FormGym: Doing Paperwork with Agents

Anonymous ACL submission

Abstract

Completing paperwork is a challenging and time-consuming problem. Form filling is especially challenging in the pure-image domain without access to OCR, typeset PDF text, or a DOM. For computer agents, it requires multiple abilities, including multi-modal understanding, information retrieval, and tool-use. We present a novel form-filling benchmark consisting of 432 fields spread across 55 documents and 3 tasks, requiring knowledge of 236 features per user. We find that baseline VLAs achieve less than 1% accuracy in most cases, primarily due to poor localization ability. GUI agents also struggle, scoring between 10.6-68.0% despite high cost and latency. Therefore, we also contribute FieldFinder, a tool to assist LLMs in identifying where to place text on a form. With FieldFinder, all models achieve equal or better performance in all six study conditions, with a maximum increase from 2% to 56%.

1 Introduction

Filling out paperwork is a pervasive and tedious task. Although some paper forms have been replaced by fillable rich-text PDFs, many are only available as pure images either in their original format or as scanned physical documents. These forms represent the most challenging task because agents can only interact with the document as an image rather than the information-rich DOM or PDF typeset text and vector graphics. This task builds on prior work on document understanding, OCR, localization, and agentic workflows to evaluate end-to-end image manipulation accuracy.

In this work, we propose a new benchmark for evaluating the ability of general-purpose vision-language agents (VLAs) to perform end-to-end form completion. Our evaluation focuses on realistic use cases where an agent must interpret a document and populate fields based on a user profile. Relevant user information is provided as raw text, a SQL database, or other completed forms containing partially overlapping responses. Across four tasks involving these inputs, we find that current baseline VLAs score under 3% accuracy in all but one case. GUI agents also struggle with this task, completing at most 3.9% of fields in the hardest Doc Transfer task. Among the steps involves in form-filling, we find that VLAs primarily struggle with text placement. GUI agents struggled with text placement, mulit-step actions, and completion within the allotted time frame.

To address the localization bottleneck, we introduce a modular architecture that separates semantic understanding from spatial grounding. Specifically, we equip any VLA with the ability to name the field it intends to complete, e.g., "Date of Birth", and delegate the task of locating the corresponding input area to an auxiliary VLM FieldFinder tool. FieldFinder predicts the bounding box of the target field's input region (e.g., an empty line, cell, check box, or empty space next to the target text). VLAs, when equipped with FieldFinder, improve accuracy by as much as 54 percentage points.

Our contributions are as follows:

A benchmark for evaluating agents on realistic form completion scenarios, showing that current VLAs struggle to accurately identify field placements.

An open-vocabulary field detection model, showing that it helps VLAs overcome spatial reasoning limitations.

We intend to release both publicly on GitHub.

2 Related Work

Several benchmarks exist for evaluating document layout understanding (Zhong et al., 2019; Pfitzmann et al., 2022; Li et al., 2020, 2019; Harley et al., 2015; Li et al., 2019). Numerous vision-language (Xu et al., 2020; Li et al., 2021; Bao et al., 2020; Appalaraju et al., 2021; Lee et al., 2022)

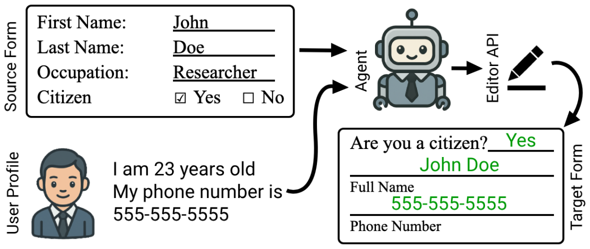


Figure 1: The FormGym task. Agents are provided with a user profile in natural language and, optionally, a source form. The agent must use an editor API to complete the target form.

have been proposed for these types of tasks. Unlike traditional QA-style benchmarks, VLA 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. Existing software, such as Mac OS Preview and Amazon Textract, can localize text fields in PDFs. However, they sometimes fail to identify non-underlined fields, including table cells or those indicated merely by a colon (e.g., "Name: "). In contrast, our work builds on these domains to explore end-to-end, real-world form completion.

3 FormGym: Realistic Form-Filling for Agents

We aim to evaluate whether VLAs can produce completely filled forms when given access to user data and image editing tools. FormGym includes a diverse set of forms, user profiles, and agent actions representing a range of realistic challenges.

3.1 Documents

Our task consists of four document tasks. The **Auto Loan Task - Text** task consists of four densely annotated American vehicle loan application forms containing a total of 357 input fields. To enable evaluation on multiple user profiles (see below), we annotate each field with the type of user information (e.g., full name) it should contain rather than a specific answer (e.g., John Doe). For each form, we provide four user profiles. User profiles contain atomic facts, such as first name and postal code. As a result, many fields, such as address or middle initial, do not map directly to user profile information and instead must be derived from one or more user profile facts. In the case of the **Auto Loans - Doc Transfer** task, we provide the facts in the form of another Auto Loans source document, densely completed with user information. Information not available in the source document is provided in natural language.

The **Database** Task consists of 49 fields on two commercial banking forms. We provide the content of 39 of these fields in a SQL database that agents must query. Several of these fields are not provided in the SQL database so must be calculated arithmetically from values in other fields according to instructions on the form.

Finally, we contribute the **FUNSD Task** for evaluating diverse formats and multilingual reasoning, derived from Jaume et al. (2019)'s document relation dataset. The FUNSD Task consists of 50 examples from the FUNSD test set with exactly one target answer field masked in each document.

3.2 Actions

To edit forms, we provide agents with the following actions:

- `PlaceText(x, y, value)` Place the text `value` centered at the coordinates (x, y).
- `DeleteText(x, y)` Delete all input text whose bounding boxes contain the coordinate (x, y).
- `SignOrInitial(x, y, value)` Place the `value` at coordinate (x, y) in the form of a signature or initials.
- `QuerySql(query)` Query the SQL database in the Database Task using `query`.
- `Terminate()` End the current session.

3.3 Flows

We evaluate agents under two workflows:
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 10 rounds, allowing it to correct mistakes. We report additional details in Appendix A.2.

3.4 Evaluation

Each field is also associated with a correctness function to provide fair evaluation of answers with multiple correct formats, such as telephone numbers. If a field contains multiple text inputs, we concatenate them. 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 is within a designated bounding box.

3.5 Baseline Agents

We experiment with both classic VLAs and GUI agents capable of interacting with browser and desktop applications.

3.5.1 Vision Language Models

We prompt VLAs with API documentation, examples of all available actions, and a natural language descriptions of the user profile (Appendix A.4).

3.5.2 GUI Agents

We instantiate GUI agents Claude Computer Use and OpenAI Operator with the free in-browser photo editing application Photopea[1], whose interface is nearly identical to Photoshop (Appendix A.3). We prompt GUI agents with natural language user profile descriptions and instructions to complete the form. For accessibility and cost reasons, we limit operators to five minutes per page. Prompts include detailed instructions on how to use the Photopea interface, without which GUI agents fail completely (Appendix A.5).

4 FieldFinder

We observe that large baseline VLAs make coherent API calls, but universally struggle to place text in appropriate locations. To ameliorate this issue, we create the FieldFinder tool. FieldFinder takes 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 2).

4.1 Dataset

To train the FieldFinder tool, we create a (Document, target field name, bounding box) dataset using question/answer relations in the FUNSD and multilingual XFUND (Xu et al., 2022) form understanding datasets. Since FUNSD and XFUND forms contain responses in answer fields, we use horizontal inward content aware fill [2] to automatically remove text while generally preserving formatting such as lines and table boundaries.

4.2 Training

We fine-tune a Florence 2 Large (Xiao et al., 2024) vision foundation model to predict the answer bounding box coordinates given the target question text and document. 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 4 epochs using early stopping, batch size 8, learning rate 1e-6 on 1x NVIDIA A100 GPU for approximately 20 hours. The FieldFinder achieves an intersect-over-union of 20.9% on the FUNSD test set.

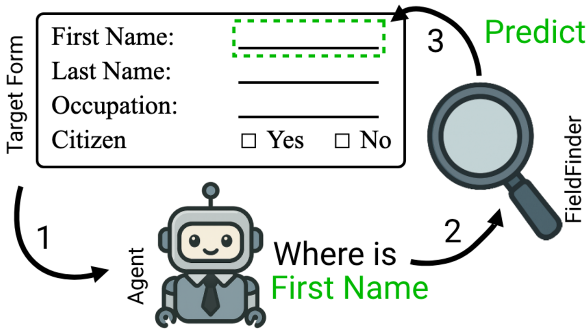


Figure 2: Agent use of the FieldFinder tool. 1) The agent ingests an input form or database. 2) The agent requests the location of an empty field by name. 3) The FieldFinder returns the bounding box around the target field to the agent.

Table 1: Generated by Spread-LaTeX

	Forms	NL Fields	DB Fields	Users
Auto Loans	4	357	0	4
Database	2	10	39	1
FUNSD	50	50	0	1

Table 2: Total form pages, fields whose values are supplied in natural language, supplied in a dataabase, and user profiles in FormGym tasks.

5 Results

Overall, VLAs struggle with this task, with models performing best on FUNSD and worst on Database (Table 3). Baseline models generally score $\leq 1\%$, except for Claude on FUNSD and Database (32% and 2.7%, respectively). When introducing FieldFinder, we observe equal or better performance in all cases. In the best case, GPT-4o's performance on FUNSD increases from 2% to 56%. We observe smaller gains, up to 16.9 percentage points on Auto Loans (GPT-4o), and 29.3 points on Database (Claude 3.7). Certain small, open-source models

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

	Auto Loans (Text)		Auto Loans (Doc Transfer)		Database	FUNSD
	One-shot	**Iterative**	**One-shot**	**Iterative**	**Iterative**	**One-shot**
Aria 25B	0.0	0.1	0.0	0.0	1.0	0.0
Claude 3.7	0.1	0.3	0.2	0.2	2.7	32.0
GPT-4o	0.6	0.6	0.0	0.6	0.0	2.0
Llava 7B	0.0	0.0	0.0	0.0	0.0	0.0
Molmo 7B	0.0	0.0	-	-	0.0	0.0
Aria 25B + FL (ours)	6.2	6.7	1.4	2.4	1.0	28.0
Claude 3.7 + FL (ours)	**18.8**	14.9	5.8	**7.2**	**32.0**	52.0
GPT-4o + FL (ours)	12.2	**17.2**	**7.1**	6.4	0.0	**56.0**
Llava 7B + FL (ours)	1.5	0.4	0.4	0.0	1.0	6.0
Molmo 7B + FL (ours)	0.4	0.0	-	-	20.0	20.0
OpenAI Operator	-	**18.3**	3.9	-	36.0	50.0
Claude Computer Use	-	10.6	1.4	-	**44.0**	**68.0**

Table 3: Average form completion percentage (correct fields / all fields). Iterative FUNSD is omitted because FUNSD forms contain only one empty field. One-shot Database is omitted because at least two turns are necessary. Molmo is not trained for multi-image prompting

including Aria 25B and Molmo 7B achieve significant performance improvements with FieldLocalizer. GPT-4o and Claude also struggle to chain actions in the more complex Doc Transfer and Database tasks. GPT-4o performs especially poorly, suggesting the user query the database herself, then signing a page footer with "Your Name".

Across all tests, GUI agents performed as comparable or better than VLAs, except in Doc Transfer. Although GUI agents still made localization errors, these were typically less distant than those of VLAs. GUI agents often did not complete the Auto Loans and Database tasks within the 5 minute timeframe, negatively impacting completion. Although Claude Computer Use was more accurate than OpenAI Operator, it performed actions about half as fast, bottlenecking completion.

6 Discussion

We attribute weak baseline model performance to several failure modes. The inability to localize answer fields and chain actions are the primary weaknesses in Claude and GPT-4o. Although Auto Loans contains 357 graded fields, Claude and GPT-4o make as few as 71 placement attempts in some cases, suggesting a failure in document understanding and completeness tracking. Claude and GPT-4o also struggle to recover from mistakes. Although they are provided with an API to delete text, its usage is vanishingly rare.

When using FieldFinder, accuracy on FUNSD is uniformly higher than on other tasks. We attribute the performance discrepancy to several factors. First, FieldFinder was trained on FUNSD, so testing on Database and Auto Loans represents a significant distribution shift in inputs. Second, Auto Loans requires differentiating between relationally complex fields, such as "applicant first reference name" versus "co-applicant second reference name", indicated by physically distant table headers. To model the upper limit of the impact of FieldFinder, we conduct an ablation study wherein models are prompted with the exact centroid coordinates of fields. Under these conditions, GPT-4o achieves 77% accuracy and Claude 3.7 achieves 82%, suggesting field localization errors account for about 4/5 errors, while document understanding accounts for the other 1/5. Future work should explore training field localizers on a broader distribution of documents and improving foundational models' visual reasoning and backtracking abilities.

Given GUI agents accurate but sluggish performance, future research should prioritize inference speed and UI generalization with a minor focus on localization. Poor inference efficiency also raises costs, which we calculate to be approximately $1 USD per Auto Loans page. We note that without iterative and specific prompt engineering, GUI models perform no successful actions.

7 Conclusion

We present a challenging agent benchmark for image-domain form filling and contribute a cross-model field localization tool that can retrofit VLAs, increasing form completion by up to 54 percentage points with minimal overhead.

8 Limitations

For cost and accessibility reasons, this benchmark only assesses performance on a small sample of commercial, English, single-page documents in the image domain. PDF features, such as attachments, page manipulation, passwords, interactive fields, and editing are also not evaluated.

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.

9 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

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.

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*.

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, 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*.

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.

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.

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*.

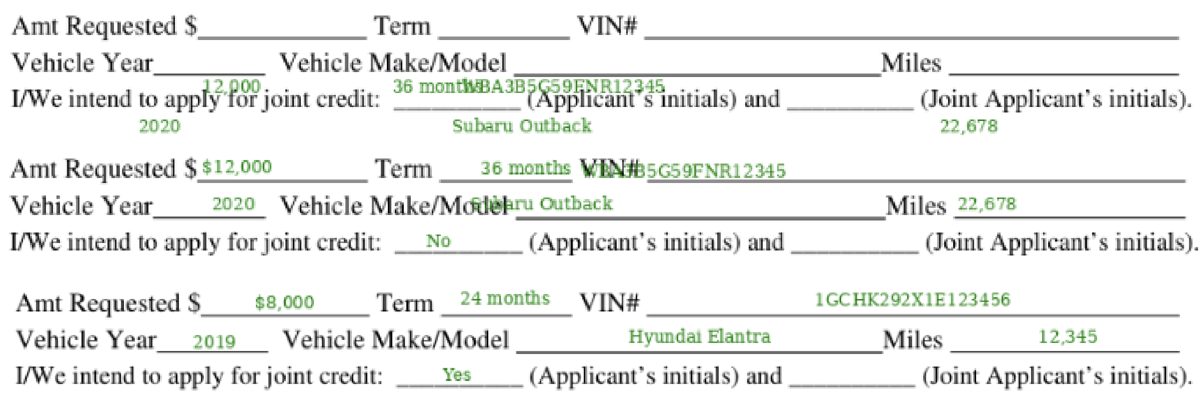
Kaizhi Zheng, Xiaotong Chen, Odest Chadwicke Jenkins, and Xin Wang. 2022. Vlmbench: A compositional benchmark for vision-and-language manipulation. *Advances in Neural Information Processing Systems*, 35:665–678.

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 Appendix

A.1 Example Output


Amt Requested $__________ Term ________ VIN# ____________________
Vehicle Year_______ Vehicle Make/Model ________________Miles __________
I/We intend to apply for joint credit: ________ (Applicant's initials) and ________ (Joint Applicant's initials).

Amt Requested $ $12,000 Term 36 months VIN# WBA3B5G59FNR12345
Vehicle Year 2020 Vehicle Make/Model Subaru Outback Miles 22,678
I/We intend to apply for joint credit: No (Applicant's initials) and ________ (Joint Applicant's initials).

Amt Requested $ $8,000 Term 24 months VIN# 1GCHK292X1E123456
Vehicle Year 2019 Vehicle Make/Model Hyundai Elantra Miles 12,345
I/We intend to apply for joint credit: Yes (Applicant's initials) and ________ (Joint Applicant's initials).

Figure 3: Output by Claude 3.7 in the Auto Loans One-shot task. Baseline (top), with FormFiller (middle), with ground truth field centroids in prompt (bottom).

A.2 Additional Experimental Details

In iterative flow, after each agent turn, the agent is presented with an updated document reflecting any text or signatures it placed. On subsequent turns, we provide the agent with the following feedback for each action in the prompt:

`PlaceText` - Whether the text was placed successfully and where

`DeleteText` - What text was deleted, if any

`SignOrInitial` - Whether the signature was placed successfully and where

`QuerySql` - The SQL output or error message

Intersection over Union (IoU) is calculated as:

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}} = \frac{|A \cap B|}{|A \cup B|}$$

Where:
A = predicted bounding box
B = ground truth bounding box
$|A \cap B|$ = area of intersection
$|A \cup B|$ = area of union

In our FUNSD dataset, the ground truth bounding box is taken to be the envelope of the answer string, which is generally a subset of the actual field. This may contribute to an underestimate of actual IoU accuracy and FUNSD placement accuracy. However, because we care predominately about the centroid of the placement text, training and predicting a smaller bounding box contained within the actual field should not negatively impact training.

A.3 GUI Agent Implementation Details

We the performance between Anthropic Claude Computer Use, and OpenAI Operator. To equip models with the necessary tools, we set up an environment to allow models to place text on the a PNG version of the form through the online graphic editor Photopea. This enables agents (notably OpenAI Operator) to operate entirely in a web browser environments. Instructions on how to use the system were provided specifically to isolate out the form filling performance and remove confounding factors with interface-use performance. We also gave specific interface instructions to prevent models from leaving the tab or deleting the form.

In the Doc Transfer task, a reference document was loaded in another tab inside Photopea. Instructions in the prompt were adjusted to account for the reference document.

We provided GUI agents a REPL connect to the database in Google Colab with ipywidgets for in-browser querying.

A.4 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"}
  ```

- **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.

A.5 Example GUI 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.

A.6 Additional Results

	Auto Loans (Text)		Auto Loans (Doc Transfer)		Database	FUNSD
	One-shot	**Iterative**	**One-shot**	**Iterative**	**Iterative**	**One-shot**
Aria 25B	0.0	0.2	0.0	0.0	0.1	0.0
Claude 3.7	0.2	0.3	0.4	0.7	5.6	34.0
GPT-4o	0.7	0.9	0.0	1.3	0.0	0.8
Llava 7B	0.0	0.0	0.0	0.0	0.0	0.0
Molmo 7B	0.0	0.0	-	-	0.0	0.0
Aria 25B + FL (ours)	14.6	5.4	4.0	2.3	0.3	3.2
Claude 3.7 + FL (ours)	**23.8**	19.3	18.9	**27.0**	**47.8**	**51.0**
GPT-4o + FL (ours)	21.6	**22.1**	**23.2**	15.7	0.0	30.4
Llava 7B + FL (ours)	19.3	5.8	2.9	0.0	4.8	1.3
Molmo 7B + FL (ours)	3.9	0.0	-	-	10.4	1.8
OpenAI Operator	-	59.4	-	-	81.8	50.0
Claude Computer Use	-	86.8	-	-	100.0	68.0

Table 4: Placement accuracy (correct placements / total placements)

	Auto Loans (Text)		Auto Loans (Doc Transfer)		Database	FUNSD
	One-shot	**Iterative**	**One-shot**	**Iterative**	**Iterative**	**One-shot**
Aria 25B	0.0	0.5	0.0	0	0.3	0.0
Claude 3.7	0.5	1.0	0.8	0.75	0.7	16.0
GPT-4o	2.0	2.0	0.0	2	0.0	1.0
Llava 7B	0.0	0.0	0.0	0	0.0	0.0
Molmo 7B	0.0	0.0	-	-	0.0	0.0
Aria 25B + FL (ours)	22.0	23.8	5	8.5	0.3	14.0
Claude 3.7 + FL (ours)	67.0	53.3	20.75	25.75	8.0	26.0
GPT-4o + FL (ours)	43.5	61.3	25.3	23	0.0	28.0
Llava 7B + FL (ours)	5.5	1.3	1.3	0	0.3	3.0
Molmo 7B + FL (ours)	1.3	0.0	-	-	5.0	10.0
OpenAI Operator		65.3		14	9	25
Claude Computer Use		37.75		5	11	34

Table 5: Average Total Correct Fields

	Auto Loans (Text)		Auto Loans (Doc Transfer)		Database	FUNSD
	One-shot	**Iterative**	**One-shot**	**Iterative**	**Iterative**	**One-shot**
Aria 25B	150.0	282.5	85.8	205.25	177.5	649.0
Claude 3.7	300.0	303.3	166.8	102.25	12.0	47.0
GPT-4o	274.0	228.0	71.0	156	3.0	127.0
Llava 7B	27.8	13.0	14.8	22.5	7.8	217.0
Molmo 7B	6.5	31.0	-	-	75.3	472.0
Aria 25B + FL (ours)	151.0	440.8	125.5	365	98.5	443.0
Claude 3.7 + FL (ours)	281.3	276.3	109.5	95.5	16.8	51.0
GPT-4o + FL (ours)	201.5	277.0	109.0	146.75	3.0	92.0
Llava 7B + FL (ours)	28.5	23.0	43.0	12.25	5.3	234.0
Molmo 7B + FL (ours)	32.0	2.0	-	-	48.0	552.0
OpenAI Operator	-	110	-	25.5	11	50
Claude Computer Use	-	43.5	-	16	11	50

Table 6: Average Total Incorrect Placements