AGENTIC AI FOR COMPUTER VISION: A REVIEW

Anwaar Ulhaq
Central Queensland University, Australia
a.anwaarulhaq@cqu.edu.au

ABSTRACT

General Computer vision models operate as passive systems that produce one output and stop, while agentic AI for computer vision represents a shift toward autonomous visual decision-making systems. In such systems, visual input is used to plan actions, decide next steps, and refine results through feedback. It allows the model to select or design the optimal processing steps and improve its output over time. This review maps current research on agentic approaches in computer vision and examines how autonomy is implemented in practice. This search is based on standard databases, including IEEE Xplore, ACM Digital Library, Scopus, Web of Science, arXiv, and OpenReview, and only studies where the final task was a computer vision outcome were included. Papers that used agents for non-visual tasks were excluded. Screening, selection, and data charting were conducted using the PRISMA Scoping Review (PRISMA ScR) guidelines. A scoping review was chosen instead of a systematic review because the field is new, and preprints dominate the evidence base. Each included study was examined for its autonomy design, visual modality, datasets, and evaluation method. The results indicate that agentic approaches are promising for tasks that involve multiple steps, self-correction, or interaction with an environment. This review clarifies the emerging landscape of agentic AI for computer vision and identifies research gaps, including the need for stronger definitions of autonomy, shared testing environments, and reproducible evaluation methods.

***K*eywords** Agentic AI · Agentic computer vision · Autonomous vision systems · Vision agents

1 Introduction

Deep learning has transformed computer vision with impressive results in tasks such as classification, detection and segmentation. Over the last ten years, progress in the field has been driven not only by architectural innovation but also by automation of the design process. Early work on neural architecture search [1] and AutoML [2] explored how models could automatically design other models by searching through vast configuration spaces for the best-performing architecture or training strategy. These developments showed that part of the design process could become automated and that intelligence could be used to improve intelligence.

However, recent advances have moved beyond model design. Instead of improving architectures before deployment, researchers began to investigate systems that can improve their own outputs during execution. This shift has contributed to the emergence of agentic AI [3], a term used to describe systems that can set goals, break tasks into steps, act in an environment, evaluate outcomes and adjust their behaviour. Traditional AI agents were designed to perform a single task within predefined boundaries. Agentic AI, in contrast, adapts, plans and improves while the task is being performed.. Therefore, Agentic AI differs from earlier AI agents by having broader autonomy, the ability to coordinate multiple tools or capabilities, and a capacity to update strategies based on feedback [4].

The idea that perception should be intrinsically coupled with action is not new [5]. Early work on active perception argued that visual processing should be guided by task goals and contextual demands rather than treated as a passive acquisition of the world [6]. This paradigm subsequently influenced research in embodied AI [7], where perception continuously informs motor decisions and navigation within an environment. Contemporary language–vision systems and multimodal agents [8] revisit these principles by integrating perception, reasoning, and action into a unified loop, enabling models to observe a scene, plan, and execute interactions rather than merely interpret visual inputs. Only recently has this thinking converged into what we refer to in this work as agentic computer vision.

In the paradigm of agentic AI for computer vision, the vision system is not a passive recogniser. Instead, visual input is used to plan actions, decide next steps and refine the output using feedback. The model can inspect intermediate results, select or design the next processing operation and improve its visual output through repeated cycles. This is fundamentally different from passive computer vision, which produces a single prediction and does not evaluate or improve that prediction. Although individual research efforts are appearing in areas such as interactive segmentation [9], object detection refinement [10], and interactive visual navigation [11], the literature is fragmented across preprints, conference papers and project repositories. There is no unified overview of how these systems are being implemented, what autonomy mechanisms they rely on or how their performance is evaluated. Existing reviews of agentic approaches focus mainly on language agents or general multimodal agent systems, without isolating cases where the final output is a computer vision task.

Given the trend novelty of this topic and the inconsistent terminology across studies, a scoping review is suitable. A scoping review allows us to map concepts, identify themes and highlight gaps without estimating effect sizes. This work follows the PRISMA Scoping Review (PRISMA-ScR) guidelines [12]. Only studies where the final goal is a computer vision outcome are included, and studies are excluded if the agent produces a non-visual end task. Given the trend and novelty of this topic and the inconsistent terminology across studies, this paper presents an early scoping review of the topic. A scoping review allows us to map concepts, identify themes and highlight gaps without estimating effect sizes. This work follows the PRISMA Scoping Review (PRISMA-ScR) guidelines [12]. Only studies where the final goal is a computer vision outcome are included, and studies are excluded if the agent produces a non-visual end task. This review addresses three research questions with relevant rationale: (1) How are agentic methods currently being applied to computer vision tasks? This matters because most prior literature focuses on general agentic AI architectures and overlooks how agentic AI breaks down and executes visual workflows. (2) What mechanisms support autonomy, including planning, iterative refinement, and tool or model selection? Clarifying these mechanisms is necessary to distinguish simple tool invocation from genuine agent-driven decision-making. (3) How are these systems evaluated, and what gaps exist in identifying and reproducibility? This question is essential because current evaluations are often inconsistent or based on ad hoc success reports, which makes it difficult to compare studies.

Various surveys [3, 13, 4] have recently been published on the broader topic of Agentic AI; none has focused specifically on agentic autonomy within computer vision. To the best of our knowledge, there is no consolidated understanding of how vision-centred agents operate, how they differ in autonomy level, or how emerging systems are evaluated. This scoping review addresses that gap by providing the first systematic synthesis of Agentic AI for computer vision, including a taxonomy of autonomy levels, associated mechanisms, and application contexts. A key strength of this work is the use of the PRISMA-ScR methodology, which ensures that the mapping of literature is transparent, reproducible, and methodologically rigorous. By applying a formal evidence-based review process, the survey establishes a structured foundation for future research and positions agentic computer vision as a distinct emerging field within Agentic AI.

The paper is organised as follows. Section 2 describes the review methodology in detail, including the screening of existing literature. It also defines the taxonomy based on analysis of the existing literature. Section 3 examines how these agents operate by analysing the architectural patterns, workflows, and mechanisms they use to plan actions, call tools, or generate executable code. It explains what current research is doing by mapping the landscape of agentic computer vision systems and describing their capabilities across the identified autonomy levels. Section 4 discusses applications, while Section 5 focuses on how these systems are evaluated, highlighting the metrics, benchmarks, and assessment practices used to measure performance, autonomy, and reliability.

1.1 Defining Agentic AI for Computer Vision

Agentic AI for Computer Vision for in short, Agentic Computer Vision is a class of systems in which computer vision operates inside a closed loop of perception, reasoning, and action rather than as a one-off output. An agent sets or receives a goal, interprets visual inputs, plans a sequence of steps, selects or composes tools or executable code, acts on the world or data, and then re-analyses the resulting images to decide what to do next. Vision is both an input and a target state, and behaviour adapts to feedback instead of producing a single fixed prediction. This excludes pipelines that only run a detector or segmenter once without planning, tool choice, or revision. Minimal agentic behaviour includes tool selection guided by visual evidence, while stronger forms include multi-step workflow orchestration and program synthesis with iterative self-correction.

Autonomy is the defining characteristic of agentic computer vision. Instead of producing a single visual output, the agent decides how a task should be approached, orchestrates the workflow, and adapts its behaviour based on the evolving state of the visual information. Autonomy is demonstrated when the system selects which tools to use, interprets intermediate results, refines its actions, or generates new code to achieve a visual goal.

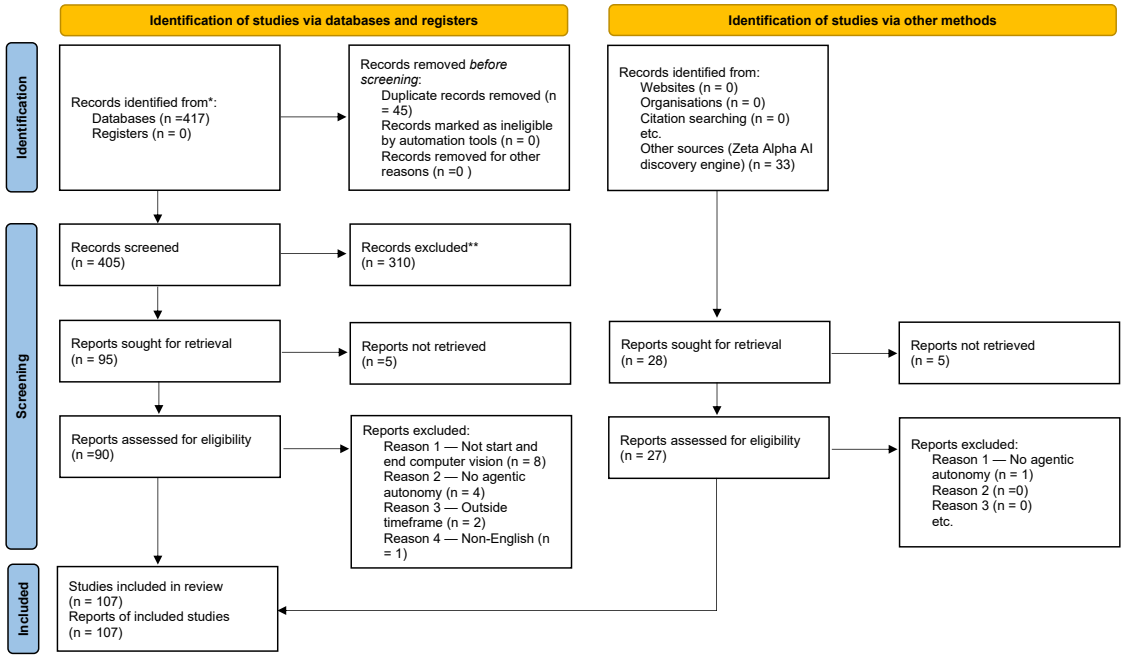

Figure 1: PRISMA-scR flow diagram for new systematic reviews which included searches of databases, registers and other sources.

Agentic computer vision is based on the core principle that perception is not a final output but the driver of active, goal-directed behaviour. Traditional computer vision models operate passively: they take an image as input and produce a fixed output, with no awareness of context, goals, or consequences. In contrast, an agentic computer vision system uses visual information to decide what to do next. Vision becomes part of a continuous perception–reasoning–action loop in which the agent observes, interprets, acts, and evaluates the outcome to guide subsequent steps. What makes these systems agentic is the presence of dynamic autonomy: perception influences action, action produces new perceptual evidence, and this cycle continues until the goal is achieved.

2 Methodology: PRISMA-ScR

A comprehensive scoping review was conducted to identify studies on agentic computer vision, following the PRISMA-ScR reporting guidelines. Five scholarly databases were searched (Google Scholar, Scopus, Web of Science, ACM Digital Library, and IEEE Xplore), resulting in 417 records. An additional 33 studies were identified from other sources through an AI-powered literature discovery tool (Zeta Alpha). After removing 45 duplicate records, 405 studies proceeded to title and abstract screening. Of these, 310 were excluded, leaving 95 records for full-text retrieval. Five full-text articles could not be obtained, and 90 studies proceeded to eligibility assessment. During full-text screening, 15 reports were excluded for the following reasons: lack of end-to-end computer vision workflow (n = 8), no evidence of agentic autonomy such as tool use, planning, or code execution (n = 4), publication outside the defined time window (n = 2), and non-English language (n = 1). A total of 107 studies met all inclusion criteria and were therefore included in the final synthesis, with 107 corresponding reports analysed. These studies formed the basis for the taxonomy development of agentic computer vision systems and the subsequent thematic and autonomy-level classification (A1: Tool-Calling Agents, A2: Vision-Orchestrating Agents, A3: Code-Executing Agents). The inclusion and exclusion criteria are reported in Tables 1 and 2, respectively.

Table 1: Inclusion Criteria Used in Study Selection

Category	Inclusion Criterion	Logical Rule (IF / THEN)	Rationale
Publication & Temporal	Date of publication within a range	IF publication date is **between 1 Jan 2024 and 30 Oct 2025**, THEN continue screening.	Agentic Computer Vision has emerged recently; it ensures the latest advancements are captured.
Publication & Language	English language	IF the full paper is written in English, THEN continue.	Ensures consistent interpretation and avoids translation bias.
Access & Completeness	Full text + abstract available	IF abstract and full PDF are accessible, THEN continue.	Required for transparent screening and data extraction.
Scope of Study	Computer Vision start–end workflow	IF workflow begins with CV input **AND** ends with CV output, THEN continue.	Ensures review focuses on **agentic computer vision**, not generic UI/chatbot agents.
Agentic Autonomy Requirement	Must demonstrate autonomy	IF the agent demonstrates AI tool-calling, AI pipeline orchestration, or code execution, THEN continue.	Distinguishes **agentic CV** from perception-only models.
Empirical Validation	Experiments or implementation included	IF the paper reports a working system or evaluation, THEN include.	Ensures results are not purely conceptual/speculative.
Publication Type	Peer-reviewed journal, conference, or preprint (full PDF)	IF formally published or preprint with full PDF, THEN include.	Allows preprints due to the rapid evolution of the field.

In our preliminary literature search, broader search queries containing generic terms such as “agent”, “AI agent”, “autonomous agent” or “agent-based system” returned several thousand irrelevant papers. As the term “agent” is widely used across non-vision domains (e.g., software agents, conversational agents, recommender agents, reinforcement learning agents), it has led to excessive noise and a low precision rate. Therefore, to ensure search feasibility and maintain conceptual alignment with the research objective, we intentionally applied a narrow and high-precision search strategy focused on terms that explicitly indicate agentic behaviour in the context of computer vision with a refined search query a follows:

```
"agentic computer vision" OR "vision agent" OR "visual agent"
```

This strategy ensured that the retrieved studies were directly relevant to the scoping review’s eligibility framework (vision-in $\rightarrow$ agentic behaviour $\rightarrow$ vision-out), while avoiding the retrieval of thousands of unrelated “agent” papers from other domains. After retrieval, snowballing and backward citation search were used to capture any relevant papers that used alternative terminology. This narrow search approach is consistent with scoping review methodology, where feasibility and relevance take priority over exhaustive sensitivity in early-stage emerging research domains.

2.1 Taxonomy of Autonomy Levels

This scoping review identified that autonomy is the core defining principle of agentic computer vision systems. Therefore, we define a new taxonomy of computer vision agents based on the autonomy mechanisms that enable agents to perceive, plan, act, and refine their behaviour. It is presented in Figure 2. Three autonomy levels in Agentic Computer Vision systems were discovered. At Level A1, Tool Calling Vision Agents (n = 5) use computer vision only to trigger an external tool or model. The agent does not coordinate a sequence of actions. Instead, it interprets a visual input and selects a tool such as SAM, YOLO, or OCR to complete the task. These systems behave like “vision routers”, because their autonomy is limited to choosing which existing tool to call. Representative studies show large language models selecting segmentation or detection tools based on visual grounding. Although these agents operate independently to a degree, they do not generate new pipelines or code. As a result, A1 represents the most basic form of agentic autonomy.

Most studies fall into Level A2, Vision Orchestrating Agents (n = 91). Here, the agent plans and executes multi-step visual workflows. Three subcategories appeared in the data. First, Screen, UI, or Web Vision Agents (n = 77) perceive

Table 2: Exclusion Criteria Applied During Screening and Full-Text Review

Exclusion Criterion	Logical Rule (IF / THEN)	Rationale (why excluded)
Outside timeframe	IF publication date is before **1 Jan 2024** or after **30 Oct 2025**, THEN exclude.	Ensures relevance to the emergence period of agentic computer vision.
Non-English language	IF the paper is not written in English, THEN exclude.	Avoids translation ambiguity and ensures consistent screening.
Unavailable or incomplete text	IF the abstract or full-text PDF cannot be accessed, THEN exclude.	Screening and extraction cannot be performed reliably.
Not start–end computer vision	IF workflow does **not begin with a CV input and end with a CV output**, THEN exclude.	Ensures adherence to the definition: **vision-in → agent → vision-out**.
No agentic autonomy present	IF no evidence of tool use, pipeline orchestration, or code execution or agentic functionality, THEN exclude.	Model is perception-only (single forward pass), not agentic.
Insufficient methodological detail	IF the autonomy level or workflow cannot be identified from the abstract/full text, THEN exclude.	Prevents invalid classification and improves reproducibility.
Text extraction blocked (OCR fail/locked PDF)	IF PDF is image-scanned or locked, preventing text extraction, THEN exclude.	PRISMA-SCR requires traceable evidence for synthesis.
Wrong domain	IF the study focuses on LLM/Chatbot/task agents without computer vision output, THEN exclude.	Ensures correct domain: **agentic computer vision**, not genetic agenticAI.

on-screen elements such as buttons, icons, or bounding boxes and take actions including clicking, scrolling, or extracting information. Second, Auto CV Pipeline Agents (n = 17) construct complete computer vision workflows that include data preprocessing, model training, and evaluation, acting as AutoML or AutoCV assistants. Third, a smaller group of Self-Refining Agents (n = 4) refine their outputs by evaluating results and attempting improved solutions, such as retrying a segmentation step or modifying prompts.

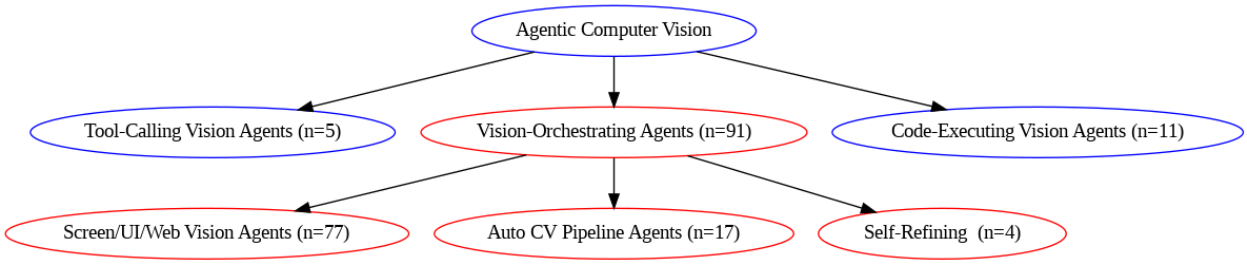

Figure 2: Agentic Computer Vision taxonomy derived via PRISMA-ScR: A1 (tool-calling), A2 (vision orchestration with verified non-zero subcategories), and A3 (code-executing). Counts reflect included studies (n=107).

Level A3, Code Executing Vision Agents (n = 11), demonstrates the highest level of autonomy observed in the review. These agents generate executable code, run it, debug errors, and repeat the process. They move beyond tool selection and are capable of creating custom computer vision algorithms through a perception, reasoning, program synthesis, and execution loop. This makes A3 the strongest expression of agentic autonomy identified in the current literature.

3 Autonomy Mechanisms and Types of Agents

The studies included in this review show that autonomy in agentic computer vision arises when visual perception is tightly coupled with reasoning and execution. Rather than functioning as passive classifiers or detectors, these systems interpret visual input, decide on an action, perform the action, and then reassess the visual state that results from that action. This perception–action loop forms the foundation of autonomy across all three observed categories.

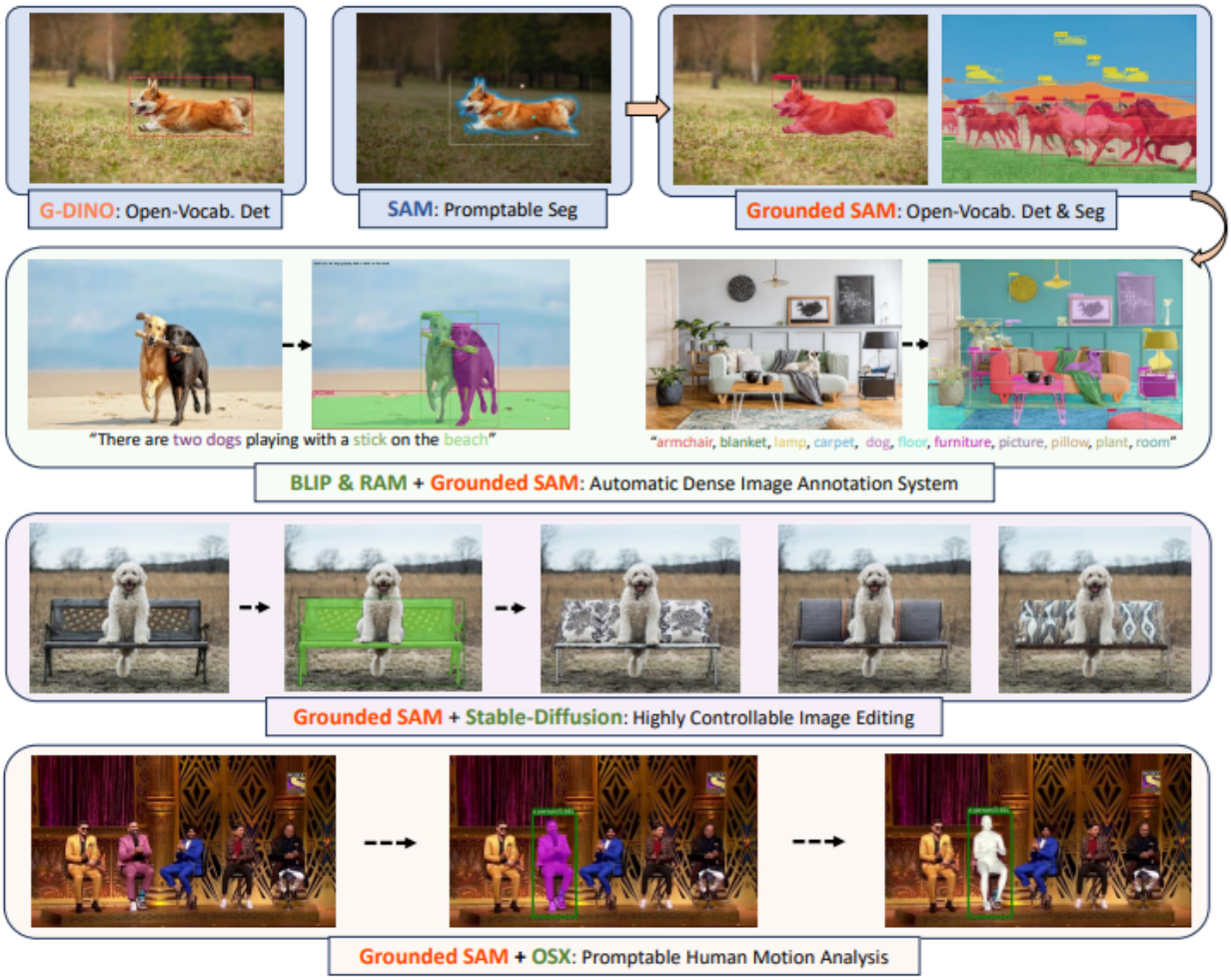

Figure 3: Grounded-SAM, an A1 tool-calling vision agent. The agent detects and segments regions in an image based on arbitrary text prompts and can integrate with other open-world vision models to perform more complex tasks. *Image courtesy of [14].*

The first autonomy mechanism is visual tool selection. In this mode, autonomy arises from the system's ability to interpret visual input and decide which external tool or model to invoke. The agent does not generate new processing logic; instead, it performs a single decision: selecting the most appropriate action based on what it sees. For example, when an image contains text, the system may choose an OCR model, or when it detects an object or region, it may call a segmentation or recognition tool. This behaviour is demonstrated in systems such as Grounded SAM [14] that represents a tool-calling autonomy mechanism because visual input is used solely to decide which existing computer vision model should be executed. The system combines Grounding DINO [20], an open-vocabulary object detector, with the Segment Anything Model (SAM) [21]. When given an image and a text prompt, it detects relevant regions using Grounding DINO and immediately passes those regions to SAM to produce segmentation masks. Autonomy is limited to selecting and invoking the appropriate tool based on the visual context. The system does not plan multiple steps, modify its workflow, or refine outputs through iteration. Instead, it behaves reactively: perception triggers a single model call, making Grounded SAM an example of agentic computer vision where vision is used to route the task to the correct specialised model.

The second autonomy mechanism is vision-guided orchestration of multi-step workflows. In this mode, autonomy emerges not from selecting a single tool, but from using visual feedback to plan and adapt a sequence of actions. The system observes the screen or image, determines a subtask, executes an action, and then re-examines the new visual state to decide what should happen next. In ScreenAgent [19], for example, the agent treats the computer interface as a visual environment: it recognises interactive elements such as buttons and icons, performs actions such as clicking or

Table 3: Representative Tool-Calling Vision Agents (A1)

Agent name / citation	Autonomy (how it is achieved)	Computer vision task(s)
Grounded-SAM (GroundingDINO + SAM) [14]	Visual input triggers GroundingDINO to detect regions and forwards bounding box to SAM for segmentation. No planning, no feedback loop. Fixed detect→segment pipeline.	Open-vocabulary detection and segmentation.
FM-Fusion: Instance-Aware Semantic Mapping Boosted by Vision-Language Foundation Models (Liu et al., 2024) [15]	Agent calls RAM + GroundingDINO + SAM to obtain object detections and masks. Tool-use is dominant; mapping fusion is deterministic (no workflow reasoning).	RGB-D semantic mapping, instance detection, segmentation.
F-ViTA: Foundation Model Guided Visible-to-Thermal Translation (Paranjape et al., 2025) [16]	Uses RAM → GroundingDINO → SAM to extract object masks, then passes masks to diffusion model. Tool-invocation only; no multi-step adaptation.	Object detection + segmentation used to guide thermal image generation.
GroundingDINO-US-SAM: Text-Prompted Multi-Organ Ultrasound Segmentation (2025) [17]	Text prompt triggers GroundingDINO to detect ROI, which is then segmented by SAM. Vision only routes to a downstream model; fixed workflow.	Medical object detection and segmentation (ultrasound imaging).
SAM-Based Automated Image Annotation (GroundingDINO→SAM) (2024) [18]	Vision triggers annotation pipeline that calls detection (GroundingDINO) and mask generation (SAM). No refinement or reasoning beyond tool call.	Automated bounding-box and segmentation mask generation.

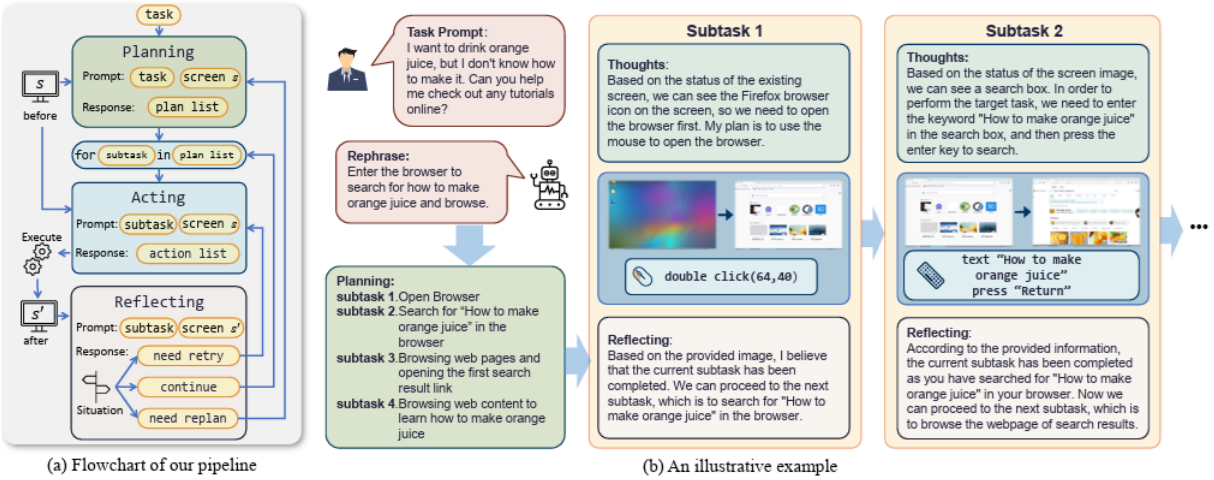

Figure 4: Overview of the computer control pipeline, an A2 vision-orchestrating agent. The agent plans, acts, and reflects by decomposing a user task into subtasks, generating mouse and keyboard actions from screen observations, and deciding whether to continue, retry, or reformulate the plan. *Image courtesy of [19].*

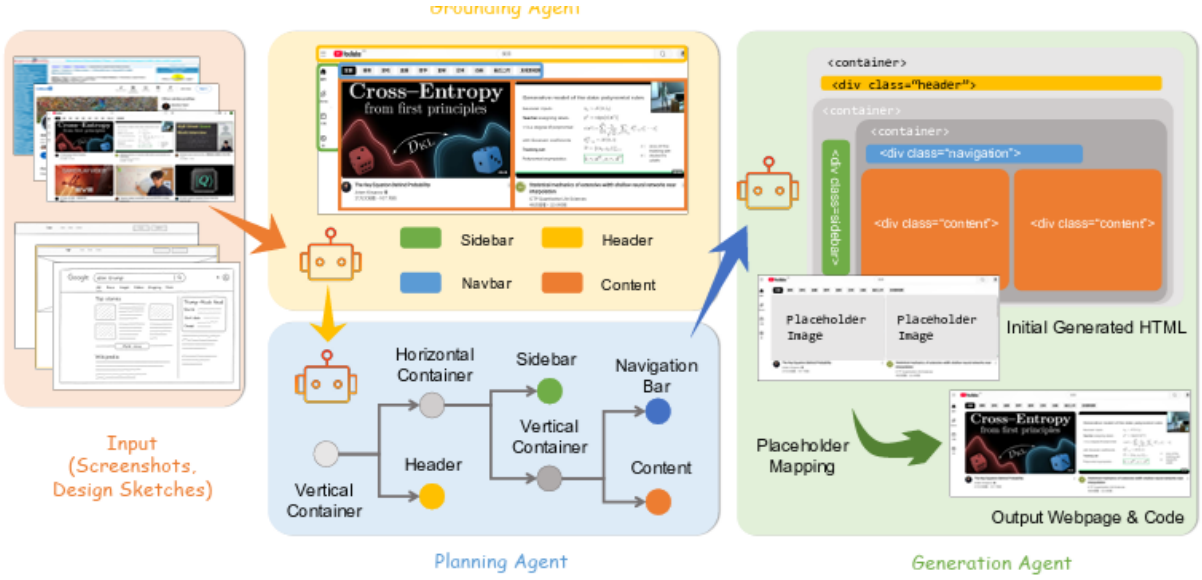

Figure 5: Overview of ScreenCoder, an A3 code-generating vision agent. The agent detects UI elements from screenshots, plans the layout, and synthesises corresponding HTML code. *Image courtesy of [22].*

scrolling, and evaluates the resulting screenshot to determine whether the goal has been achieved or if further steps are required. Other systems follow a similar pattern when assembling computer vision pipelines, passing outputs from one model (such as detection) to another (such as segmentation or attribute extraction) and adapting the workflow when intermediate results are unsatisfactory. Across these systems, autonomy is expressed through a loop of planning, acting, observing, and revising. Vision is no longer the final output, but the information source that continuously drives decision-making.

The third autonomy mechanism is vision-conditioned code synthesis and execution. In this mode, autonomy emerges when the system generates executable code rather than relying solely on predefined tools. The agent analyses the input (visual or task-level), generates code to perform the required operation, runs the code, inspects the outcome, and revises the code if the result is incorrect or incomplete. This creates a closed loop where perception leads to program generation, execution produces new evidence, and that evidence guides further code refinement. CodeAgent [35] is a representative example of this mechanism: it decomposes a task into subtasks, writes Python code to address each part, executes it, evaluates whether the output satisfies the goal, and rewrites portions of the code when necessary. CodeAgent represents the highest level of autonomy because the system creates new solutions rather than selecting from predefined ones. Unlike tool-calling or pipeline-orchestrating mechanisms, it generates executable code to achieve a task, runs that code, evaluates the outcome, and rewrites the program when the result does ot meet the goal.

Across these three autonomy mechanisms, a common capability becomes evident. A system becomes truly agentic when visual information not only enables recognition but actively drives reasoning and triggers actions that alter the visual state. The progression from selecting tools to orchestrating multi-step workflows, to generating and refining executable code, illustrates a clear evolution in autonomy. The field is moving from reactive systems that apply tools when instructed, toward proactive systems that design, execute, and test their own visual workflows. This shift marks the transition from computer vision systems that merely answer questions to computer vision systems that independently solve problems. In light of these mechanisms, we classify vision agents into three types, each reflecting a distinct level of autonomy.

Definition 1 (Tool-Calling Vision Agents (A1))**.** *Tool-calling vision agents are agents that use visual input only to select and execute an existing computer vision tool or model. Their autonomy is limited to choosing which tool to call; they do not construct workflows, refine results, or use feedback from previous steps to determine what happens next. Vision functions merely as a trigger for a single action. A representative example is Grounded SAM [14], where the agent detects objects in an image using Grounding-DINO and immediately forwards the detected region to SAM to generate a segmentation mask.*

Definition 2 (Vision-Orchestrating Agents (A2))**.** *Vision-orchestrating agents plan and execute multi-step visual workflows. Instead of calling a single tool and stopping, they break a task into visual subtasks, determine the sequence*

Table 4: Representative Vision–Orchestrating Agents (A2), showing how autonomy emerges through perception→action→observation loops and how each aligns with A2 sub-categories.

Agent name / citation	How autonomy is achieved (why A2)	Computer vision task(s)	Sub-category (why)
ScreenAgent [19]	Observes interface, plans click/scroll, executes, re-reads screenshot, and revises plan with perception→action→observation loop.	GUI grounding; on-screen interaction.	**A2-S1** as screen/UI actions are chosen from visual feedback.
OmniParser [23]	Parses UI layout, selects next action from grounded regions, executes, observes new screenshot, and adapts sequence.	UI parsing; region extraction.	**A2-S1** as decisions are conditioned on screenshot updates.
SeeAct / GPT-4V Web Agent [24]	Grounds elements on a screenshot, performs an action, inspects the resulting page, and adjusts the next step when the expected change fails.	Web navigation using bounding-box grounding.	**A2-S1** as the next action is determined by the visual state.
Mobile-Agent-V2 [25]	Planner→executor→reflection; evaluates screen outcome after each action and re-plans based on visual feedback.	Mobile UI automation.	**A2-S1** since reflection is triggered by screenshot results.
MobileUse [26]	Checks the visual success/failure of each executed step, updates the workflow, and retries actions when needed.	Mobile workflow via visual reasoning.	**A2-S1** because visual confirmation controls the next UI step.
WebVoyager [27]	Reads screenshot, selects click/scroll, observes new webpage, continues or adjusts depending on what appears.	Web navigation from screenshots.	**A2-S1** since navigation depends on sequential visual changes.
WebGUM [28]	Uses screenshot + HTML grounding, executes action, and retries when the visual goal is not achieved.	Multimodal visual web navigation.	**A2-S1** as planning is screen-based and feedback-driven.
HAMMR [29]	Selects between OCR/detector/captioning models based on intermediate visual outputs; reroutes tools if the result is insufficient.	Multi-tool orchestration for VQA.	**A2-S2** since it assembles a CV workflow using visual routing.
Autonomous CV Development with Agentic AI [30]	Builds CV pipeline (preprocess→segment→evaluate), uses visual validation to refine module choices, then executes training/inference.	Auto CV pipeline construction; segmentation + evaluation.	**A2-S2** as agent constructs a full CV workflow rather than UI actions.
SAM2Auto (SMART-OD + FLASH) [31]	Evaluates segmentation/track quality frame-by-frame, adjusts configuration, and retries to improve output.	Tracking + segmentation refinement.	**A2-S3** as self-correction is triggered by visual evaluation.

of actions, observe the intermediate visual outcome, and adapt subsequent actions based on what they see. Autonomy emerges from a continuous perception–reasoning–action loop in which the agent evaluates visual progress toward the goal and updates its plan accordingly. A representative example is ScreenAgent [19], where the agent identifies actionable elements in a screenshot, executes an operation such as clicking or scrolling, inspects the resulting screen, and co-acts until the task is completed. Unlike tool-calling agents, these agents act strategically and refine their decisions over time.

Three subcategories of A2 agents were observed. The first subcategory consists of Screen, UI and Web Interaction Agents (A2 S1). These agents operate inside visual environments such as screens, graphical interfaces or web pages. They observe the interface, take an action such as clicking or scrolling, examine the new screenshot, and continue acting only after confirming visual progress. The second subcategory consists of Auto Computer Vision Pipeline Agents (A2 S2). These agents construct multi-stage computer vision workflows. They choose and combine operations such as preprocessing, segmentation, model selection and evaluation, and adapt the workflow in response to intermediate visual outputs. The third subcategory consists of Self-Refining Vision Agents (A2 S3). These agents evaluate the quality of their own visual output and attempt an improved version when the result is not satisfactory. While A2 S1 agents choose actions based on what appears on the screen, A2 S2 agents design computer vision workflows by interpreting visual data, and A2 S3 agents actively critique and refine their own visual reasoning.

Definition 3 (Code-Executing Vision Agents (A3))**.** *Code-executing vision agents generate executable code to create a computer vision solution, run that code, inspect the resulting visual output, and revise the code when necessary. Rather than selecting or orchestrating existing tools, they construct the vision pipeline itself through program synthesis. Autonomy emerges from an iterative process of generating code, executing it, evaluating the visual result, and self-correcting until the goal is achieved. A representative example of this agent type is CodeAgent [35], which demonstrates*

Table 5: Representative Code-Executing Vision Agents (A3)

Agent name / citation	Autonomy (how it is achieved)	Computer vision task(s) automated by the agent
RECODE: Reasoning Through Code Generation [32]	Transforms a visual input into executable code, runs the program, compares the rendered output with the image and rewrites code until the visual objective is met. Autonomy arises from the image→code→execution→revision loop.	Image de-rendering; diagram and layout reconstruction via code; visual question answering through program execution.
ChartGen: Scaling Chart Understanding via Code-Guided Chart Generation [33]	Generates plotting code from a chart image, executes it, visually compares the plot and iteratively edits code to match structure and style.	Chart-to-code synthesis for bar, line, scatter and pie charts; axis, legend and style reconstruction from images.
ScreenCoder: Visual-to-Code Generation for Front-End Automation [22]	Grounds UI elements in screenshots, plans code blocks, generates front-end code, renders the result and revises when the rendering diverges from the screenshot.	UI screenshot to HTML/CSS/JS; component detection and layout parsing; text/icon extraction for pixel-to-code front-end reproduction.
VisCodex: Unified Multimodal Code Generation [34]	Uses visual inputs with language prompts to produce executable code, verifies outcomes by running code and updates code when the visual goal is not met.	Vision-conditioned code generation for HTML/CSS from UI images; chart plotting scripts from chart images; SVG and image-processing script synthesis.
CodeAgent [35]	Reads a visual goal, plans and writes executable Python or OpenCV code, executes it, inspects the visual output and rewrites code until the goal is satisfied. Autonomy is expressed through code generation, execution and iterative self-correction.	Programmatic image processing and CV pipelines: thresholding and binary segmentation; edge and contour extraction; morphological operations; geometric measurement and annotation.

code-executing autonomy through program synthesis and iterative self-correction. These agents do not merely apply computer vision but they also invent it for the task at hand.

4 Agentic AI Computer Vision: Applications

The reviewed literature demonstrates that agentic computer vision systems are now being applied across a broad range of domains. In all cases, the defining characteristic is that the system does more than interpret visual input. It makes decisions about subsequent actions and carries them out autonomously.

Agents are now heavily employed in image- and video-based reasoning and action, where perception unfolds over time [36, 37, 38, 39]. In VideoAgent [40], the agent moves along the video timeline, selects informative frames, trims clips, and decides whether additional analysis is required. The emphasis shifts from static classification accuracy to the agent's capacity to reason about temporal sequences and adjust actions as the visual context evolves. Video understanding becomes an iterative perception-and-action process rather than a single inference pass. Agent-based video trimming is introduced in [41]. Image restoration agents are also on the rise [42], and AgenticIQA [43] is an agentic framework for adaptive and interpretable image quality assessment, with [44] extending this to video quality assessment. Agent-based image generation was also introduced recently [45]. Video keyframe extraction is performed by [46] for video question answering, video anomaly detection in [47], guided segmentation in [48], and object detection in [49].

In medical imaging, agentic systems support diagnostic workflows that require segmentation, measurement, refinement, and verification [50, 51]. MedAgent-Pro [52] shows how an agent segments imaging data, evaluates whether the output satisfies diagnostic requirements, and activates alternative tools when needed. The agent does not simply return a segmentation result; instead, it constructs an imaging workflow, evaluates intermediate outputs, and improves the pipeline autonomously. Chest X-ray lungs, heart, and ribs segmentation is performed in a proof-of-concept visual agent system architecture illustrating the interaction between the OpenManus agentic AI framework and the SimpleMind cognitive AI environment [53].

Remote sensing and environmental monitoring illustrate how these capabilities extend to large-scale imagery [54]. RS-Agent [55] is an AI agent designed to interact with human users and autonomously leverage specialized models to address the demands of real-world remote sensing applications. In Geo-OLM [56], the agent selects appropriate

analysis tools, segments satellite images, identifies land features, and evaluates outputs iteratively. Instead of relying on pre-generated masks, the agent reasons about land-cover changes and adapts its own actions during processing.

One major application domain is screen, user interface, and web automation. In this context, agents receive screenshots of software interfaces, detect actionable elements, and choose how to interact with them [57]. For example, ScreenAgent [19] shows how an agent visually identifies buttons, icons, and menus, and autonomously performs tasks such as navigating applications or completing data entry. Similarly, Vision [58] demonstrates that agents can decompose complex user interface tasks into smaller visual subtasks and execute them from beginning to end without relying on predefined automation scripts.

Another substantial cluster of applications relates to document understanding. Instead of performing traditional optical character recognition, the agent actively determines where to focus and whether to repeat extraction if initial results are unsatisfactory. FormFactory [59] inspects document layouts, selects relevant regions, performs optical character recognition, and verifies extracted content. This behaviour reveals a form of visual reasoning that adapts to document structure. The visual document agent OmniParser [23] is a document-understanding agent driven by visual cues.

The final category reflects the most advanced form of autonomy: code-generating vision agents. CodeAgent [35] is an agent that writes Python or OpenCV code to perform segmentation or filtering, executes the code, evaluates the visual output, identifies mistakes, and rewrites the program until the desired outcome is achieved. Autonomy here comes from the creation of new computer vision pipelines through program synthesis rather than selecting among existing tools.

Across all these domains, the common theme is a shift from passive perception to proactive decision-making. Agentic computer vision systems do not stop at recognising objects or extracting information. They choose actions, execute workflows, evaluate their own output, and make refinements. The representative studies illustrate a transformation of computer vision from static interpretation to dynamic visual autonomy.

5 Evaluation Methods, Benchmarks, and Reproducibility

Evaluation across the reviewed literature reveals that benchmarking agentic computer vision remains inconsistent and underdeveloped. Unlike traditional computer vision, which relies on fixed accuracy-based metrics, agentic systems must be assessed not only on the quality of their visual output but on their ability to make decisions, refine actions, and complete multi-step tasks. The result is an emerging shift from static prediction benchmarks toward process-oriented evaluation [60, 61].

In studies focusing on GUI and web automation, performance was primarily measured through task completion rates across unseen interactive interfaces. For example, ScreenAgent [19], a Vision-Language Agent for GUI Automation, evaluates agents on their ability to complete real interface workflows, using metrics such as end-to-end task success, number of steps taken, and error recovery when misclicks occur. Similarly, Vision Task Agent, Agvis [58] introduces a benchmark of human-computer tasks requiring navigation through nested interface structures, and evaluates success not only by final outcome but by whether the agent maintained autonomy throughout the task without reverting to a fallback scripted routine. The emphasis is placed on grounded action and consistency, not just recognition.

In document understanding, reproducibility is approached through structured benchmark suites that enforce consistent input–output evaluation. The most comprehensive example is FormFactory[59] introduces a controlled dataset of PDF and scanned forms where accuracy is measured on the correct extraction of field values and on whether the agent is able to self-correct errors through reprocessing of visual regions. This benchmark highlights a key distinction in agentic systems: performance depends on the interaction loop, not solely on the OCR or visual model accuracy. The agent must decide where to zoom, crop, or retry extraction.

Video-based agentic systems introduce yet another evaluation dimension. In VideoAgent[40], the authors measure success through clip relevance and efficiency, evaluating whether the agent can navigate to the correct location in a temporal sequence and isolate the target event with minimal iterations. Unlike classical video models, the evaluation reflects reasoning and planning ability, acknowledging that the agent's correctness depends on multi-step temporal decisions.

More advanced forms of autonomy, particularly those involving code synthesis, require evaluation strategies that resemble software testing rather than computer vision benchmarking. In CodeAgent[35], the evaluation involves visual correctness of the generated result and the agent's ability to detect failures and fix them through iterative debugging. Success is recorded when the output image satisfies a visually defined goal or when the generated code completes execution without producing errors. Here, performance reflects the agent's capacity for self-repair and learning from visual outcomes, not the accuracy of a single inference.

Table 6: Application Domains of Agentic Computer Vision Identified in the Literature

Application Domain	Real-world Problem	Agent Input	Agent Output	How the Agent Operates
Video-based Reasoning and Action	Agent navigates video timeline, selects keyframes, trims clips, performs anomaly detection or keyframe extraction [40, 46, 47].	Video frames / temporal sequence	Selected keyframes, trimmed clips, anomaly flags	Iteratively observes → selects frames → evaluates outcome → adapts next action.
Image Restoration / Image Quality / Image Generation	Image restoration, intelligent enhancement, adaptive quality assessment, agent-based image generation [42, 43, 45].	Images or degraded frames	Restored/enhanced output or generated image	Agent runs models, evaluates quality, retries or refines output autonomously.
Medical Imaging Workflows	Segmentation, measurement, verification of diagnostic outputs [52, 50, 51, 53].	MRI / CT / X-ray images	Segmentation masks, intermediate diagnostic results	Agent constructs a workflow, evaluates outputs, and refines pipeline until diagnostic criteria are satisfied.
Remote Sensing / Environmental Monitoring	Land-use mapping and large-scale satellite analysis [55, 56, 54].	Satellite / aerial images	Segmented regions, land-cover maps	Agent selects tools, analyses regions, verifies results, and adapts steps iteratively.
Screen / UI / Software Automation	Executes end-to-end automation: clicking, typing, form completion [57, 19, 58].	UI screenshots	Navigation actions, completed workflow	Observes UI → detects visual targets → executes action → verifies → continues.
Document Understanding	Form parsing, OCR, layout reasoning [59, 23].	Scanned forms / documents	Parsed structured data, verified OCR text	Determines region of interest, performs OCR, verifies output, repeats if incorrect.
Code-Generating Vision Agents	Generates and executes CV code; rewrites until result is correct [35].	Natural language goal or image	Executable Python/OpenCV code	Synthesises pipeline, executes code, inspects output, self-corrects and rewrites.

Across all studies, a recurring issue is that benchmarks are not standardised. Each paper defines its own success metrics, datasets, and task structures, and no shared community benchmark captures the full spectrum of agentic autonomy across visual domains. Reproducibility is further challenged by the reliance on closed-source APIs, model selection based on proprietary tool calls, and environments that are difficult to emulate. Some works, such as Geo-OLM [56], attempt to address this by making datasets and workflows public, yet tool availability and platform dependency still limit replication of full experiments.

The literature demonstrates that existing evaluation methods are sufficient for measuring perception, but insufficient for evaluating autonomy. Across domains, benchmarks are fragmented and overly task-specific, preventing meaningful comparison between different agentic vision systems. The reviewed studies collectively indicate that the field now needs standardisation: datasets capturing multi-step visual decision-making, metrics that quantify autonomy and reactivity, and public frameworks that allow repeatable experimentation.

6 Discussion, Research Gaps, and Future Directions

The reviewed literature shows that agentic computer vision marks a shift from passive perception to systems that attempt to plan, act, and refine their behaviour based on visual feedback. Instead of answering a question about an image, these systems are designed to complete a task using the image as a source of information. Across studies, autonomy is claer but narrow. Agents complete tasks successfully under controlled conditions but lose reliability when layouts, visual states, or task structures deviate from what they were designed for.

In graphical user interface and web automation, systems such as ScreenAgent [19] and Vision Task Agent [58] demonstrate that agents can interact with software interfaces. However, when an unexpected visual state appears or the layout changes, the agent often becomes stuck in action loops or commits incorrect actions. This indicates that grounding and memory remain insufficient for sustained autonomy. In document understanding, FormFactory [59] illustrates iterative reasoning over visual segments, yet error detection and recovery only succeed when failures are predictable. In video reasoning tasks, including VideoAgent [40], planning across time is possible but limited to temporally structured datasets. Similar challenges appear in remote sensing and medical imaging. Although agents can select tools, segment images, or verify results, they operate within rigid system prompts and lack true strategy discovery.

Several insights emerge from full-text review and PRISMA ScR synthesis. First, current systems are constrained not by perception, but by the absence of persistent memory, long-horizon planning, and self-evaluation. Even in agents capable of code synthesis, such as CodeAgent [35], decision making occurs step by step without linking earlier failures to future actions. No study demonstrates an agent that can retain knowledge across tasks or transfer strategies across visual environments. Autonomy is therefore task-bound rather than general.

Second, evaluation and reproducibility remain serious limitations. Almost every paper introduces its own dataset and success metric, which prevents comparison across studies. Existing computer vision metrics, such as accuracy or mean intersection over union, do not measure planning quality, error recovery, or sustained autonomy. Only a few papers attempt a structured evaluation of agentic behaviour, and none provide stable benchmarks. Reproducibility is further hindered by proprietary APIs, closed vision language models, and undocumented tool selection rules. The PRISMA ScR confirms high heterogeneity across studies, a lack of shared terminology, and low transparency in reporting.

Third, most papers focus on short-horizon orchestration, where the agent selects and executes tools but does not explain its decisions. Reasoning traces are rarely reported, making it difficult to analyse whether agents are truly reasoning or only following explicit system prompts. A3 code-generating agents exhibit stronger autonomy, yet they lack safety layers such as sandboxing, constrained execution, or reliable error containment, which limits real-world deployment.

Future research should address these shortcomings by aiming for generalisable autonomy rather than task-specific automation. Agentic architectures require hierarchical planning, explicit reasoning traces, and persistent memory, enabling learning across episodes rather than single-use decisions. Benchmarking must evolve toward open, environment-based evaluation that assesses error recovery, adaptation to unseen visual states, and robustness under uncertainty. Progress also requires transparent pipelines, open datasets, and accessible frameworks to enable replication. Finally, agents should be capable of self-improvement, integrating outcomes from previous attempts to refine prompts, code, or model selection strategies over time.

Agentic computer vision shows promise, but the field is early. The next stage of research must turn isolated demonstrations into reproducible, generalisable, and memory-driven agents capable of sustained visual autonomy.

7 Conclusion

This article has reviewed the rapidly emerging field of Agentic AI for Computer Vision and identified a clear shift from passive perception models to autonomous visual decision-making systems. The analysis of 417 records retrieved from major scholarly databases resulted in 107 included studies that met strict inclusion criteria. Our literature review revealed various insights. Across the literature, autonomy consistently emerged from the integration of perception, reasoning, and execution, where vision is not simply used to recognise objects but to guide action. Autonomy mechanisms were observed across diverse application domains, including GUI automation, document understanding, video analysis, medical imaging, remote sensing, industrial inspection, and visual editing. As autonomous visual agents become more capable of designing their own workflows, selecting and sequencing models, and generating executable code, computer vision will progress from static prediction to dynamic interaction. The trajectory of current research suggests that the future of visual intelligence will not be defined by how well systems see, but by how intelligently they plan, act and refine.

References

[1] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Computing Surveys (CSUR)*, 54(4):1–34, 2021.

[2] Xin He, Kaiyong Zhao, and Xiaowen Chu. Automl: A survey of the state-of-the-art. *Knowledge-based systems*, 212:106622, 2021.

[3] Deepak Bhaskar Acharya, Karthigeyan Kuppan, and B Divya. Agentic ai: Autonomous intelligence for complex goals–a comprehensive survey. *IEEe Access*, 2025.

[4] Ranjan Sapkota, Konstantinos I Roumeliotis, and Manoj Karkee. Ai agents vs. agentic ai: A conceptual taxonomy, applications and challenges. *arXiv preprint arXiv:2505.10468*, 2025.

[5] John Aloimonos, Isaac Weiss, and Amit Bandyopadhyay. Active vision. *International journal of computer vision*, 1(4):333–356, 1988.

[6] Ruzena Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, 1988.

[7] Jiafei Duan, Samson Yu, Hui Li Tan, Hongyuan Zhu, and Cheston Tan. A survey of embodied ai: From simulators to research tasks. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(2):230–244, 2022.

[8] Junlin Xie, Zhihong Chen, Ruifei Zhang, Xiang Wan, and Guanbin Li. Large multimodal agents: A survey. *arXiv preprint arXiv:2402.15116*, 2024.

[9] Minghao Zhou, Hong Wang, Qian Zhao, Yuexiang Li, Yawen Huang, Deyu Meng, and Yefeng Zheng. Interactive segmentation as gaussion process classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19488–19497, 2023.

[10] Yaoqi Sun, Lidong Ma, Peiyao Shou, Hongfa Wen, YuHan Gao, Yixiu Liu, Chenggang Yan, and Haibing Yin. Dynamic interactive refinement network for camouflaged object detection. *Neural Computing and Applications*, 36(7):3433–3446, 2024.

[11] Kuo-Hao Zeng, Luca Weihs, Ali Farhadi, and Roozbeh Mottaghi. Pushing it out of the way: Interactive visual navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9868–9877, 2021.

[12] Jessie McGowan, Sharon Straus, David Moher, Etienne V Langlois, Kelly K O'Brien, Tanya Horsley, Adrian Aldcroft, Wasifa Zarin, Chantelle Marie Garitty, Susanne Hempel, et al. Reporting scoping reviews—prisma scr extension. *Journal of clinical epidemiology*, 123:177–179, 2020.

[13] Soodeh Hosseini and Hossein Seilani. The role of agentic ai in shaping a smart future: A systematic review. *Array*, page 100399, 2025.

[14] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024.

[15] Chuhao Liu, Ke Wang, Jieqi Shi, Zhijian Qiao, and Shaojie Shen. Fm-fusion: Instance-aware semantic mapping boosted by vision-language foundation models. *IEEE Robotics and Automation Letters*, 9(3):2232–2239, 2024.

[16] Jay N Paranjape, Celso de Melo, and Vishal M Patel. F-vita: Foundation model guided visible to thermal translation. *arXiv preprint arXiv:2504.02801*, 2025.

[17] Hamza Rasaee, Taha Koleilat, and Hassan Rivaz. Groundingdino-us-sam: Text-prompted multi-organ segmentation in ultrasound with lora-tuned vision-language models. *arXiv preprint arXiv:2506.23903*, 2025.

[18] Fuseini Mumuni and Alhassan Mumuni. Segment anything model for automated image data annotation: empirical studies using text prompts from grounding dino. *arXiv preprint arXiv:2406.19057*, 2024.

[19] Runliang Niu, Jindong Li, Shiqi Wang, Yali Fu, Xiyu Hu, Xueyuan Leng, He Kong, Yi Chang, and Qi Wang. Screenagent: A vision language model-driven computer control agent. *arXiv preprint arXiv:2402.07945*, 2024.

[20] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European conference on computer vision*, pages 38–55. Springer, 2024.

[21] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023.

[22] Yilei Jiang, Yaozhi Zheng, Yuxuan Wan, Jiaming Han, Qunzhong Wang, Michael R Lyu, and Xiangyu Yue. Screencoder: Advancing visual-to-code generation for front-end automation via modular multimodal agents. *arXiv preprint arXiv:2507.22827*, 2025.

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

[24] Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. Gpt-4v (ision) is a generalist web agent, if grounded. *arXiv preprint arXiv:2401.01614*, 2024.

[25] Junyang Wang, Haiyang Xu, Haitao Jia, Xi Zhang, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. Mobile-agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration. *Advances in Neural Information Processing Systems*, 37:2686–2710, 2024.

[26] Ning Li, Xiangmou Qu, Jiamu Zhou, Jun Wang, Muning Wen, Kounianhua Du, Xingyu Lou, Qiuying Peng, and Weinan Zhang. Mobileuse: A gui agent with hierarchical reflection for autonomous mobile operation. *arXiv preprint arXiv:2507.16853*, 2025.

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

[28] Hiroki Furuta, Kuang-Huei Lee, Ofir Nachum, Yutaka Matsuo, Aleksandra Faust, Shixiang Shane Gu, and Izzeddin Gur. Multimodal web navigation with instruction-finetuned foundation models. *arXiv preprint arXiv:2305.11854*, 2023.

[29] Lluis Castrejon, Thomas Mensink, Howard Zhou, Vittorio Ferrari, Andre Araujo, and Jasper Uijlings. Hammr: Hierarchical multimodal react agents for generic vqa. *arXiv preprint arXiv:2404.05465*, 2024.

[30] Jin Kim, Muhammad Wahi-Anwa, Sangyun Park, Shawn Shin, John M Hoffman, and Matthew S Brown. Autonomous computer vision development with agentic ai. *arXiv preprint arXiv:2506.11140*, 2025.

[31] Arash Rocky and QM Wu. Sam2auto: Auto annotation using flash. *arXiv preprint arXiv:2506.07850*, 2025.

[32] Junhong Shen, Mu Cai, Bo Hu, Ameet Talwalkar, David A Ross, Cordelia Schmid, and Alireza Fathi. Recode: Reasoning through code generation for visual question answering. *arXiv preprint arXiv:2510.13756*, 2025.

[33] Jovana Kondic, Pengyuan Li, Dhiraj Joshi, Zexue He, Shafiq Abedin, Jennifer Sun, Ben Wiesel, Eli Schwartz, Ahmed Nassar, Bo Wu, et al. Chartgen: Scaling chart understanding via code-guided synthetic chart generation. *arXiv preprint arXiv:2507.19492*, 2025.

[34] Lingjie Jiang, Shaohan Huang, Xun Wu, Yixia Li, Dongdong Zhang, and Furu Wei. Viscodex: Unified multimodal code generation via merging vision and coding models. *arXiv preprint arXiv:2508.09945*, 2025.

[35] Xunzhu Tang, Kisub Kim, Yewei Song, Cedric Lothritz, Bei Li, Saad Ezzini, Haoye Tian, Jacques Klein, and Tegawendé F Bissyandé. Codeagent: Autonomous communicative agents for code review. *arXiv preprint arXiv:2402.02172*, 2024.

[36] Sahil Shah, Harsh Goel, Sai Shankar Narasimhan, Minkyu Choi, SP Sharan, Oguzhan Akcin, and Sandeep Chinchali. A challenge to build neuro-symbolic video agents. *arXiv preprint arXiv:2505.13851*, 2025.

[37] Konstantinos I Roumeliotis, Ranjan Sapkota, Manoj Karkee, and Nikolaos D Tselikas. Orchestrator-agent trust: A modular agentic ai visual classification system with trust-aware orchestration and rag-based reasoning. *arXiv e-prints*, pages arXiv–2507, 2025.

[38] Kevin Dela Rosa. Raven: An agentic framework for multimodal entity discovery from large-scale video collections. *arXiv preprint arXiv:2504.06272*, 2025.

[39] Xiaoyi Zhang, Zhaoyang Jia, Zongyu Guo, Jiahao Li, Bin Li, Houqiang Li, and Yan Lu. Deep video discovery: Agentic search with tool use for long-form video understanding. *arXiv preprint arXiv:2505.18079*, 2025.

[40] Xiaohan Wang, Yuhui Zhang, Orr Zohar, and Serena Yeung-Levy. Videoagent: Long-form video understanding with large language model as agent. In *European Conference on Computer Vision*, pages 58–76. Springer, 2024.

[41] Lingfeng Yang, Zhenyuan Chen, Xiang Li, Peiyang Jia, Liangqu Long, and Jian Yang. Agent-based video trimming. *arXiv preprint arXiv:2412.09513*, 2024.

[42] Kaiwen Zhu, Jinjin Gu, Zhiyuan You, Yu Qiao, and Chao Dong. An intelligent agentic system for complex image restoration problems. *arXiv preprint arXiv:2410.17809*, 2024.

[43] Han Zhu, Yi Tian, Ke Ding, Bowen Chen, Bohan Chen, Siheng Wang, and Weisi Lin. Agenticiqa: An agentic framework for adaptive and interpretable image quality assessment. *arXiv preprint arXiv:2509.26006*, 2025.

[44] Shuo Xing, Soumik Dey, Mingyang Wu, Ashirbad Mishra, Hansi Wu, Binbin Li, and Zhengzhong Tu. Q-router: Agentic video quality assessment with expert model routing and artifact localization. *arXiv preprint arXiv:2510.08789*, 2025.

[45] Jui-En Huang, I-Chun Fang, Tsun-Han Huang, Chia-Yao Wang, and Jia-Ching Chen. Gen-n-val: Agentic image data generation and validation. *arXiv preprint arXiv:2506.04676*, 2025.

[46] Shuo Fan, Meng-Hao Guo, and Shiyi Yang. Agentic keyframe search for video question answering. *arXiv preprint arXiv:2503.16032*, 2025.

[47] Zhiwei Yang, Chen Gao, and Mike Zheng Shou. Panda: Towards generalist video anomaly detection via agentic ai engineer. *arXiv preprint arXiv:2509.26386*, 2025.

[48] Tuyen Tran, Thao Minh Le, and Truyen Tran. Towards agentic ai for multimodal-guided video object segmentation. *arXiv preprint arXiv:2508.10572*, 2025.

[49] Furkan Mumcu, Michael J Jones, Anoop Cherian, and Yasin Yilmaz. Llm-guided agentic object detection for open-world understanding. *arXiv preprint arXiv:2507.10844*, 2025.

[50] Songhao Li, Jonathan Xu, Tiancheng Bao, Yuxuan Liu, Yuchen Liu, Yihang Liu, Lilin Wang, Wenhui Lei, Sheng Wang, Yinuo Xu, et al. A co-evolving agentic ai system for medical imaging analysis. *arXiv preprint arXiv:2509.20279*, 2025.

[51] Jinghao Feng, Qiaoyu Zheng, Chaoyi Wu, Ziheng Zhao, Ya Zhang, Yanfeng Wang, and Weidi Xie. M 3 builder: A multi-agent system for automated machine learning in medical imaging. In *International Workshop on Agentic AI for Medicine*, pages 115–124. Springer, 2025.

[52] Ziyue Wang, Junde Wu, Linghan Cai, Chang Han Low, Xihong Yang, Qiaxuan Li, and Yueming Jin. Medagent-pro: Towards evidence-based multi-modal medical diagnosis via reasoning agentic workflow. *arXiv preprint arXiv:2503.18968*, 2025.

[53] Jin Kim, Muhammad Wahi-Anwa, Sangyun Park, Shawn Shin, John M. Hoffman, and Matthew S. Brown. Autonomous computer vision development with agentic ai. *arXiv preprint arXiv:2506.11140*, 2025.

[54] Chaehong Lee, Varatheepan Paramanayakam, Andreas Karatzas, Yanan Jian, Michael Fore, Heming Liao, Fuxun Yu, Ruopu Li, Iraklis Anagnostopoulos, and Dimitrios Stamoulis. Multi-agent geospatial copilots for remote sensing workflows. *arXiv preprint arXiv:2501.16254*, 2025.

[55] Wenjia Xu, Zijian Yu, Boyang Mu, Zhiwei Wei, Yuanben Zhang, Guangzuo Li, and Mugen Peng. Rs-agent: Automating remote sensing tasks through intelligent agent. *arXiv preprint arXiv:2406.07089*, 2024.

[56] Dimitrios Stamoulis and Diana Marculescu. Geo-olm: Enabling sustainable earth observation studies with cost-efficient open language models & state-driven workflows. In *Proceedings of the ACM SIGCAS/SIGCHI Conference on Computing and Sustainable Societies*, pages 608–619, 2025.

[57] Damiano Marsili, Rohun Agrawal, Yisong Yue, and Georgia Gkioxari. Visual agentic ai for spatial reasoning with a dynamic api. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 19446–19455, 2025.

[58] Yiheng Xu, Zekun Wang, Junli Wang, Dunjie Lu, Tianbao Xie, Amrita Saha, Doyen Sahoo, Tao Yu, and Caiming Xiong. Aguvis: Unified pure vision agents for autonomous gui interaction. *arXiv preprint arXiv:2412.04454*, 2024.

[59] Bobo Li, Yuheng Wang, Hao Fei, Juncheng Li, Wei Ji, Mong-Li Lee, and Wynne Hsu. Formfactory: An interactive benchmarking suite for multimodal form-filling agents. *arXiv preprint arXiv:2506.01520*, 2025.

[60] Manish Shukla. Adaptive monitoring and real-world evaluation of agentic ai systems. *arXiv preprint arXiv:2509.00115*, 2025.

[61] Kaixin Li, Yuchen Tian, Qisheng Hu, Ziyang Luo, Zhiyong Huang, and Jing Ma. Mmcode: Benchmarking multimodal large language models for code generation with visually rich programming problems. *arXiv preprint arXiv:2404.09486*, 2024.