

VASPilot: MCP-facilitated multi-agent intelligence for autonomous VASP simulations

Jiaxuan Liu(刘家轩), Tiannian Zhu(朱天念), Caiyuan Ye(叶财渊), Zhong Fang(方忠), Hongming Weng(翁红明), and Quansheng Wu(吴泉生)

Citation: Chin. Phys. B, 2025, 34 (**11**): 117106. DOI: 10.1088/1674-1056/ae0681
http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

First-principles calculations on strain tunable hyperfine Stark shift of shallow donors in Si

Zi-Kai Zhou(周子凯) and Jun Kang(康俊)
Chin. Phys. B, 2025, 34 (**8**): 087102. DOI: 10.1088/1674-1056/addeba

Ab initio prediction of ground-state magnetic ordering and high-pressure magnetic phase transition of uranium mononitride

Jing-Jing Zheng(郑晶晶), Yuxi Chen(陈禹西), Chengxiang Zhao(赵承祥), Junfeng Zhang(张均锋), Ping Zhang(张平), Bao-Tian Wang(王保田), and Jiang-Jiang Ma(马江将)
Chin. Phys. B, 2025, 34 (**8**): 087101. DOI: 10.1088/1674-1056/add4e6

Anomalous ultrafast thermalization of photoexcited carriers in two-dimensional materials induced by orbital coupling

Zhuoqun Wen(文卓群), Haiyu Zhu(诸海渝), Wen-Hao Liu(刘文浩), Zhi Wang(王峙), Wen Xiong(熊稳), and Xingzhan Wei(魏兴战)
Chin. Phys. B, 2025, 34 (**7**): 077103. DOI: 10.1088/1674-1056/adc191

Interacting Dirac semi-metal state in nonsymmorphic Kondo-lattice compound $CeAgSb_2$

Da-Liang Guo(郭达良), and Huan Li(黎欢)
Chin. Phys. B, 2025, 34 (**6**): 067102. DOI: 10.1088/1674-1056/adc6f6

Growth, characterization, and Raman spectra of the 1*T* phases of $TiTe_2$, $TiSe_2$, and TiS_2

Xiao-Fang Tang(唐筱芳), Shuang-Xing Zhu(朱双兴), Hao Liu(刘豪), Chen Zhang(章晨), Qi-Yi Wu(吴旗仪), Zi-Teng Liu(刘子腾), Jiao-Jiao Song(宋姣姣), Xiao Guo(郭晓), Yong-Song Wang(王永松), He Ma(马赫), Yin-Zou Zhao(赵尹陬), Fan-Ying Wu(邬钒颖), Shu-Yu Liu(刘姝妤), Kai-Hui Liu(刘开辉), Ya-Hua Yuan(袁亚华), Han Huang(黄寒), Jun He(何军), Wen Xu(徐文), Hai-Yun Liu(刘海云), Yu-Xia Duan(段玉霞), and Jian-Qiao Meng(孟建桥)
Chin. Phys. B, 2022, 31 (**3**): 037103. DOI: 10.1088/1674-1056/ac306a

---

COMPUTATIONAL PROGRAMS FOR PHYSICS

VASPilot: MCP-facilitated multi-agent intelligence for autonomous VASP simulations

Jiaxuan Liu(刘家轩)[1,2], Tiannian Zhu(朱天念)[1,2], Caiyuan Ye(叶财渊)[1,2], Zhong Fang(方忠)[1,2], Hongming Weng(翁红明)[1,2,†], and Quansheng Wu(吴泉生)[1,2,‡]

[1] *Beijing National Laboratory for Condensed Matter Physics and Institute of Physics, Chinese Academy of Sciences, Beijing* 100190, *China*
[2] *University of Chinese Academy of Sciences, Beijing* 100049, *China*

(Received 25 August 2025; revised manuscript received 9 September 2025; accepted manuscript online 15 September 2025)

Density-functional-theory (DFT) simulations with the Vienna *Ab initio* Simulation Package (VASP) are indispensable in computational materials science but often require extensive manual setup, monitoring, and postprocessing. Here, we introduce VASPilot, an open-source platform that fully automates VASP workflows via a multi-agent architecture built on the CrewAI framework and a standardized model context protocol (MCP). VASPilot's agent suite handles every stage of a VASP study from retrieving crystal structures and generating input files to submitting Slurm jobs, parsing error messages, and dynamically adjusting parameters for seamless restarts. A lightweight Quart-based web interface provides intuitive task submission, real-time progress tracking, and drill-down access to execution logs, structure visualizations, and plots. We validated VASPilot on both routine and advanced benchmarks: automated band-structure and density-of-states calculations (including on-the-fly symmetry corrections), plane-wave cutoff convergence tests, lattice-constant optimizations with various van der Waals corrections, and cross-material band-gap comparisons for transition-metal dichalcogenides. In all cases, VASPilot completed the missions reliably and without manual intervention. Moreover, its modular design allows easy extension to other DFT codes simply by deploying the appropriate MCP server. By offloading technical overhead, VASPilot enables researchers to focus on scientific discovery and accelerates high-throughput computational materials research.

Keywords: VASP, agent, model context protocol (MCP), VASPilot

PACS: 71.15.Mb

DOI: 10.1088/1674-1056/ae0681 **CSTR:** 32038.14.CPB.ae0681

1. Introduction

With the advancement of computer hardware and algorithms, density functional theory (DFT) calculations have become one of the most important research tools in materials science, chemistry, and condensed matter physics. Starting from quantum mechanics, DFT can predict the mechanical, electromagnetic, and optical properties of materials, playing a crucial role in areas such as batteries,[1–3] topological phenomena[4–6] and transport.[7–9]

Researchers have developed numerous DFT software packages based on different basis sets and treatments of core electrons. For example, Vienna *Ab initio* Simulation Package (VASP),[10] Quantum ESPRESSO,[11,12] and CASTEP[13] employ plane-wave basis sets, while OpenMX[14,15] and SIESTA[16] are based on atomic orbital basis sets. Additionally, there are codes like WIEN2k[17] and Elk[18] that utilize all-electron wavefunctions in calculations. Although these softwares are widely used in various research fields, researchers still need to invest considerable effort in the complicated workflow of these software.

VASPKIT,[19] ASE,[20] and Pymatgen[21] have alleviated this pain point to a certain extent. VASPKIT is a command-line-based software that can quickly generate corresponding input files for calculations based on crystal structures upon user's command. Moreover, it facilitates rapid plotting using predefined templates after the computations are completed. ASE is a Python-based molecular dynamics simulation software that integrates interfaces with various DFT packages such as VASP, Quantum ESPRESSO, CASTEP, and SIESTA. By writing Python scripts, researchers can efficiently conduct density functional theory (DFT) calculations. Furthermore, ASE supports interfaces with machine learning interatomic potentials software like DeepMD,[22] NequIP,[23] and CHGNET.[24] This capability enables rapid and accurate predictions of mechanical properties, bridging the gap between traditional DFT calculations and large-scale atomistic simulations. Pymatgen is a Python-based high-throughput computational materials science library with extensive interfaces for VASP. Through tailored Python scripting, researchers can efficiently prepare input files and seamlessly import calculation results into Python for data analysis and visualization.

These robust interfaces have made high-throughput DFT calculations a practical reality. For repetitive tasks, these tools, when used in conjunction with custom scripts, can drastically

†Corresponding author. E-mail: hmweng@iphy.ac.cn
‡Corresponding author. E-mail: quansheng.wu@iphy.ac.cn

© 2025 Chinese Physical Society and IOP Publishing Ltd. All rights, including for text and data mining, AI training, and similar technologies, are reserved.

cut down on redundant work. However, in practical research scenarios, scientists often encounter far more intricate challenges. They may need to analyze certain properties across different materials or compare the behavior of a single material under varying conditions and parameters. Such diverse and complex computational tasks demand not only the preparation of various input files tailored to each specific case but also the repetitive submission of jobs and waiting for their completion. This process can be time-consuming and labor-intensive, highlighting the necessity for further advancements in automation and integration within computational workflows. Automatic execution and management of these tasks will allow researchers to focus more on scientific discovery rather than on the fussy details of computation.

In recent years, the rapid advancement of large language models (LLMs)[25–30] and agent-based technologies has opened new possibilities for building automatic DFT calculation schema. Large language models are deep neural networks pretrained on vast amounts of general textual corpora and further fine-tuned on dialogue data, enabling them to emulate human-like language understanding and reasoning to a significant extent. LLMs have been applied to research in chemistry and material science. Some use LLMs as a predictive model to predict the synthesizability of crystal structures[31,32] and other properties. Others utilize LLMs to extract data from articles.[33,34] There is also research to fine-tune LLMs on data of domain-specific data.[35]

Based on conversational capabilities, LLM developers have incorporated structured input–output formats into the fine-tuning process, equipping these models with the ability to invoke external programs. This ability is known as function calling or tool use. As LLMs continue to improve in performance, agent systems augmenting LLMs by tools and context have attracted wide attention. These systems not only provide more accurate responses but also gain the ability to interact with computational tools and environments.

Due to the accumulation of lengthy context from multiple tool invocations, single-agent systems often struggle to maintain optimal performance when executing complex, multistep tasks. To address this limitation, multi-agent frameworks such as MetaGPT,[36] LangGraph,[37] and CrewAI[38] have emerged as preferred solutions for managing complex workflows. These frameworks decompose intricate tasks into a series of simpler subtasks, which are then assigned to specialized agents equipped with distinct tools, domain knowledge, and independent contextual states. By enabling coordinated execution, such frameworks achieve improved robustness and scalability, empowering LLMs to effectively handle complex scientific computing tasks. In this way, multi-agent systems have potentials to become an AI coworker of human researchers.

There have been some successful attempts to leverage LLM-based multi-agent frameworks (MAF) to assist materials research. For instance, AtomAgents[39] aims to automate alloy design by integrating a knowledge base with LAMMPS[40]-based molecular dynamics simulation tools. TopoMAS[41] accelerates the discovery and investigation of topological materials by combining literature mining, generative models, and VASP calculations. PhysAgent[42] utilizes domain specified simulation tools like Quantum ESPRESSO and VASP to help accelerate discovery of physical laws. There are attempts such as MatPilot[43] and ChemAgent[44] that leverage robotic arms to oversee and conduct experimental procedures. Moreover, El Agente Q[45] and DREAMS,[46] respectively, explore the MAF on xTB[47] and Quantum ESPRESSO to accelerate DFT computations.

Despite these initial demonstrations for multi-agent frameworks (MAFs) to contribute to chemistry and materials science research, several challenges remain to be addressed for their effective use in facilitating practical computations. Firstly, beyond routine tasks like band structure and density of states (DOS) calculations, MAFs should also handle complex tasks without task-specific examples to maximize its adaptability. Secondly, although MAFs can expedite research processes, the role of human scientists remains indispensable. Consequently, the user interface is crucial. Scientists need real-time, intuitive understanding of the status of MAF to ensure the validity and correctness of results. Lastly, while VASP software boasts a substantial user base within the field of condensed matter physics, there has been a notable lack of focus on developing MAFs specifically tailored for VASP, indicating an area ripe for further exploration and development.

To bridge these gaps, in this work, we introduce an open source software, VASPilot. VASPilot leverages the CrewAI framework for multi-agent cooperation, facilitating automated DFT calculation and visualization for more complicated missions. This tool aims to further simplify the workflow of VASP even for zero-shot missions and practically reduce the researchers' effort executing VASP calculations.

In the following, we first describe the VASPilot architecture, covering the CrewAI-based multi-agent collaboration core, the model context protocol (MCP) tool server, and the web-based user interface. We then demonstrate VASPilots robustness and versatility through a series of benchmark studies. Finally, we present a concise conclusion and outlook.

2. Methods

2.1. Overall structure of VASPilot

The VASPilot program is primarily composed of three components: a web-based interface, a multi-agent collaborative core, and a tool server for the agents (Fig. 1). The multi-agent collaboration module is implemented using the CrewAI

framework,[38] an open-source Python library for constructing cooperative AI agent teams. CrewAI facilitates the automation of complex tasks through modular workflows, role-specific agents, and flexible tool integration. The tool server is encapsulated within the MCP,[48] a standardized interface for tool invocation and data provisioning. By adhering to MCP, our tools can be seamlessly incorporated into other frameworks with minimal additional effort.

Within this framework, we deploy one manager agent (orange box in Fig. 1) alongside three specialized worker agents (dashed box in Fig. 1): the crystal structure agent, the VASP agent, and the result validation agent. In this work, all agents are powered by DeepSeek-V3-0324,[26] but they can be easily replaced with other LLMs in practice. The crystal structure agent performs crystal structure searches, manipulations, and analyses. When the user does not specify a certain crystal structure file, the agent will search and download a structure from Materials Project.[39] The VASP agent orchestrates VASP calculations and produces the corresponding visualizations. The result validation agent evaluates the accuracy and reliability of the computed outcomes.

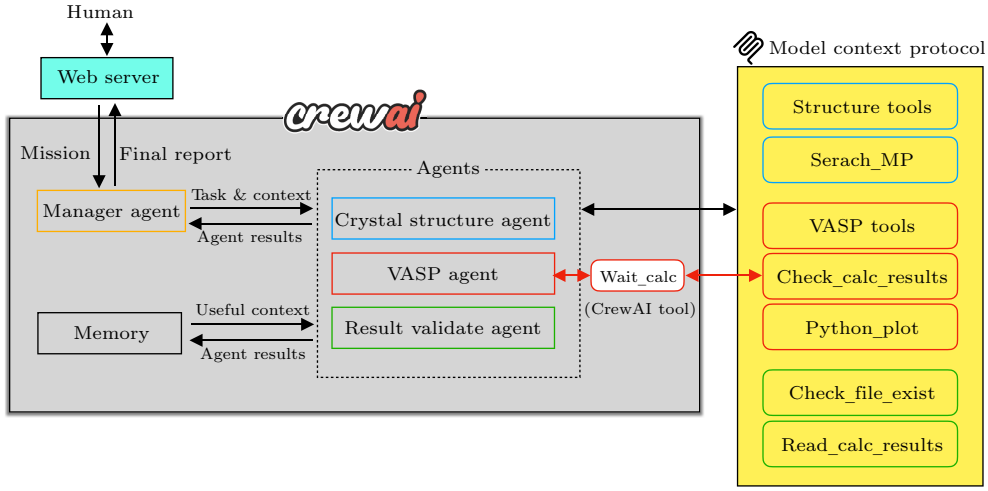


Fig. 1. Overall stucture of VASPilot. VASPilot consists of three main components: web server (blue background box), CrewAI based multi agent cooperation core (gray background box), and model context protocol type tool server (yellow backgound box).

Each worker agent maintains its own independent context and system prompt and can invoke only the subset of MCP-provided tools relevant to its tasks. In the case of the VASP agent, we supplement the standard VASP-specific MCP tools with a custom CrewAI utility, 'wait_calc_tool'. This tool periodically polls the MCP server to determine whether ongoing calculations have finished, thereby mitigating potential network timeouts during long-running jobs.

When a user submits a mission via the web interface, the manager agent decomposes it into discrete tasks and dispatches each to the appropriate worker agent. Upon completion, each agent returns a detailed execution report to the manager. All outputs are archived in a retrieval-augmented generation (RAG)[50]-based memory pool: as an agent begins a task, it retrieves relevant historical entries to enrich its context and inform its decisions.

2.2. Tools and model context protocol server

In agentic LLM systems, tool design plays a pivotal role in ensuring both efficient performance and robust task execution. Because VASP generates extensive output that exceeds the native I/O capacity of most language models, we have implemented a coherent suite of utilities built on the pymatgen library to manage every stage of the VASP workflow, as shown in Fig. 2.

First, our input-preparation tools (dashed box in Fig. 2) automatically generate all necessary VASP input files and submit jobs to a Slurm scheduler. Each submission is tagged with a unique calculation ID, which, together with its Slurm job number, is recorded in a centralized database. To minimize communication overhead, only the calculation ID is returned to the agent. In the event of a workflow restart (for example, a non-self consistent band-structure run), the tool retrieves the relevant metadata and input files by calculation ID, reconstructs the working directory, and resubmits the job seamlessly.

Second, we provide a status-query utility that reports on both failed and successful calculations (the check_calc_results tool in Fig. 2). For failed jobs, the tool returns the VASP error messages; for successful runs, it reads the full result set via pymatgen, archives the data in the database, and extracts a concise summary of key properties (e.g., total energy, band gap, and Fermi energy) for downstream LLM consumption.

Finally, to enable flexible post-processing, we developed a plotting tool that executes client-provided Python code (python_plot tool in Fig. 2). Given a list of calculation IDs and plotting instructions, the tool retrieves the corresponding datasets, runs the plotting code, and returns either the path to

the generated image or any error messages encountered. This end-to-end framework offloads heavy I/O from the LLM, guarantees full traceability of every task, and supports reliable, error-tolerant computation and visualization.

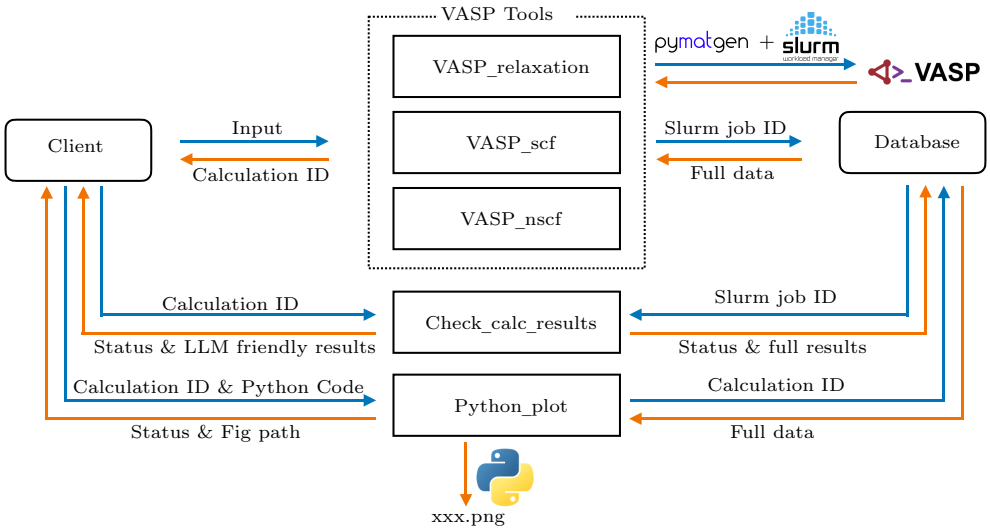


Fig. 2. Specially tailored tools in VASPilot. VASP tools prepare input files and submit the calculation through Pymatgen and Slurm. 'Check_calc_results' checks the calculation status, reads the results and stores full data into database. 'Python_plot' reads full calculation data from database and executes the python code given by the client.

2.3. Intuitive web interface

Although VASPilot automates complex workflows, user oversight remains indispensable to ensure the accuracy of both intermediate steps and final outcomes. To facilitate this interaction, we have developed an intuitive, web-based interface using Python's Quart framework[51] and Flask framework[52] (Fig. 3).

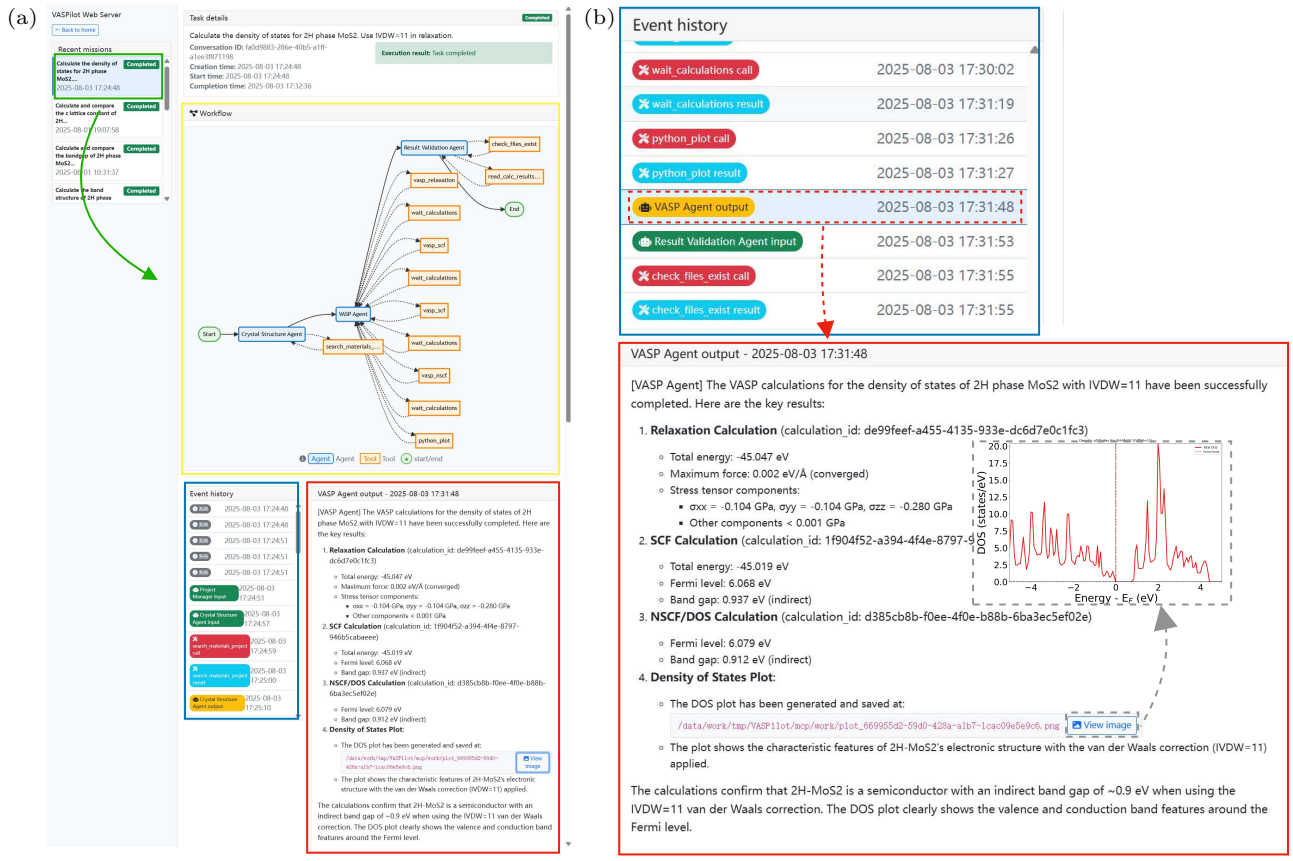


Fig. 3. Web user interface of VASPilot. (a) The overview of the web user interface. By clicking the mission record in green box, the detail of mission will be shown on the right. The yellow box is the real-time workflow of the VASPilot. The blue box is the execution history of agents and tools. The red box is the details of selected execution history. (b) Enlarged view of the execution history. The detailed information will be shown when user selects an event. In detailed information area, figures and crystal structure files can be immediately shown by clicking the button.

On the main page, users submit mission directives in natural language via a dedicated input dialog. On the left side of this dialog, a chronological history of all previously submitted tasks is presented. By selecting any entry (green box in Fig. 3(a)), the user is navigated to a detailed task view in which a flowchart (yellow box in Fig. 3(a)) graphically represents the execution sequence and current status. Beneath the flowchart, expandable "Agent Execution" and "Tool Execution" logs (Fig. 3(b)) provide full visibility into each agent's and tool's inputs and outputs. Moreover, within these logs, crystal-structure visualizations and generated plots can be inspected in greater detail via clearly labeled buttons (marked by gray dashed box in Fig. 3(b)). This design empowers users to both monitor overall progress at a glance and drill down into the computational data for rigorous verification.

3. Examples

To evaluate VASPilot reliability, we performed band-structure and density-of-states (DOS) calculations for 2H-MoS_2, as illustrated in Fig. 4. The workflow began when the manager agent instructed the crystal structure agent to provide the crystal structure. This agent automatically retrieved the structure from the Materials Project database[49] using the specified spacegroup. Following task delegation, the VASP agent sequentially executed structural relaxation and self-consistent field (SCF) calculations by invoking requisite tools. During the initial SCF calculation, VASP returned a Bravais lattice inconsistency error. In response, VASPilot intelligently adjusted the symmetry tolerance parameter, resolving the issue and enabling successful completion of the subsequent SCF calculation. The VASP agent then performed a non-self-consistent field (NSCF) band-structure calculation and generated the electronic band structure plot via built-in visualization tools. The manager agent subsequently directed the result validation agent to verify the computational results. Acting autonomously, this agent confirmed the existence of the band structure image and the integrity of the computation using its dedicated validation tools. Finally, the manager agent synthesized all the results and compiled a comprehensive task report. The workflow of the DOS calculation is almost the same as that of the band structure calculation.

In this band-structure calculation, the large language model (LLM) backend consumed approximately 160000 tokens in total, of which only about 4000 were output tokens. The significant input overhead arises because, in multi-turn agent execution, the conversation history is repeatedly included in the context. Such redundant token usage is a common issue for multi-turn agent systems, and we expect it will diminish as LLM frameworks continue to evolve.

We further evaluated VASPilot's ability to execute novel, more complex workflows by defining three benchmark scenarios. First, we performed a plane-wave cutoff convergence test by calculating the total energy of 2H-MoS_2 over a range of ENCUT values from 300 eV to 500 eV. Second, we conducted structural relaxations of 2H-MoS_2 using several van der Waals correction schemes, then compared the optimized lattice constants against experimental measurements. Third, we computed and contrasted the band gaps of the 2H phases of MoS_2, $MoSe_2$, WS_2, and WSe_2. VASPilot successfully completed all three tasks. The detailed prompts and resulting data for these test cases are summarized in Fig. 5.

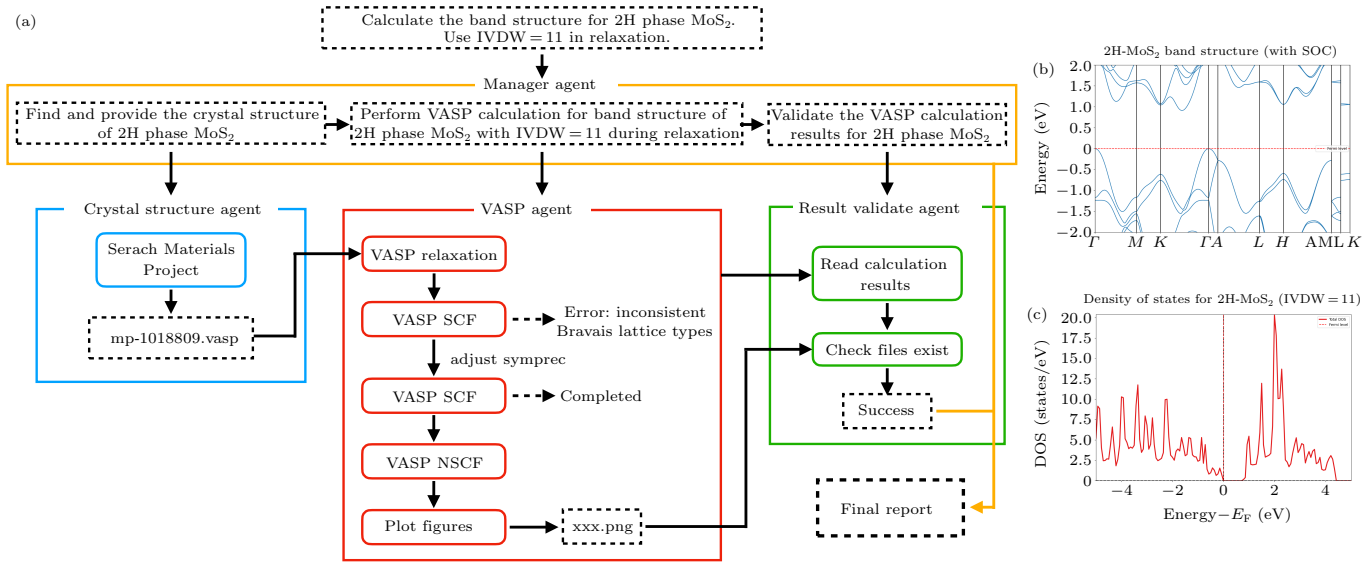


Fig. 4. Band structure and density of states calculations of 2H-MoS_2 utilizing VASPilot. (a) Workflow of band structure calculation. (b) Band structure figure calculated and plotted by VASPilot. (c) Density of states figure calculated and plotted by VASPilot. The workflow of density of states calculation is almost the same as that of the band structure calculation.

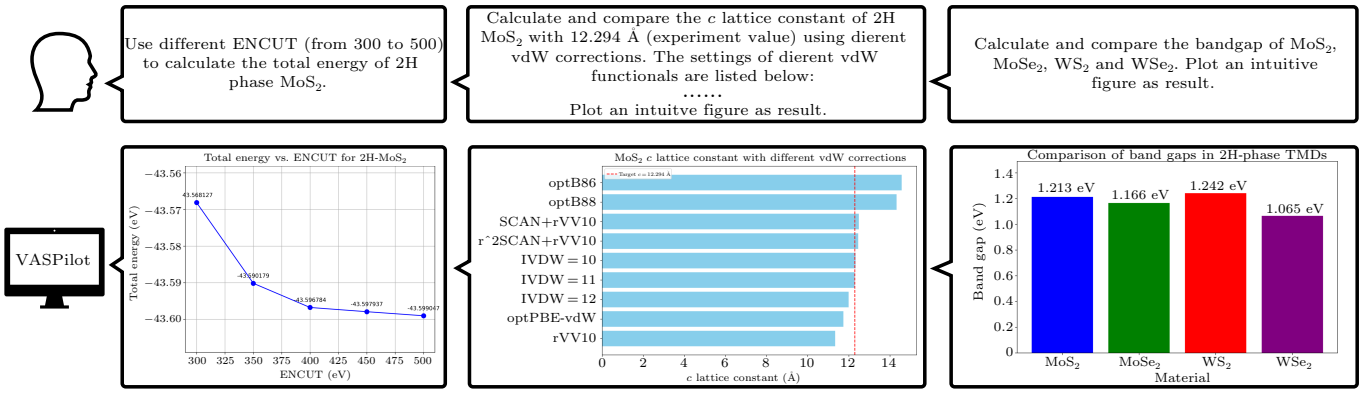


Fig. 5. Prompts and results of three complex missions. The resulting figures are given by VASPilot in its final report. All these missions are finished without examples as context. In the second column, IVDW = 10 is the DFT-D2 method of Grimme.[53] IVDW = 11 is the DFT-D3 method of Grimme with zero-damping function.[54] IVDW = 12 is the DFT-D3 method with Becke–Johnson damping function.[55] IVDW = 20 is the Tkatchenko–Scheffler method.[56] OptB86,[57] optB88,[58] SCAN + rVV10,[59] r^{2}SCAN + rVV10,[60] rVV10[61] and optPBE-vdW[58] are nonlocal vdW corrections.

These case studies underscore VASPilot's robustness and flexibility. The platform reliably executed standard workflows, such as electronic band-structure and density-of-states calculations, and upon encountering VASP execution errors, automatically parsed the error messages, adjusted input parameters, and successfully resumed the runs. Moreover, even for complex missions lacking any contextual examples, VASPilot completed the tasks without intervention. Overall, these results demonstrate that VASPilot is a powerful and adaptable framework for fully automated DFT calculations.

4. Conclusion

In this work, we have developed VASPilot, an open source framework for fully automated VASP calculations. Built atop the CrewAI architecture and the model context protocol (MCP), VASPilot seamlessly invokes specialized tools for structure retrieval and manipulation, input file generation, Slurm job orchestration, and postprocessing of results. We demonstrated its capability by executing automated band-structure and density-of-states workflows in which VASPilot parsed error messages, adapted input parameters on the fly, and successfully resumed interrupted runs. Beyond standard tasks, VASPilot handled more demanding benchmarks: converging total energies over a range of ENCUT values, optimizing lattice constants under different van der Waals correction schemes, and comparing band gaps across multiple 2H phase transition metal dichalcogenides.

These case studies attest to the robustness and adaptability of VASPilot, enabling truly autonomous DFT workflows. Moreover, these capabilities can be readily extended through the integration of additional agents and tools, and the underlying framework can be easily adapted to other DFT codes by deploying the corresponding MCP server. We anticipate that VASPilot will streamline VASP-based research pipelines, reducing both the technical overhead and turnaround time for complex simulations. More broadly, we hope this work will inspire the development of similar agent-tool ecosystems for other computational platforms, accelerating automation across materials modeling and computational physics.

Program availability

The VASPilot source code and comprehensive documentation are publicly available under an open-source license on GitHub (https://github.com/JiaxuanLiu-Arsko/VASPilot).

Acknowledgments

Project supported by the Science Center of the National Natural Science Foundation of China (Grant No. 12188101), the National Key R&D Program of China (Grant Nos. 2023YFA1607400 and 2022YFA1403800), the National Natural Science Foundation of China (Grant Nos. 12274436, 11925408, and 11921004), and the New Cornerstone Science Foundation through the XPLORER PRIZE. The AI-driven experiments, simulations and model training were performed on the robotic AI-Scientist platform of the Chinese Academy of Science.

References

[1] Xiao R, Li H and Chen L 2012 *Chem. Mater.* **24** 4242
[2] Wang Y, Yu X, Xu S, Bai J, Xiao R, Hu Y S, Li H, Yang X Q, Chen L and Huang X 2013 *Nat. Commun.* **4** 2365
[3] Wang X, Xiao R, Li H and Chen L 2017 *Phys. Rev. Lett.* **118** 195901
[4] Jia Y, Yu J, Liu J, Herzog-Arbeitman J, Qi Z, Pi H, Regnault N, Weng H, Bernevig B A and Wu Q 2024 *Phys. Rev. B* **109** 205121
[5] Wang C, Zhang X W, Liu X, He Y, Xu X, Ran Y, Cao T and Xiao D 2024 *Phys. Rev. Lett.* **132** 036501
[6] Devakul T, Crpel V, Zhang Y and Fu L 2021 *Nat. Commun.* **12** 6730
[7] Zhang S, Wu Q, Liu Y and Yazyev O V 2019 *Phys. Rev. B* **99** 035142
[8] Liu Z, Zhang S, Fang Z, Weng H and Wu Q 2021 *Phys. Rev. Res.* **6** 043185
[9] Pi H, Zhang S, Xu Y, Fang Z, Weng H and Wu Q 2024 *npj Comput. Mater.* **10** 276
[10] Kresse G and Furthmller J 1996 *Phys. Rev. B* **54** 11169
[11] Giannozzi P, Baroni S, Bonini N, *et al.* 2009 *J. Phys.: Condens. Matter* **21** 395502

[12] Giannozzi P, Andreussi O, Brumme T, *et al.* 2017 *J. Phys.: Condens. Matter* **29** 465901
[13] Clark S J, Segall M D, Pickard C J, Hasnip P J, Probert M I J, Refson K and Payne M C 2005 *Z. Kristallogr.* **220** 567
[14] Ozaki T 2003 *Phys. Rev. B* **67** 155108
[15] Weng H, Ozaki T and Terakura K 2009 *Phys. Rev. B* **79** 235118
[16] Soler J M, Artacho E, Gale J D, Garca A, Junquera J, Ordejn P and Snchez-Portal D 2002 *J. Phys.: Condens. Matter* **14** 2745
[17] Blaha P, Schwarz K, Tran F, Laskowski R, Madsen G K H and Marks L D 2020 *J. Chem. Phys.* **152** 074101
[18] The Elk Code URL http://elk.sourceforge.net/
[19] Wang V, Xu N, Liu J C, Tang G and Geng W T 2021 *Comput. Phys. Commun.* **267** 108033
[20] Hjorth Larsen A, Jrgen Mortensen J, Blomqvist J, *et al.* 2017 *J. Phys.: Condens. Matter* **29** 273002
[21] Ong S P, Richards W D, Jain A, Hautier G, Kocher M, Cholia S, Gunter D, Chevrier V L, Persson K A and Ceder G 2013 *Comput. Mater. Sci.* **68** 314
[22] Wang H, Zhang L, Han J and E W 2018 *Comput. Phys. Commun.* **228** 178
[23] Batzner S, Musaelian A, Sun L, *et al.* 2022 *Nat. Commun.* **13** 2453
[24] Deng B, Zhong P, Jun K, Riebesell J, Han K, Bartel C J and Ceder G 2023 *Nat. Mach. Intell.* **5** 1031
[25] Yang A, Li A, Yang B, *et al.* 2025 arXiv:2505.09388 [cs.CL]
[26] DeepSeek-AI, Liu A, Feng B, *et al.* 2024 arXiv:2412.19437 [cs.CL]
[27] DeepSeek-AI, Guo D, Yang D, *et al.* 2025 arXiv:2501.12948 [cs.CL]
[28] Team G V, Hong W, Yu W, *et al.* 2025 arXiv:2507.01006 [cs.CV]
[29] Meta AI 2025 *Llama 4: A new era in multimodal intelligence*
[30] Team G, Kamath A, Ferret J, *et al.* 2025 arXiv:2503.19786 [cs.CL]
[31] Kim S, Jung Y and Schrier J 2024 *J. Am. Chem. Soc.* **146** 19654
[32] Song Z, Lu S, Ju M, Zhou Q and Wang J 2024 arXiv:2407.07016 [cond-mat.mtrl-sci]
[33] Zheng Z, Zhang O, Borgs C, Chayes J T and Yaghi O M 2023 *J. Am. Chem. Soc.* **145** 18048
[34] Lee S, Cruse K, Baibakova V, Ceder G and Jain A 2025 *Text-mined dataset of solid-state syntheses*
[35] Wang X D, Chen Z R, Guo P J, Gao Z F, Mu C and Lu Z Y 2025 arXiv:2507.16307 [cs.LG]
[36] Hong S, Zhuge M, Chen J, *et al.* 2024 *MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework*
[37] LangChain AI 2023 *LangGraph: Build resilient language agents as graphs*
[38] CrewAI 2024 *CrewAI: The Leading Multi-Agent Platform*
[39] Ghafarollahi A and Buehler M J 2024 arXiv:2407.10022 [cs.AI]
[40] Thompson A P, Aktulga H M, Berger R, *et al.* 2022 *Comput. Phys. Commun.* **271** 108171
[41] Zhang B, Li X, Xu H, Jin Z, Wu Q and Li C 2025 arXiv:2507.04053 [cond-mat.mtrl-sci]
[42] Han X Q, Gao Z F, Guo P J and Lu Z Y 2025 *PhysAgent: A Multi-Agent Approach to Automated Discovery of Physical Laws*
[43] Ni Z, Li Y, Hu K, *et al.* 2024 arXiv:2411.08063 [physics.soc-ph]
[44] Song T, Luo M, Zhang X, *et al.* 2025 *J. Am. Chem. Soc.* **147** 12534
[45] Zou Y, Cheng A H, Aldossary A, *et al.* 2025 *Matter* **8** 102263
[46] Wang Z, Huang H, Zhao H, Xu C, Zhu S, Janssen J and Viswanathan V 2025 arXiv:2507.14267 [cs.AI]
[47] Bannwarth C, Caldeweyher E, Ehlert S, *et al.* 2021 *WIREs Comput. Mol. Sci.* **11** e1493
[48] Model Context Protocol 2024 *Model context protocol*
[49] Jain A, Ong S P, Hautier G, *et al.* 2013 *APL Mater.* **1** 011002
[50] Lewis P, Perez E, Piktus A, *et al.* 2020 arXiv:2005.11401 [cs.CL]
[51] Pallets Projects 2025 *Quart*
[52] Pallets Projects 2025 *Flask*
[53] Grimme S 2006 *J. Comput. Chem.* **27** 1787
[54] Grimme S, Antony J, Ehrlich S and Krieg H 2010 *J. Chem. Phys.* **132** 154104
[55] Grimme S, Ehrlich S and Goerigk L 2011 *J. Comput. Chem.* **32** 1456
[56] Tkatchenko A and Scheffler M 2009 *Phys. Rev. Lett.* **102** 073005
[57] Klime J, Bowler D R and Michaelides A 2011 *Phys. Rev. B* **83** 195131
[58] Klime J, Bowler D R and Michaelides A 2010 *J. Phys.: Condens. Matter* **22** 022201
[59] Peng H, Yang Z H, Perdew J P and Sun J 2016 *Phys. Rev. X* **6** 041005
[60] Ning J, Kothakonda M, Furness J W, *et al.* 2022 *Phys. Rev. B* **106** 075422
[61] Sabatini R, Gorni T and De Gironcoli S 2013 *Phys. Rev. B* **87** 041108