УДК 004.8:004.896

АГЕНТНАЯ ОРКЕСТРАЦИЯ ЛОКАЛЬНЫХ VLA-ПОЛИТИК В ЗАДАЧАХ МНОГОШАГОВОЙ РОБОТИЗИРОВАННОЙ МАНИПУЛЯЦИИ

Рашитов И.Н.[1]

Магистрант

Московский технический университет связи и информатики

Россия, Москва

Аннотация. В статье рассматривается агентная архитектура многошагового управления роботом-манипулятором, в которой LLM/VLM выполняет функции оркестратора, а локальная VLA-политика используется как исполнительный навык. Показано, что монолитные VLA-модели ограничены при задачах с длинным горизонтом из-за накопления ошибок, частичной наблюдаемости сцены и отсутствия независимой проверки результата. Обоснована схема управления с декомпозицией задачи, выбором инструмента, локальным исполнением, верификацией и перепланированием.

Ключевые слова: роботизированная манипуляция, многошаговое управление, Vision-Language-Action, VLA, большие языковые модели, LLM, агентная архитектура, инструментальное взаимодействие, верификация, локальные навыки.

AGENTIC ORCHESTRATION OF LOCAL VLA POLICIES IN LONG-HORIZON ROBOTIC MANIPULATION TASKS

Rashitov I.N.

Graduate Student

Moscow Technical University of Communications and Informatics

[1] Научный руководитель – к.т.н., доцент Воронов В.И.

Russia, Moscow

Abstract. The article examines an agentic architecture for long-horizon robotic manipulation, in which an LLM/VLM acts as an orchestrator, while a local VLA policy is used as an executable skill. It is shown that monolithic VLA models are limited in long-horizon tasks due to error accumulation, partial scene observability, and the lack of independent result verification. A control scheme based on task decomposition, tool selection, local execution, verification, and replanning is substantiated.

Keywords: robotic manipulation, long-horizon control, Vision-Language-Action, VLA, large language models, LLM, agentic architecture, tool use, verification, local skills.

Введение

Развитие моделей класса «зрение – язык – действие» (Vision-Language-Action, VLA) позволило связать изображение рабочей сцены, естественно-языковую инструкцию и управляющее действие робота-манипулятора в едином контуре. Однако при переходе от одиночных операций к многошаговым сценариям монолитная VLA-архитектура сталкивается с ограничениями: ошибка на раннем этапе изменяет состояние сцены, влияет на последующие действия и может привести к каскадному сбою всего сценария.

Основная проблема заключается в том, что одна модель вынуждена одновременно интерпретировать инструкцию, выделять цель, учитывать историю выполнения, выбирать следующий шаг и генерировать моторные команды. Для задач с длинным горизонтом такая совмещенность недостаточна, поскольку требуется не только исполнение локального действия, но и проверка промежуточного результата, обновление состояния среды и перепланирование при неудаче.

Цель и задачи исследования

Цель статьи состоит в обосновании агентной архитектуры многошагового управления, в которой большая языковая модель (Large Language Model, LLM) или визуально-языковая модель (Vision-Language Model, VLM) выполняет функции оркестратора, а компактная VLA-модель используется как локальный исполнительный навык. Объектом анализа являются архитектуры роботизированной манипуляции с естественно-языковым управлением, предметом – разделение функций между планированием, восприятием, VLA-исполнением и независимой верификацией результата.

Для достижения цели решаются задачи: выявить ограничения монолитных VLA-моделей при длинном горизонте задачи; сопоставить агентные подходы к управлению роботизированными действиями; описать гибридную архитектуру локальных навыков; определить роль проверки результата и перепланирования.

Научная новизна заключается в формулировании архитектурного критерия: чем сильнее успех задачи зависит от промежуточных состояний, тем более обоснованным становится выделение отдельного уровня оркестрации, обратной связи и восстановления. В такой модели VLA-политика рассматривается не как единый центр управления, а как параметризуемый исполнительный компонент внутри цикла «исполнение – проверка – перепланирование».

Основная часть

VLA-модель в общем виде преобразует визуальное наблюдение, языковую инструкцию и, при необходимости, проприоцептивное состояние робота в действие или последовательность действий. В архитектурах RT-2 и OpenVLA данная логика получила развитие в направлении обобщенных робототехнических политик: модель не просто распознает сцену, а связывает семантику инструкции с физическим действием робота [1,2]. Достоинство такого подхода состоит в уменьшении разрыва между языковым описанием задачи и

моторным исполнением. Вместо отдельного символического планировщика, набора правил и вручную заданных переходов используется обучаемая модель, способная переносить часть визуально-языковых обобщений на робототехнический домен. Современный обзор VLA-систем показывает, что развитие этой области связано с объединением восприятия, языка и управления в единой обучаемой архитектуре [12].

Однако в задачах с длинным горизонтом успех не равен качеству одного действия. Он определяется устойчивостью всей цепочки промежуточных переходов. Пусть исходная инструкция декомпозируется на последовательность подзадач $g_1, g_2, \ldots, g_n$, а результат каждой подзадачи проверяется функцией $V_i(s_i)$, где s_i – состояние сцены после выполнения i-го шага. Тогда успех полного сценария можно представить как конъюнкцию:

$$S = \bigwedge_{i=1}^{n} V_i\,(s_i) \qquad (1)$$

где $V_i(s_i)$ – функция проверки результата i -й подзадачи; s_i – состояние сцены после ее выполнения; n – число подзадач.

Формула отражает инженерное ограничение длинного горизонта: ошибка захвата, распознавания или размещения объекта изменяет состояние среды и влияет на последующие действия. Поэтому многошаговая манипуляция требует не только исполнительной политики, но и независимого механизма проверки промежуточных результатов.

В этом контексте длинный горизонт следует понимать не только как увеличение числа действий, но и как рост зависимости между промежуточными состояниями. Каждая подзадача изменяет рабочую сцену и формирует исходные условия для следующего шага. Если система не фиксирует фактический результат локального действия, то последующее планирование начинает опираться на ожидаемое, а не на действительное состояние среды. Для физических манипуляторов такое расхождение особенно существенно, поскольку ошибки контакта, проскальзывание объекта или частичная окклюзия

могут быть незаметны для исполнительной политики, но критичны для дальнейшего сценария.

Исследование Long-VLA прямо фиксирует, что многие VLA-фреймворки в основном ориентированы на задачи короткого горизонта, тогда как длинные сценарии ограничиваются зависимостью между подзадачами и сложностью связывания навыков [3]. Эта постановка не отрицает перспективность сквозных VLA-архитектур, но показывает границу их прямого применения. В длинном сценарии модель должна учитывать не только текущее изображение, но и фазу задачи, историю уже выполненных действий, ожидаемый результат предыдущего шага и условия перехода к следующему действию. Близкий вывод дает UniVLA: для длинных сценариев важны временная и причинная структура визуальных наблюдений, а не только языковая интерпретация команды [11].

Проблема усиливается в физических средах. В симуляции состояние может быть полностью наблюдаемым и дискретизированным. На реальном манипуляторе сцена частично наблюдаема: объект может быть закрыт захватом, находиться под неудачным ракурсом камеры или сместиться после контакта. Локальная моторная политика может успешно завершить траекторию с точки зрения внутреннего таймера, но не достичь требуемого физического результата. В монолитной архитектуре это создает риск продолжения сценария на основе недостигнутого состояния. Для гибридной системы это становится основанием для вызова внешнего инструмента проверки.

Именно поэтому VLA-модель в многошаговой архитектуре целесообразно трактовать как исполнительный слой. Она отвечает за преобразование локально сформулированной подзадачи в моторные действия, но не должна единолично определять, достигнуто ли новое состояние и какой шаг должен следовать далее. Такое разделение снижает когнитивную нагрузку на модель: высокоуровневый выбор, проверка и восстановление выносятся в агентный контур, а VLA-политика концентрируется на физически ограниченном действии.

Агентный подход к управлению манипуляцией основан на разделении рассуждения и физического исполнения. LLM или VLM не генерирует напрямую низкоуровневые команды робота, а выбирает инструмент, формирует параметры вызова и интерпретирует результат взаимодействия со средой. Для робототехники это принципиально, поскольку действие должно быть ограничено доступными навыками, геометрией сцены и текущим состоянием манипулятора.

В SayCan данная логика реализуется через выбор навыка с учетом не только языковой релевантности, но и физической выполнимости действия [4]. Inner Monologue развивает этот принцип за счет включения обратной связи после каждого шага: система получает описание сцены, результат проверки или сообщение оператора и может скорректировать дальнейший план [5]. В совокупности эти подходы показывают, что для многошаговой манипуляции важен не заранее заданный полный план, а замкнутый цикл исполнения, проверки и перепланирования.

Code as Policies и ProgPrompt ограничивают свободу LLM через исполняемые интерфейсы: модель генерирует код или план с учетом доступных действий, объектов и программных интерфейсов управления [6,7]. VoxPoser и RoboTool демонстрируют близкую модульную логику: высокоуровневая модель формирует ограничения, параметры или промежуточные представления, а физическое действие реализуется отдельными планировщиками и навыками [8,9]. ReAct задает общий принцип чередования рассуждения и действия, при котором обращение к среде становится способом уточнения состояния и снижения ошибки [10].

Сравнение указанных подходов показывает, что надежные робототехнические архитектуры не сводят управление к одному предсказанию. Они вводят явные интерфейсы между смысловой интерпретацией, восприятием,

выбором инструмента и физическим действием. Для VLA-систем это означает, что локальная политика должна получать не длинную исходную инструкцию, а уже подготовленную подзадачу с заданным объектом, целью и условиями проверки.

Таблица 1. Сравнение агентных подходов к управлению роботизированными действиями

Подход	**Функция LLM/VLM**	**Механизм снижения ошибки**	**Значение для многошагового управления**
SayCan	Выбор следующего навыка	Учет физической выполнимости действия	Связывает семантику инструкции с доступными действиями
Inner Monologue	Планирование с обратной связью	Проверка результата после шага	Поддерживает цикл «исполнение – проверка – перепланирование»
Code as Policies	Генерация программной политики	Вызов интерфейсов восприятия и управления	Переводит рассуждение в исполняемые процедуры
ProgPrompt	Генерация плана в программном формате	Учет списка доступных действий и объектов	Снижает риск формирования невозможных команд
VoxPoser	Построение ограничений и карт ценности	Разделение логического вывода и планирования движения	Показывает значимость промежуточных представлений
RoboTool	Многоэтапное инструментальное взаимодействие	Разделение анализа, планирования, расчета и генерации кода	Поддерживает задачи со сложными физическими ограничениями

Рассмотренные подходы различаются по способу связывания языковой модели с физическим действием, однако в них прослеживается общий архитектурный принцип: высокоуровневая модель не должна действовать в неограниченном пространстве текстовой генерации. Ее решения необходимо ограничивать набором доступных навыков, программных интерфейсов, результатов восприятия и процедур проверки. Для многошаговой манипуляции это особенно важно, поскольку ошибка планирования здесь не является изолированной: неверно выбранный навык или некорректно заданная цель изменяют состояние сцены и создают ошибочные условия для последующих действий. Поэтому агентная оркестрация должна рассматриваться не как

надстройка над VLA-политикой, а как механизм управления переходами между локальными состояниями задачи.

Для многошаговой манипуляции рациональна архитектура, в которой агентный уровень выполняет функции диспетчера, а исполнительный уровень представлен набором локальных навыков. В общем виде цикл включает наблюдение сцены, планирование, выбор инструмента, исполнение, проверку результата и перепланирование.

На рисунке 1 отражено разделение функций между уровнями системы. Подсистема восприятия обновляет состояние сцены; LLM/VLM-оркестратор декомпозирует задачу и выбирает инструмент; локальная VLA-политика выполняет физическое действие; контур верификации проверяет достижение результата. Возвратная стрелка от перепланирования к наблюдению показывает, что после каждого действия система должна актуализировать состояние среды, а не продолжать сценарий на основе прежнего контекста.

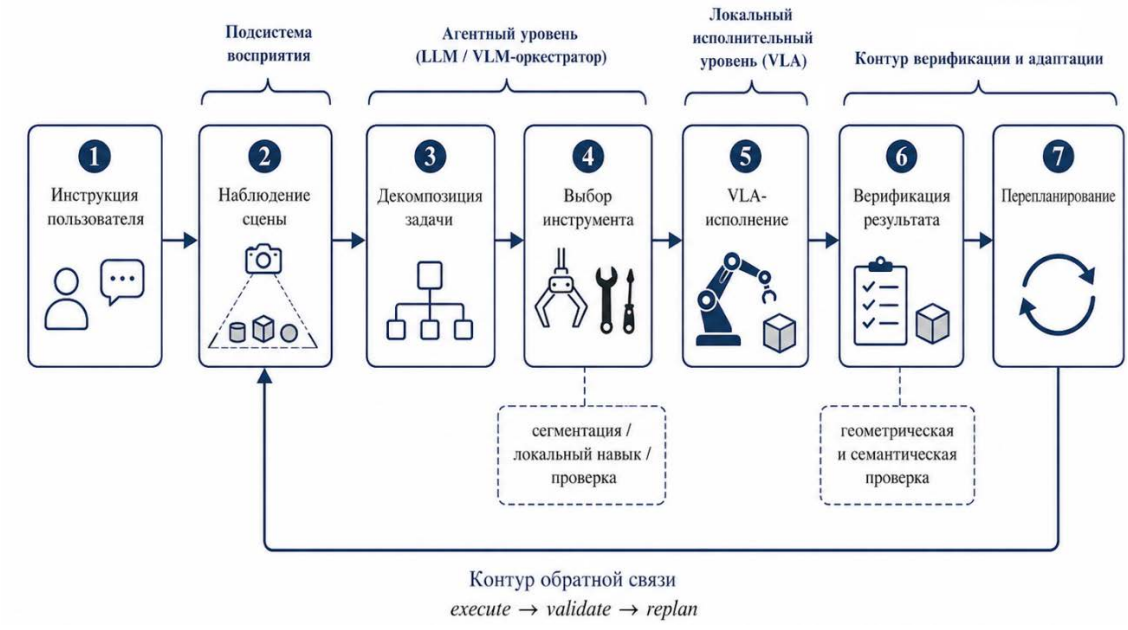

Рисунок 1. Агентный цикл многошагового управления с разделением восприятия, VLA-исполнения и независимой верификации

Примечание – разработано автором.

В такой архитектуре наблюдение сцены формирует текущее представление среды, планирование выделяет ближайшую подцель, выбор инструмента определяет необходимое действие системы: сегментацию, локализацию, запуск VLA-навыка или проверку результата. Исполнение передает управление локальной политике, а верификация определяет возможность перехода к следующему шагу. При неудаче агентный уровень инициирует повторное наблюдение, изменение подцели или ограниченное восстановление, не передавая принятие решения самой VLA-политике.

Принципиальное значение имеет архитектурное отделение исполнения от проверки. VLA-политика отвечает за локальное физическое действие, но не определяет успешность всего сценария. Независимая верификация может включать пространственную проверку положения объекта и семантическую оценку результата, что снижает риск ложного продолжения плана после неудачного моторного эпизода.

Для агентного уровня ключевыми являются пять критериев: атомарность подзадачи, наблюдаемость состояния, верифицируемость результата, восстанавливаемость и ограниченность автономии. Такая модель делает управление диагностируемым: ошибка локализуется в конкретном шаге, а не растворяется внутри длинной последовательности действий. При явном выделении агентного контура VLA-политика остается исполнительным навыком, а управление состояниями, ошибками и переходами переносится на уровень оркестрации.

Результаты

VLA-модели целесообразно рассматривать прежде всего как исполнительные политики для локальных визуомоторных действий. Их использование в качестве единого механизма управления длинными сценариями ограничено накоплением ошибок, частичной наблюдаемостью среды и отсутствием независимой проверки промежуточных состояний.

Заключение

Гибридная архитектура с LLM/VLM-оркестратором позволяет разделить языковое планирование, восприятие, моторное исполнение и верификацию. Для многошаговой манипуляции наиболее обоснована схема, в которой VLA-политика выполняет атомарные действия, а агентный уровень управляет их последовательностью, проверяет результат и инициирует восстановление при ошибке. Такая организация повышает диагностируемость системы, ограничивает распространение локальных ошибок и позволяет модифицировать отдельные компоненты без полного переобучения всего контура управления.

Библиографический список

1. Brohan A. et al. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control // Proceedings of the 7th Conference on Robot Learning. – PMLR, 2023.
2. Kim M. J. et al. OpenVLA: An Open-Source Vision-Language-Action Model // arXiv preprint. – 2024. – arXiv:2406.09246. – DOI: 10.48550/arXiv.2406.09246.
3. Fan Y. et al. Long-VLA: Unleashing Long-Horizon Capability of Vision Language Action Model for Robot Manipulation // arXiv preprint. – 2025. – arXiv:2508.19958. – DOI: 10.48550/arXiv.2508.19958.
4. Ahn M. et al. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances // arXiv preprint. – 2022. – arXiv:2204.01691. – DOI: 10.48550/arXiv.2204.01691.
5. Huang W. et al. Inner Monologue: Embodied Reasoning through Planning with Language Models // arXiv preprint. – 2022. – arXiv:2207.05608. – DOI: 10.48550/arXiv.2207.05608.

6. Liang J. et al. Code as Policies: Language Model Programs for Embodied Control // arXiv preprint. – 2022. – arXiv:2209.07753. – DOI: 10.48550/arXiv.2209.07753.
7. Singh I. et al. ProgPrompt: Generating Situated Robot Task Plans using Large Language Models // arXiv preprint. – 2022. – arXiv:2209.11302. – DOI: 10.48550/arXiv.2209.11302.
8. Huang W. et al. VoxPoser: Composable 3D Value Maps for Robotic Manipulation with Language Models // arXiv preprint. – 2023. – arXiv:2307.05973. – DOI: 10.48550/arXiv.2307.05973.
9. Xu M. et al. Creative Robot Tool Use with Large Language Models // arXiv preprint. – 2023. – arXiv:2310.13065. – DOI: 10.48550/arXiv.2310.13065.
10. Yao S. et al. ReAct: Synergizing Reasoning and Acting in Language Models // Proceedings of the International Conference on Learning Representations. – 2023.
11. Wang Y. et al. Unified Vision-Language-Action Model // arXiv preprint. – 2025. – arXiv:2506.19850. – DOI: 10.48550/arXiv.2506.19850.
12. Ud Din M. et al. Vision Language Action Models in Robotic Manipulation: A Systematic Review // arXiv preprint. – 2025. – arXiv:2507.10672. – DOI: 10.48550/arXiv.2507.10672.