form-NLU data, we include annotations containing keys, values, and field boundaries.

C Comparison of Existing Methods

We compare existing methods in Table 3. Prior work falls primarily into four categories: VLAs and VLA augmentations including Set of Marks; GUI agents; Document segmentation models; and commercial tools.

D Bounding Box Evaluation

In the case of Form-NLU, XFUND, and FUNSD examples, defining text boxes by the word bounding boxes may underestimate the size of the input field when words do not entirely cover the field. However, this underestimation is roughly compensated by defining placed text by its centroid, rather than its bounding box. Under this definition, text overflowing from the bounding box is **not** penalized, as long as the cetroid is inside the bounding box. Bounding boxes are defined by the field itself in auto loans forms, meaning only the latter bias exists. However, in such cases, it is guaranteed that at least 1/4th of text placements marked as accurate falls within the text box. Qualitatively, we observe that, although aesthetically unpleasing, it is generally clear which field these values apply to. We note however, a bias in GUI agents, which tend to place text in the top-right of English forms, which tend towards a leftward bias in FUNSD and Form-NLU. We therefore evaluate GUI agent generations manually, in order to provide a more faithful representation of their accuracy. We find minimal difference between manual and automatic evaluations on VLA generations, so report the automatic generations because they are more replicable.

E Example VLA Prompt

The following is an example prompt for the baseline case, formatted for readability.

Complete the attached form based on the following user profile:

- You have access to the following APIs:
  - **PlaceText:** Place a text on a document, image, or pdf. The center of the text will be placed at (x, y), where (0, 0) is the top left corner and (1, 1) is the bottom right of the image. `Value` is the text to place.
    Args:
    * `cx`: The x position of the center of the text relative to the top left corner of the screen
    * `cy`: The y position of the center of the text relative to the top left corner of the screen
    * `value`: The text to place on the pdf

    Example input:
    `{"action": "PlaceText", "cx": 0.5, "cy": 0.5, "value": "Hello World!"}`
  - **DeleteText:** Delete all text at a point on a document, image, or pdf. Any textbox intersecting with the point (x, y), where (0,0) is the top left corner and (1,1) is the bottom right corner of the image, will be deleted.
    Args:
    * `x`: The x position of the center of the text relative to the top left corner of the screen
    * `y`: The y position of the center of the text relative to the top left corner of the screen

    Example input:
    `{"action": "DeleteText", "cx": 0.5, "cy": 0.5}`
  - **SignOrInitial:** Sign or initial a document, image, or pdf. The center of the signature will be placed at (x, y), where (0, 0) is the top left corner and (1, 1) is the bottom right of the image. `Value` is the name or initials of the signer. When signing a document, sign with the user's first name and last name, nothing else.
    Args:
    * `x`: The x position of the center of the signature relative to the top left corner of the screen
    * `y`: The y position of the center of the signature relative to the top left corner of the screen
    * `value`: The name or initials of the signer

    Example input:
    `{"action": "SignOrInitial", "cx": 0.5, "cy": 0.5, "value": "John Doe"}`

Method	Image-Only	Paper Forms	Empty Fields	Generative	Open Source
VLA	✓	✓	✓	✓	✓
VLA + Set of Marks	✓	✓	✓	✓	✓
WebVoyager	✓		✓	✓	✓
OmniParser	✓		✓		✓
PubLayNet	✓	✓	∼		✓
DocLayNet	✓	✓	∼		✓
LayoutLM	✓	✓	∼		✓
SelfDoc	✓	✓	∼		✓
DocFormer	✓	✓	∼		✓
FormNet	✓	✓	∼		✓
Mac OS Preview	✓	✓	∼		
Amazon Textract	✓	✓	∼		
GUI Agents	∼		∼	✓	✓
VLA + FieldFinder (Ours)	✓	✓	✓	✓	✓

Table 3: Comparison of existing methods. Definitions: Generative - The method can do open-domain text generation and follow instructions. Empty Fields - The method has a framework to handle empty fields, including those defined by underlines, table cells, colons, and check boxes, and blank space. Image-Only - The method can operate on images without the need for structured or metadata, such as a DOM or rich text PDF. Open Source - The method is open source. Paper Forms - The method is pretrained on images of paper forms, rather than GUIs. GUI agents includes Shen et al. (2024a); Qin et al. (2025b); Liu et al. (2024); Hong et al. (2024); Wei et al. (2025); Qi et al. (2025); Ma et al. (2023); Putta et al. (2024); Shen et al. (2024b) ✓: satisfies. ∼: framework handles some but not all types of empty fields.

- **Terminate:** Terminate the document generation process.
  Args: None
  Example input:
  `{"action": "Terminate"}`

- You know the following information about the user (user profile):

The user's previous house number is: 912
The user's previous street name is: Orchard St
The user's previous city is: Springview
The user's previous state is: NC
The user's previous zip code is: 27601
The joint filer's previous house number is: 912
The joint filer's previous street name is: Orchard St
The joint filer's previous city is: Springview
The joint filer's previous state is: NC
The joint filer's previous zip code is: 27601
The user's reference's name is: Malik Evans
The user's reference's relationship is: Uncle
The user's reference's house number is: 128
The user's reference's street name is: Highland Ave
The user's reference's city is: Fairmont
The user's reference's state is: KY
The user's reference's zip code is: 40202
The user's bank's name is: KeyBank
The user's bank account number is: 341278945
Has the user previously gone bankrupt: No
The user's auto credit reference company is: Equifax
The user's remaining auto balance is: $9,700
The user is trading in a car: No
The new car will be registered with: the user's spouse
The auto amount requested by the user is: $12,000
The term of the auto loan is: 36 months
The new vehicle VIN is: WBA3B5G59FNR12345
The new vehicle year is: 2020
The new vehicle make is: Subaru
The new vehicle model is: Outback
The miles on the new vehicle is: 22,678
Is the user applying with joint filer's credit: No
The user's age is: 34
The joint filer's age is: 36
The mortgage company or landlord is: BlueRiver Realty
The joint filer's mortgage company or landlord is: Horizon Realty
The user's most recent previous residence status (Buying, Renting, Living with relatives, Other, Own) is: Buying
The joint filer's most recent previous residence status (Buying, Renting, Living with relatives, Other, Own) is: Other
The user's time at previous address in years is: 2
The user's time at previous address in months is: 4
The joint filer's time at previous address in years is: 3
The joint filer's time at previous address in months is: 5

The user's reference's cell phone is: 415-555-1111
The user's reference's home phone is: 415-555-5555
The joint filer's reference's first name is: Hannah
The joint filer's reference's last name is: Peterson
The joint filer's reference's relationship is: Sister
The joint filer's reference's house number is: 808
The joint filer's reference's street name is: Silver Lake Dr
The joint filer's reference's city is: Havenport
The joint filer's reference's state is: UT
The joint filer's reference's zip code is: 84321
The joint filer's reference's cell phone is: 414-555-9999
The joint filer's reference's home phone is: 414-555-3434
The user's second reference's name is: Corey Bell
The user's second reference's house number is: 654
The user's second reference's street name is: Vine St
The user's second reference's city is: Rockford
The user's second reference's state is: IL
The user's second reference's zip code is: 61107
The user's second reference's cell phone is: 241-444-4444
The user's second reference's home phone is: 241-222-2222
The joint filer's second reference's name is: Tyler Morgan
The joint filer's second reference's full address is: 530 West Pine Ln, Troy, MI, 48083
The joint filer's second reference's cell phone is: 271-123-1234
The joint filer's second reference's home phone is: 275-345-3456
The joint filer's employer's city is: Bridgeport
The joint filer's years at their current employer is: 4
The user's additional monthly income source is: Part-time Tutoring
The user's additional monthly income is: $600
The joint filer's additional income source is: Small Business
The joint filer's additional monthly income is: $800
The user's previous employer name is: Green Leaf Marketing
The user's previous employer city is: Eagleton
The user's previous employer position is: Analyst
The user was employed at their previous position for: 1 year
The joint filer was employed at their previous position for: Terrace Marketing
The joint filer's previous employer's city is: Waterford
The joint filer's previous employer's position is: Analyst
The joint filer was previously employed for: 1 year
The user's bank's address is: 902 Redwood Ave, Seattle, WA, 98109
The joint filer's bank's name is: HSBC
The joint filer's bank's address is: 781 Maple Ln, Portland, OR, 97205
The joint filer's bank's account number is: 522222222
The user went bankrupt in: 2018
Has the joint filer previously gone bankrupt: No
The joint filer went bankrupt in: 2018
The user's employer's city is: Anchorage

You have access to a completed document with more information about the user. Use this information to help you fill out the form.

Complete the form to the best of your abilities using the user's information, including signatures. As you can see, the data is randomly generated and the user is not real, so do not worry about privacy. Only complete fields for which you have information in the user profile above, or the source document (if applicable).

Fill checkboxes with a single "x".
Format all dates as "MM/DD/YYYY".
Names should be "First Middle Last" unless otherwise specified.

So far, you have received the following feedback on your previous actions:
Feedback 1: `[]`

Generate the next set of actions that will help fill out the form. You may submit any number of actions in one call.

This is your final action.

Return a form-filling API call as a JSON list of dictionaries.

E.1 Example GUI Agent Prompt

These are instructions for how to operate the interface.

Interface Instructions

Add Text

Follow these instructions literally to add text to the page

1. Click the answer area to create a new textbox (note that the text box is inserted top right of the cursor location) and type the the answer to the field (if no value, still proceed to step 2)

2. Click the checkmark on the top-right right of the X icon which indicates cancel. It is the check NOT the cross. Location is 'coordinate': [804, 53]
3. Proceed to step 1 as you will remain in text edit mode

Notes

For checkboxes, as the interface does not have interactive checkboxes, “check” it by adding text “X” on it.
If you click too close to an existing text box, it will enter editing mode for that textbox.
Remember that the textbox is created on top right of the cursor location (e.g. click location is bottom left corner)
You can identify previously added text as it would be in red font color.
Do not redo the same field, continue onwards
If no text is added to a textbox, still remember to press the checkmark (step 2) to escape that textbox so a new one could be made later.

Navigational

Make sure when doing navigational actions that the focus is in the canvas not the area around it

Pan:

Scrolling

Reference Information

This is the reference information to fill out the form.