

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
МОСКОВСКИЙ ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ  
(национальный исследовательский университет)  
ФИЗТЕХ-ШКОЛА ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ  
МАГИСТЕРСКАЯ ПРОГРАММА  
«МЕТОДЫ И ТЕХНОЛОГИИ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА»

Ивченков Ярослав Павлович

**Объектно-центричное представление мира агента  
в обучении с подкреплением**

03.04.01 — Прикладные математика и физика

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

**Научный руководитель:**  
д. т. н  
Матвеев Иван Алексеевич

Москва  
2022 г.

## Содержание

<b>1 Введение</b>	<b>4</b>
<b>2 Обзорно-постановочный раздел работы</b>	<b>6</b>
2.1 Основные понятия и определения . . . . .	6
2.2 Обзор современного состояния проблемы . . . . .	9
2.2.1 Модельный подход в обучении с подкреплением . . . . .	9
2.2.2 Алгоритмы объектно-центричного подхода . . . . .	12
2.3 Формальная постановка задачи . . . . .	16
<b>3 Факторизованная модель мира</b>	<b>19</b>
3.1 Семейство моделей прямого влияния . . . . .	20
3.2 Семейство моделей косвенного влияния . . . . .	21
3.2.1 Однозадачный вектор влияния . . . . .	24
3.2.2 Многозадачный вектор влияния . . . . .	25
3.3 Архитектуры моделей . . . . .	25
<b>4 Вычислительные эксперименты</b>	<b>27</b>
4.1 Эксперименты в наборе сред MetaWorld . . . . .	27
4.2 Эксперименты в наборе сред CasualWorld . . . . .	29
<b>5 Заключение</b>	<b>33</b>

### **Аннотация**

В данной работе рассматриваются задача улучшения обобщающей способности алгоритмов модельного подхода в обучении с подкреплением, решающих задачу визуального роботизированного контроля. Предложены различные объектно-центричные модели мира агентов, учитывающие причинно-следственные связи между манипулятором и объектом в среде. Рассматривается семейство моделей, выделяющее в явном виде влияние манипулятора на объект и проанализированы их качества и потенциал для разных сред. Для оценки качества предложенных алгоритмов проведены эксперименты в двух наборах сред, CasualWorld и MetaWorld. Полученные результаты позволяют утверждать, что рассмотренные модели мира позволяют добиться большей обобщающей способности, чем базовый алгоритм модельного подхода.

## 1 Введение

Обучение с подкреплением является одним из многообещающих направлений изучения искусственного интеллекта. Данная область науки занимается задачами последовательного принятия решений. Достаточно большое количество задач в реальной жизни подходит под это описание, например, контроль действий манипулятора, управление беспилотным автомобилем или компьютерные игры. В каждой из подобных задач можно выделить две основные сущности, взаимодействующие друг с другом - агента и среду. Агент может получать от среды информацию различного типа на каждом шаге - например, вектор позиций конечностей манипулятора или картинку с камеры автомобиля. Как и в остальных областях машинного обучения, работа с визуальными данными требует особых решений. В последние годы были достигнуты значительные успехи в решении подобного рода задач - в частности, в 2015 году были достигнуты результаты, сравнимые с человеческими в играх *Atari*, а позже - побиты результаты игры в *Starcraft 2* и *Dota 2*. Несмотря на подобные успехи, применение алгоритмов обучения с подкреплением в реальной жизни достаточно ограничено по причине имеющихся недостатков большинства подходов. Для алгоритмов RL свойственно требовать очень большое количество взаимодействия со средой, они обладают плохой обобщающей способностью а также плохо интерпретируются. Обобщающая способность является чрезвычайно важным свойством моделей, от которого напрямую зависит возможность применения алгоритма для реальных задач, а не только в искусственных средах.

Одним из способов преодолеть данные недостатки является построение модели среды, с которой взаимодействует агент. Улучшение качества модели среды позволяет алгоритму проще адаптироваться к новым задачам в той же среде. Однако без специальных модификаций, направленных на улучшение обобщающих способностей, данные алгоритмы все равно показывают слабые обобщающие способности.

За последние годы было предложено много методов улучшения обобщающих способностей алгоритмов в этой области, доминирующим направлением в них является предобучение алгоритма с целью максимизации определенного внутреннего информационного показателя качества. Максимизируя их, алгоритм более эффективно исследует среду и выучивает более богатую и устойчивую модель среды, что позволяет ему избежать переобучения для определенной функции награды. Данный подход - предтренировка алгоритма - был успешно адаптирован из другой области

машинного обучения, машинного зрения.

Во многих задачах, рассматриваемых в машинном зрении, таких как генерация сцен и генерация видео, в последние годы успешно применяется объектно-центричный подход, заключающийся в выделении на изображении отдельных сущностей. Введение подобного рода структуры позволяет упростить моделирование отдельных компонент изображения и сделать этот процесс более интерпретабельным. Также данный подход выглядит весьма уместным и перспективным в случаях, когда объекты зависят друг от друга, позволяя моделировать причинно-следственные связи.

В обучении с подкреплением объектно-центричный подход известен достаточно давно, однако исследований в данном направлении велось достаточно мало. Несмотря на то, что не во всех задачах можно с успехом выделить отдельные объектные сущности, в задачах роботизированного контроля переход к подобного рода абстракциям достаточно тривиален. В среде подразумевается присутствие манипулятора и, опционально, некоторого количества объектов, с которыми манипулятор может взаимодействовать.

Модели мира могут получать большие преимущества от переиспользования смоделированной или заранее заданной структуры среды [6]. В связи с этим представляется достаточно перспективным внедрение объектно-центричного подхода в алгоритмах, основанных на модели среды, решающих задачи роботизированного контроля, в которых агент получает в качестве наблюдений изображения среды. Тогда как его влияние на скорость обучения в решении конкретных задач не очевидно, обобщающие способности алгоритмов должны возрасти за счет отдельного моделирования динамики объектов на изображении.

## 2 Обзорно-постановочный раздел работы

### 2.1 Основные понятия и определения

Обучение с подкреплением занимается задачами последовательного принятия решений. Данная задача в большинстве случаев формализуется в терминах Марковского Процесса Принятия Решений (МППР), определяющегося следующим образом:

**Определение 2.1.** Марковский Процесс Принятия Решений (МППР) задается кортежем из четырех элементов  $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, \mathcal{R} \rangle$ , где:

- $\mathcal{S}$  - набор состояний;
- $\mathcal{A}$  - набор действий;
- $P(s_{t+1} | s_t, a_t)$  - функция перехода состояний среды;
- $\mathcal{R}(s_t, a_t, s_{t+1})$  - функция вознаграждения среды;

В работе рассматривается случай конечного МППР с вероятностной функцией перехода состояний. Состояние и действие предполагаются конечномерными векторами  $s_t \in \mathbb{R}^{d_s}$ ,  $a_t \in \mathbb{R}^{d_a}$ .

Поведение агента определяется действиями, совершаемыми им в тех или иных состояниях среды. Агент принимает их в соответствии со *стратегией*  $\pi(a | s)$ , представляющей собой функцию, ставящую в соответствие каждому состоянию среды  $s_t$  действие, которое агент должен предпринять. В общем виде эта функция имеет вид вероятностного распределения над множеством действий.

Для полного определения задачи обучения с подкреплением также необходимо задать оптимизируемый функционал. Вводится понятие *коэффициента дисконтирования*  $\gamma \in [0, 1]$ , отвечающего за уменьшение текущей ценности награды, которую агент может получить в будущем. Используя это понятие, формулируется следующее:

**Определение 2.2.** Отдача  $R(\tau) = \sum_{t=0}^T \gamma^t r_t$  - дисконтированная сумма вознаграждений на определенной траектории  $\tau$ .

Отдача является случайной величиной, поскольку и выбор действий агентом, и функция перехода состояний среды могут иметь случайный характер в общем случае.

Таким образом, от стратегии агента ожидается, что матожидание отдачи по всем траекториям, которые мы можем получить, будет максимальным. Формально это выражается следующим образом:

**Определение 2.3.** Целевая функция, определяемая как  $G(\pi, \mathcal{M}) = \mathbb{E}_{\tau \sim \pi} [R(\tau)] = \mathbb{E}_\pi \sum_t \gamma^t r_t$ , есть математическое ожидание отдачи за эпизод, полученный при помощи использования агентом стратегии  $\pi$ .

Задача агента в обучении с подкреплением в общем случае - максимизировать целевую функцию  $G(\pi, \mathcal{M})$  путем изменения стратегии  $\pi$ . Для упрощения задачи вводится функция полезности  $V^\pi(s) = \mathbb{E}_{s_o=s, \tau \sim \pi} [R(\tau)] = \mathbb{E}_{s_o=s, \tau \sim \pi} \left[ \sum_{t=0}^T \gamma^t r_t \right]$ , означающая ожидаемую отдачу в эпизоде, если агент следует стратегии  $\pi$  и стартует из состояния  $s$ . Алгоритмы, фокусирующиеся на модели среды, также пытаются аппроксимировать истинную функцию перехода состояний  $P(s_{t+1} | s_t, a_t)$  своей собственной моделью среды  $P(s' | s, a)$ . Также возможно моделирование функции награды  $R(s', a, s)$  различного вида. Совокупность моделей, использующихся алгоритмом, называется моделью мира.

В постановке задачи, использующей МППР, подразумевается, что получаемые агентом состояния  $s_t^{agent}$  и состояния среды  $s_t^{env}$  одинаковы, то есть агент имеет доступ к истинному состоянию среды. Однако в большинстве ситуаций, в особенности практических, получить такое состояние затруднительно либо вообще невозможно, например, когда агент получает в качестве наблюдения изображение с камеры. Даный факт вынуждает разделить понятия наблюдений, получаемых агентом, и состояний среды, что приводит к новой обобщенной формулировке:

**Определение 2.4.** Частично Наблюдаемый Марковский Процесс Принятия Решений задается кортежем из шести элементов  $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, \mathcal{R}, \Omega, \mathcal{O} \rangle$ , где:

- $\mathcal{S}$  - набор состояний;
- $\mathcal{A}$  - набор действий;
- $P(s_{t+1} | s_t, a_t)$  - функция перехода состояний среды;
- $\mathcal{R}(s_t, a_t, s_{t+1})$  - функция вознаграждения среды;
- $\Omega(o_t | s_t)$  - функция наблюдений;
- $\mathcal{O}$  - множество наблюдений.

Предполагается, что состояние  $s_t$  является функцией от истории эпизода  $H_t = a_0, o_1, a_1, \dots, a_{t-1}, o_t$ , такой, что  $s_t = f(H_t)$  сохраняет всю информацию эпизода, необходимую для максимально возможно точного предсказания  $a_{t+1}$ . Определенное таким образом состояние  $s_t$  сохраняет Марковское свойство, следовательно, может использоваться для получения всех результатов для стандартного МППР. Поскольку функция  $f$  должна быть как можно менее дорогой в вычислительном плане, её ищут в виде:

$$s_{t+1} = p(s_t, a_t, o_{t+1}) \quad (2.1)$$

Алгоритмы модельного подхода, в основе которых лежит частично наблюдаемый МППР, аппроксимируют как функцию перехода внутренних состояний среды вида  $P(s' | s, a)$ , так и функцию вида 2.1.

Для измерения обобщающей способности алгоритмов удобно использовать понятие контекстного МППР [7]:

**Определение 2.5.** Контекстный МППР задается кортежем из четырех элементов  $\langle \mathcal{S}, \mathcal{A}, \mathcal{M}(c) \rangle$ , где:

- $\mathcal{C}$  - множество контекстов;
- $\mathcal{S}$  - множество состояний;
- $\mathcal{A}$  - множество действий;
- $\mathcal{M}$  - отображение из множества контекстов  $c \in \mathcal{C}$  в МППР  $\mathcal{M}(c) = (\mathcal{S}, \mathcal{A}, p^c(y | x, a), r^c(x), \pi_0^c)$

Контекст является скрытой переменной, определяющей конкретную задачу в контекстном МППР. Каждое значение контекста определяет состояния, которые могут быть встречены в эпизоде. Отдельные части среды могут быть параметризованы значением произвольного вектора, например, значение массы какого-либо объекта. В этом случае, масса может быть включена в значение контекста, либо может считаться частью стохастичности среды.

Объектно-центричный подход в обучении с подкреплением не имеет устоявшейся теоретической базы. Общей чертой является разделение состояния  $s_t$  на кортеж из нескольких состояний  $(s_t^1, s_t^2, \dots, s_t^n)$ , каждое из которых соответствует состоянию отдельной объектной сущности. В данной работе рассмотрен именно этот случай, без дополнительного разделения пространства действий.

Возможность комбинирования объектно-ориентированного и модельного подходов заключается в возможности факторизации функций перехода состояний. В общем случае для этого используются графовые сети, однако для более простых случаев, например, взаимодействия манипулятора и единственного объекта, можно искусственно ввести причинно-следственные связи в модель мира. Благодаря этому ожидается прирост обобщающей способности алгоритма и увеличение интерпретабельности.

## 2.2 Обзор современного состояния проблемы

Поскольку работа представляет собой применение идей объектно-центричного подхода к алгоритмам обучения с подкреплением модельного подхода, будут описаны современные подходы в обоих областях. Особое внимание уделено описанию моделей мира в работах объектно-центричного подхода.

### 2.2.1 Модельный подход в обучении с подкреплением

Серия работ [10, 3, 12] является одной из наиболее заметных за последние годы в модельном подходе. Первой в серии статей является [10], в которой предложена новая модель мира RSSM (Recurrent State-Space Model). Марковские состояния моделируются при помощи апостериорного распределения следующего вида:

$$q(s_{1:T} | o_{1:T}, a_{1:T}) = \prod_{t=1}^T q(s_t | s_{t-1}, a_{t-1}, o_t) \quad (2.2)$$

По имеющимся действиям  $a_{1:T}$  и наблюдениям  $o_{1:T}$  вариационный автокодировщик  $q$  выводит скрытые состояния  $s_{1:T}$ . Имплементация  $q(s_t | s_{t-1}, a_{t-1}, o_t)$  выдает параметры для гауссовского распределения, случайной реализацией которого является следующее состояние  $s_t$ . Используя полученный автокодировщик, авторы выводят нижнюю вариационную границу для правдоподобия наблюдений:

$$\begin{aligned} & \ln p(o_{1:T}, r_{1:T} | a_{1:T}) \\ & \triangleq \ln \int \prod_t p(s_t | s_{t-1}, a_{t-1}) p(o_t | s_t) p(r_t | s_t) ds_{1:T} \\ & \geq \sum_{t=1}^T \left( \mathbb{E}_{q(s_t | o_{\leq t}, a_{<t})} [\ln p(o_t | s_t) + \ln p(r_t | s_t)] \rightleftharpoons \right. \\ & \quad \left. - \mathbb{E} [\text{KL}[q(s_t | o_{\leq t}, a_{<t}) \| p(s_t | s_{t-1}, a_{t-1})]] \right). \end{aligned} \quad (2.3)$$

Наибольший интерес представляет анализ различных архитектур состояний и функций, при помощи которых организовано взаимодействие между ними. Авторы находят, что реализация функции переходов состояний только при помощи вероятностных распределений не является оптимальным решением. Из-за вероятностного характера функции переходов, запоминание информации, произошедшей некоторое количество шагов назад, не является простой задачей. Теоретически, модель может имитировать поведение детерминированной функции, однако выучить параметры таких распределений может быть проблематичным. С другой стороны, детерминированная функция переходов, хотя и решает проблемы вероятностной, не учитывает возможной стохастики среды.

Учитывая все вышеперечисленное, авторы приходят к выводу об уместности разделения состояния на две части, одна из которых,  $h_t$ , генерируется при помощи детерминированной функции, а другая,  $z_t$ , является параметризованным распределением:

$$\begin{aligned}
 \text{Детерминированная модель переходов:} & \quad h_t = f_\varphi(h_{t-1}, z_{t-1}, a_{t-1}) \\
 \text{Вывод стохастической части:} & \quad z_t \sim q_\varphi(z_t | h_t, o_t) \\
 \text{Генерация стохастической части:} & \quad \hat{z}_t \sim p_\varphi(\hat{z}_t | h_t) \\
 \text{Декодер изображения:} & \quad \hat{o}_t \sim p_\varphi(\hat{o}_t | h_t, z_t) \\
 \text{Декодер награды:} & \quad \hat{r}_t \sim p_\varphi(\hat{r}_t | h_t, z_t) \quad (2.4)
 \end{aligned}$$

где функция  $f_\varphi(h_{t-1}, z_{t-1}, a_{t-1})$  является рекуррентной нейронной сетью. Можно рассматривать  $h_t$  и  $z_t$  как части состояния среды, ответственные за динамику и наблюдения соответственно, поскольку во многих задачах динамика имеет более детерминированный характер, а наблюдения обладают явно выраженной стохастикой. Авторы подчеркивают, что при выводе состояния  $z_t$  необходимо получить это состояние как реализацию случайной величины из полученного распределения, чтобы не допустить детерминированного прохода информации от наблюдения  $o_t$  к реконструкции  $\hat{o}_t$ . Эксперименты демонстрируют, что и стохастическая часть состояния  $z_t$ , и детерминированная часть  $h_t$  важны для успешного решения задач, в частности, без  $z_t$  процесс обучения модели не приносит результатов.

В следующей статье [3] авторы предлагают один из сильнейших современных модельных алгоритмов, Dreamer, в основе которого лежит предложенная в прошлой статье модель мира RSSM. Главное наблюдение, легшее в основу работы, состоит в

возможности проведения эффективной в вычислительном плане тренировки стратегии агента на траекториях из скрытых марковских состояний  $s_t$ , полученных исключительно из модели мира без взаимодействия со средой. Процесс обучения на сгенерированных при помощи модели переходов состояний траекториях авторы называют обучением в “воображении” (imagination). Тренировка всех частей агента происходит следующим образом:

- Взаимодействие со средой при использовании стратегии  $\pi(a_t | s_t)$
- Тренировка модели мира на траекториях, полученных из буфера опыта  $\mathcal{D}$
- Тренировка стратегии в “воображении” модели

При генерировании траекторий в пространстве скрытых марковских состояний, алгоритм не может получить никаких наблюдений, поэтому использует только модель переходов скрытых состояний  $p(s_t | s_{t-1}, a_{t-1})$  и стратегию  $\pi(a_t | s_t)$ . Модель переходов скрытых состояний обучается на этапе обучения модели мира как целого. Модель мира обучается, следуя принципам статей [19, 2], максимизируя вариационную нижнюю оценку взаимной информации:

$$\max I(s_{1:T}; (o_{1:T}, r_{1:T}) | a_{1:T}) - \beta I(s_{1:T}, i_{1:T} | a_{1:T}), \quad (2.5)$$

где  $\beta$  - действительный гиперпараметр, а  $i_t$  - индексы буфера данных, определяющие наблюдения следующим образом:  $p(o_t | i_t) \doteq \delta(o_t - \bar{o}_t)$ . Функционал, максимизируемый в процессе тренировки моделью мира, состоит из нескольких частей и выглядит следующим образом:

$$\mathcal{J}_{\text{REC}} \doteq E_p \left( \sum_t (\mathcal{J}_{\text{O}}^t + \mathcal{J}_{\text{R}}^t + \mathcal{J}_{\text{D}}^t) \right) + \text{const} \quad \mathcal{J}_{\text{O}}^t \doteq \ln q(o_t | s_t) \quad (2.6)$$

$$\mathcal{J}_{\text{R}}^t \doteq \ln q(r_t | s_t) \quad \mathcal{J}_{\text{D}}^t \doteq -\beta \text{KL}(p(s_t | s_{t-1}, a_{t-1}, o_t) \| q(s_t | s_{t-1}, a_{t-1})) \quad (2.7)$$

Он представляет собой видоизмененную версию нижней вариационной границы для наблюдения и наград из RSSM 2.3. Таким образом, на этапе тренировки модели мира, алгоритм получает наблюдения и награды и учится предсказывать их без использования изображений. Выученные таким способом модели используются для эффективной тренировки стратегии в “воображении”. Низкие вычислительные затраты на такую тренировку обеспечиваются низкой размерностью марковских состояний  $s_t$  по сравнению с наблюдениями  $o_t$ .

Поскольку при переходе к скрытым марковским состояниям  $s_t$ , мы переходим к обычному МППР, для тренировки стратегии в воображении можно применять традиционные алгоритмы обучения с подкреплением. В работе использован подход актора-критика [18], то есть происходит параллельное обучение следующих моделей:

$$\begin{aligned} \text{Актор:} & \quad a_\tau \sim \pi_\varphi(a_\tau \mid s_\tau) \\ \text{Критик:} & \quad v_\psi(s_\tau) \approx \mathbb{E}_{q(\cdot \mid s_\tau)} \left( \sum_{\tau=t}^{t+H} \gamma^{\tau-t} r_\tau \right) \end{aligned} \quad (2.8)$$

В последующей статье [12], предложена улучшенная версия алгоритма, содержащая в качестве  $s_t$  не Гауссово распределение, а категориальное. Авторы аргументируют выбор такого вида распределения более простой оптимизацией, возможностью более точного совпадения генеративного и апостериорного распределений и возможными особенностями среды Atari. Также предложена техника балансирования дивергенции Кульбака-Лейблера (KL balancing), позволяющая в процессе тренировки модели мира оптимизировать генеративное  $p_\varphi(\hat{z}_t \mid h_t)$  и апостериорное  $q_\varphi(z_t \mid h_t, o_t)$  распределения с разной эффективностью. Данное изменение мотивировано тем, что обучение генеративного распределения хорошего качества может быть достаточно долгим процессом, в течение которого априорное распределение будет оптимизироваться в сторону совпадения с генеративным распределением плохого качества. Следуя этой логике, авторы предлагают увеличить шаг обучения относительно генеративного распределения. Также предложены изменения в модели актора-критика, не представляющие большого интереса для данной работы.

### 2.2.2 Алгоритмы объектно-центричного подхода

В ряде работ в области поиска представления предложена идея структурировать переменную, отвечающую за представление изображения. Более формально, следуя идеям, предложенным в [8], рассматривается датасет  $\mathbf{X} = \{\mathbf{x}^{(i)}\}_{i=1}^N$ , состоящий из  $N$  независимых и одинаково распределенных сэмплов случайной величины  $\mathbf{x}$ . Предполагается, что также существует случайная величина  $\mathbf{z}$ , и данные сгенерированы, используя распределения  $p(\mathbf{z})$  и  $p(\mathbf{x} \mid \mathbf{z})$ . Реализации  $\mathbf{z}_i$  называются скрытыми переменными, и представляют собой сжатые представления изображений  $\mathbf{x}_i$ . В [8] предложены способы эффективной оценки некоторых распределений в данной задаче, используя вероятностную аппроксимацию апостериорного распределения  $q_\psi(\mathbf{z} \mid \mathbf{x})$ . Таким образом, переходя к рассмотрению объектно-центричных подходов к решению

данной задачи, их отличительной чертой является введение структуры в случайную величину  $\mathbf{z}$ . Наиболее часто встречаются подходы, где  $\mathbf{z}$  является кортежем из величин  $\mathbf{z}_i$ , каждая из которых соответствует за реализацию отдельного объекта на изображении. Процесс генерации  $\mathbf{x}$  по  $\mathbf{z}$  может отличаться.

В работе [14] предполагается независимость  $\mathbf{z}_k \in \mathbb{R}^M$ , при этом каждая из  $\mathbf{z}_k$  делает вклад в изображение  $\mathbf{x} \in \mathbb{R}^D$ . Количество объектных скрытых переменных  $K$  является гиперпараметром модели, представляющим верхнюю границу количества отдельных объектов на изображении. Генеративный механизм реализован в виде смеси гауссиан:

$$p(\mathbf{x}|\mathbf{z}) = \prod_{i=1}^D \sum_{k=1}^K m_{ik} \mathcal{N}(x_i; \mu_{ik}, \sigma^2) \quad (2.9)$$

Каждый пиксель  $x_i$ , таким образом, является вероятностной смесью различных объектных компонент  $\mu_{ik}$  с весами  $m_{ik}$ ,  $\sum_{k=1}^K m_{ik} = 1, \forall i = [1 \dots D]$ . Маска  $\mu_{ik}$  и среднее значение пикселя  $m_{ik}$  являются выходами декодера, принимающего на вход скрытую переменную  $\mathbf{z}_i$ . Распределение  $q(\mathbf{z}|\mathbf{x})$  является нормальным с параметрами  $\lambda = (\mu_{\mathbf{z}}, \sigma_{\mathbf{z}})$ . Алгоритм обучается путем минимизации нижней вариационной оценки обоснованности. Несмотря на достаточно большую долю вычислений, способных проводится параллельно, параметры распределения скрытой переменной обновляются при помощи механизма итеративного вывода (iterative inference), что делает алгоритм достаточно медленным.

В работе [15] представлен иной способ выделения объектов на изображении. В статье отказываются от подробного моделирования генеративного процесса изображения, предлагая модуль Slot-Attention, представляющий собой применение механизма “внимания” к признакам, полученным из свёрточной сети и случайно инициализированным объектным “слотам”. Полученные векторы подаются в рекуррентную нейронную сеть, обновляющую “слоты”. Для вывода проводят  $\sim 10$  итераций этого процесса. Полученные вектора, по сути являющиеся скрытыми объектными переменными  $\mathbf{z}_k$ , используются для декодирования. Архитектура декодера аналогична используемой в [14]. Алгоритм может быть адаптирован для широкого спектра задач, и для самой базовой, детектирования объектов на изображении, обучающий сигнал предоставляет исключительно ошибкой предсказания изображения. Принципиальное отличие упомянутых алгоритмов состоит в скорости работы: в [14], каждый шаг итеративного вывода сопряжен с вычислительно дорогой операцией предсказания изображения и вычисления ошибки в пространстве изображений, тогда как в [15]

все итерации проводятся исключительно в пространстве скрытых переменных, делая алгоритм значительно быстрее.

Введение дополнительной семантической структуры в изображение путем факторизации скрытой переменной позволяет авторам не только добиться улучшения различных показателей качества, но и производит качественное распутывание (“disentanglement”) представления изображения. Это качество проявляется в интерпретируемости отдельных компонент представления, и в объектно-центрических методах, где объектные компоненты отвечают за различные объекты, данное свойство явно проявляется. Однако необходимо подчеркнуть, что в методах, принимающих на вход единственное изображение, возможность выделить объект зависит преимущественно от датасета и полученные репрезентации по сути являются просто сегментациями, без возможности учета какой-либо причинности и в целом взаимосвязей между объектами. Для возможности учета таких факторов необходимо присутствие как минимум временного контекста. Его добавление приводит к постановке задачи предсказания видео, в котором также представлены алгоритма объектно-центрического подхода, например, [11, 17].

Дальнейший шаг в сторону обучения с подкреплением происходит путем добавления в задачу условных факторов, то есть действий. Промежуточную ступень между работами в области поиска представления и обучения с подкреплением представляют работы, рассматривающие задачу моделирования динамики среды. В отличие от обучения с подкреплением, в них отсутствует понятие награды, и от предложенных алгоритмов требуется только выучить модель среды. Присутствие нескольких объектных сущностей на изображении оставляет свободу в постановке задачи исследователям. В частности, сущности могут быть рассмотрены как части агента, взаимодействующие друг с другом и имеющие разделенные наборы действий. Подобная постановка задачи рассмотрена в [9] и формализуется при помощи введения факторизованных пространств скрытых состояний и действий следующим образом:

$$\mathcal{Z} = \mathcal{Z}_1 \times \mathcal{Z}_2 \times \dots \times \mathcal{Z}_K \text{ — пространство скрытых состояний;}$$

$$\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2 \times \dots \times \mathcal{A}_K \text{ — пространство действий;}$$

Для вывода объектных скрытых переменных из изображения, используется более простая по сравнению с уже описанными подходами модель, состоящая из двух частей: выделитель объектов  $E_{ext}$  и объектный энкодер  $E_{enc}$ . На первом этапе изображение подается в сверточную сеть с  $K$  фильтрами на последнем слое, которые

интерпретируются как объектные маски  $m^k$ . Далее, полученные выходы  $m^k$  кодируются полносвязной нейронной сетью  $E_{enc}$ , имеющей общие веса для всех  $m^k$ . Данный процесс описывается следующим образом:

$$\mathbf{z}^k = E_{enc}(m^k), \quad m^k = [E_{ext}(\mathbf{x})]_k$$

Взаимодействие между различными объектами и действиями, совершенными над ними, моделируется при помощи графовой нейронной сети (GNN), использующей в качестве связей между вершинами сообщения. Минимизируемый функционал представлен в виде адаптированной для объектной абстракции функции потерь из статьи [20].

В работе [4] авторы предполагают, что, хотя объекты необходимо моделировать как отдельные сущности, главный интерес представляют общие “правила” среды, например, законы физики, одинаково справедливые для всех объектов, присутствующих в среде. Моделирование подобных правил позволяет алгоритму проще адаптироваться к новым конфигурациям задач, поскольку законы физики продолжают выполняться для любого количества и комбинации объектов. В отличие от [9], пространство действий не является факторизованным. Генеративная модель наблюдений выглядит следующим образом:

$$p(X^{(0:T)}, Z_{1:K}^{(0:T)} | a^{(0:T-1)}) = p(Z_{1:K}^{(0)}) \prod_{t=1}^T p(Z_{1:K}^{(t)} | Z_{1:K}^{(t-1)}, a^{(t-1)}) \prod_{t=0}^T p(X^{(t)} | Z_{1:K}^{(t)}) \quad (2.10)$$

Модель мира  $P(s_{t+1} | s_t, a_t)$  декомпозирует сложную модель взаимодействия сущностей и действия на следующие составляющие:

- эффект действия  $a_t$  на каждую сущность
- взаимодействие сущностей между собой

Важной особенностью данной модели является использование для всех сущностей и пар сущностей одних и тех же функций, за счет чего авторы добиваются полной симметричности моделирования отношений объектов. Тогда как подобный подход позволяет добиться обобщения в комбинаторных задачах, в средах, подразумевающих несимметричность отношений объектов между собой, подобное моделирование может быть проблематичным. Например, моделирование взаимодействия достаточно тяжелого робота-агента и легкого кубика подразумевает почти полное пренебрежение влияния кубика на динамику робота. В средах, с сильно различающимися по своим

физическим свойствам объектами, выучить подобные законы взаимодействия может быть проще моделировать по отдельности.

Объектно-центричные алгоритмы обучения с подкреплением достаточно редки. Одним из них является ROLL [16], алгоритм, решающий задачу обусловленного задания (goal-conditioned) обучения с подкреплением, то есть алгоритм вместе с обычным набором наблюдений получает также изображение среды  $I_g$ , которое необходимо достичь. В отличие от предлагаемого в работе подхода, ROLL не требует истинной сегментации и способен самостоятельно сегментировать изображения. В алгоритме применяются два отдельных VAE: для сегментированных объектов и для изначального изображения. Также, в алгоритме применяется специальный формат тренировки для большей устойчивости к перекрытию объектами и манипулятором друг друга. Явно сформулированной модели мира в алгоритме не присутствует. Таким образом, ROLL в некотором роде дополняет предложенный в данной работе алгоритм, предлагаая способы сегментации и борьбы с перекрытиями объектов, что делает его ценным для изучения в дальнейшей работе.

Еще одним примером объектно-центричных алгоритмов является GATSBI [5]. Аналогично предыдущей работе, алгоритму не требуется доступа к истинным сегментациям и манипулятору, задний фон и объекты моделируются по-разному. Также, в отличие от предыдущей работы, взаимодействие между объектами моделируется при помощи графовой нейронной сети. Сегментируются различные объекты также разными способами, более подходящими для свойств конкретного типа объектов. Тренировка производится стандартным образом для моделей, работающих в постановке частично наблюдаемого МПР - при помощи максимизации нижней вариационной оценки правдоподобия. Тогда как подход, предложенный в работе демонстрирует хорошие результаты в средах с большим количеством различных простых объектов, ему может не хватить экспрессивности в случае сложного взаимодействия между единственным манипулятором и объектом со сложной динамикой.

## 2.3 Формальная постановка задачи

Ставится задача построения модели мира, обладающей структурированным скрытым состоянием и разделяющей модели переходов состояний различных объектов в среде с целью увеличения обобщающих способностей базового алгоритма при сохранении прочих показателей качества.

В среде предполагаются выполненными следующие условия задачи визуального роботизированного контроля:

- В среде присутствуют манипулятор и объект, с которым манипулятор может взаимодействовать
- Функция награды зависит и от манипулятора, и от объекта
- В качестве наблюдений алгоритм получает трехмерные изображения среды
- Агент управляет при помощи действий только манипулятором

Также предполагаются выполненными следующие дополнительные предположения:

- Имеется доступ к истинным маскам объекта и манипулятора на изображениях, получаемых из среды
- Имеется доступ к истинному вектору параметров конкретной задачи, называемому контекстом задачи  $c$ .

Задача моделируется в виде частично наблюдаемого МППР 2.4, однако нотация заимствует некоторые понятия из постановки задачи контекстного МППР 2.5, поскольку они являются легко адаптируемыми для рассматриваемого случая.

В качестве базового алгоритма используется Dreamer [3]. Для справедливого сравнения предлагаемых моделей с базовым алгоритмом предполагается возможным использование преимуществ, приобретаемых благодаря факторизации модели мира, однако изменения модели за рамками вышеупомянутых, как например использование сильно отличающегося способа тренировки стратегии (например, изменения алгоритма актора-критика, не связанные с объектной факторизацией), считаются недопустимыми.

Для сравнения эффективности обучения алгоритма и его обобщающих способностей предлагается использовать недисконтированную усредненную по эпизодам награду, полученную за эпизод:

$$\mathcal{R} = \mathbb{E}_{\tau \sim \pi} \left[ \sum_{t=0}^T r_t \right] \quad (2.11)$$

График значений математического ожидания по траекториям задач из тренировочного распределения служит оценкой качества обучения алгоритма. График, полученный на тестовых данных, служит оценкой обобщающей способности алгоритма.

Важно подчеркнуть, что алгоритм обучается только на тренировочных задачах, во время оценки качества работы на тестовых алгоритм не выполняет шагов оптимизации. Оценка графика проводится путем сравнения значений показателя качества в конце обучения.

### 3 Факторизованная модель мира

Типичные задачи визуального контроля состоят в том, чтобы совершить какое-либо действие над объектом при помощи манипулятора. Поскольку в формулировке задачи обучения с подкреплением при помощи МППР не присутствует в явном виде понятия объектов и сущностей, модели мира агентов модельного подхода в подавляющем большинстве случаев не учитывают структуры среды в алгоритме. Типичным примером такой модели среды служит модель среды алгоритма Dreamer [3], использованного в качестве базового алгоритма.

Тогда как в общем случае моделирование взаимодействий разумно производить, не выделяя заранее роли тех или иных сущностей, можно предположить, что в среде присутствует только манипулятор и объект (за исключением, возможно, жестко зафиксированных объектов, выполняющих роль обстановки). В этом случае становится возможным моделирование причинных отношений агента и объекта в явном виде при помощи структуры нейронной сети. Таким образом, в состоянии среды  $s_t$  выделяются две компоненты,  $s_t^r$  и  $s_t^o$ , соответствующие состоянию робота и состоянию объекта в момент времени  $t$ . Также предположим, что в среде имеется возможность получения истинных наблюдений манипулятора и объекта, то есть в качестве наблюдения выступает пара  $(o_t^r, o_t^o)$ . В дальнейшем предполагается, что в качестве наблюдений выступают сегментированные изображения среды, однако это не является строгим требованием.

Принципиальная разница между моделью переходов состояний  $s_t^r$  и  $s_t^o$  состоит в том, какое влияние на них оказывают действия, принимаемые агентом. В конкретном рассматриваемом случае визуального роботизированного контроля уместно предположить, что действия агента изменяют непосредственно *только* состояние манипулятора, а на состояние объекта влияют лишь косвенно. Таким образом, мы приходим к заключению, что исходная модель функции переходов состояний может оставаться неизменной для части манипулятора:

$$q(s_t | s_{t-1}, a_{t-1}) \quad (3.1)$$

Воздействие манипулятора на объект может быть смоделировано разными способами. Функция переходов состояний объекта должна быть обусловлена на переменную, содержащую информацию о состоянии манипулятора, поскольку предполагается, что именно манипулятор заставляет объект изменять свое состояние. Пред-

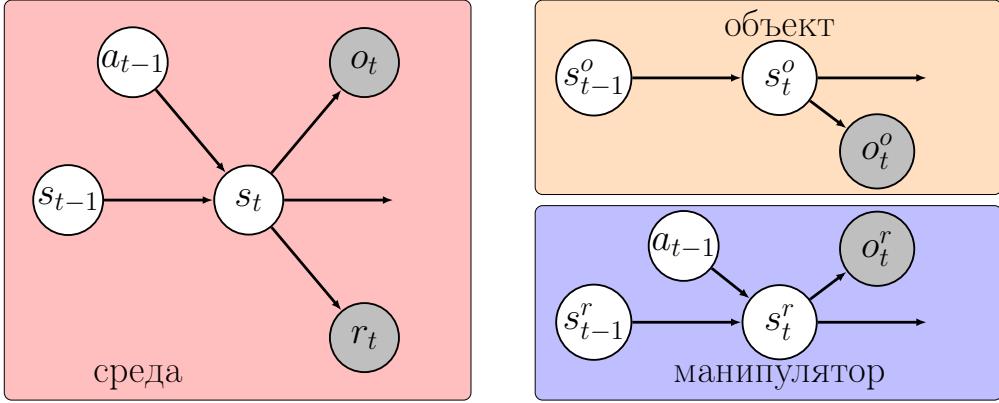


Рис. 1: Сравнение структур моделей переходов состояний алгоритма Dreamer [3] и предлагаемой объектно-центрической модели мира

ставляются возможными два варианта:

- Состояние  $s_t^o$  обусловлено на состояние  $s_t^r$  (прямого влияния)
- Состояние  $s_t^o$  обусловлено на вектор  $u_t$ , отражающий “влияние” манипулятора на объекта (косвенного влияния)

Модели первого класса содержат меньшее количество скрытых переменных, упрощая генеративную модель, однако также обладают меньшим потенциалом для улучшения обобщающей способности алгоритма, так как состояние  $s_t^r$  может содержать нерелевантную информацию для предсказания состояния  $s_t^o$ . Сравнение моделей мира базового алгоритма и предлагаемых моделей без спецификации того, каким образом манипулятор и объект взаимодействуют друг с другом, приведено на рисунке 1.

В работе рассмотрены возможные архитектуры из обоих семейств.

### 3.1 Семейство моделей прямого влияния

Простейшей моделью перехода состояний в этом семействе является:

модель объектной динамики:	$p(s_t^o   s_{t-1}^o, s_t^r)$
модель динамики робота:	$p(s_t^r   s_{t-1}^r, a_t)$
модель представления объекта:	$q(s_t^o   s_{t-1}^o, s_t^r, o_t^o)$
модель представления робота:	$q(s_t^r   s_{t-1}^r, a_{t-1}, o_t^r)$

(3.2)

Динамика робота и объекта в явном виде разделена на две части, при этом влияние объекта на робота вообще не учитывается. Влияние же робота на объект выражено

исключительно через  $s_t^r$ , что позволяет трактовать  $s_t^r$  как абстрактное действие робота по отношению к объекту. Подобное понятие весьма условно, поскольку в явном виде контроля над этими действиями не имеется.

Допущение незначительности влияния объекта на агента ограниченно выполняется для многих задач визуального контроля, однако в общем случае оно не является справедливым. Даже в простейших примерах, симулирующих условия реального мира, обязано выполняться ограничение, не позволяющее манипулятору и объекту пересекаться друг с другом. Для преодоления этих ограничений можно обусловить модель динамики робота также и на состояние объекта, однако при этом возникают проблемы с тренировкой, поскольку возникают циклы при попытке оценить апостериорную вероятность.

Следует упомянуть, что подобная генеративная модель не является единственной возможной в этом семействе. Можно также предположить, что обновленное состояние  $s_t^o$  зависит не от обновленного состояние робота  $s_t^r$ , а, скорее, от его предыдущего состояния  $s_{t-1}^r$ , что производит модель мира следующего вида:

$$\begin{aligned} \text{модель объектной динамики:} & p(s_t^o | s_{t-1}^o, s_{t-1}^r) \\ \text{модель динамики робота:} & p(s_t^r | s_{t-1}^r, a_t) \\ \text{модель презентации объекта:} & q(s_t^o | s_{t-1}^o, s_{t-1}^r, o_t^o) \\ \text{модель презентации робота:} & q(s_t^r | s_{t-1}^r, a_{t-1}, o_t^r) \end{aligned} \quad (3.3)$$

Отличие моделей может проявляться в уменьшенной ошибке, вызванной предположение о слабой обратной связи объекта и робота. Пусть в состоянии манипулятора  $s_{t-1}^r$  выполнялось данное предположение, но уже в следующем момент времени  $t$  оно не является справедливым. Состояние  $s_t^r$  будет получено с ошибкой, поскольку оно не обусловлено на  $s^o$ . В таком случае, в модели 3.2 ошибка будет накапливаться, поскольку обновленное состояние  $s_t^o$  обусловлено на состояние  $s_t^r$ , содержащее ошибку из-за не выполненного предположения в момент  $t$ . В модели 3.3 ошибка проявится в меньшем виде, поскольку в момент  $s_{t-1}^r$  предположение еще выполнено.

## 3.2 Семейство моделей косвенного влияния

Будем называть переменную, на которую обусловлено состояние объекта  $s_t^o$ , вектором влияния  $u_t$ . Если в семействе моделей прямого влияния предполагалось, что влияние манипулятора на объект представлено в виде состояния манипулято-

ра  $u_t = s_t^r$ , то в семействе моделей косвенного влияния вектор влияния  $u_t$  отделен от состояния робота. Поскольку вектор  $u_t$  представляет влияние манипулятора, то логично обусловить его на состояние  $s_t^r$ . Собирая вместе, получаем следующую генеративную модель:

модель объектной динамики:	$p(s_t^o   s_{t-1}^o, u_t)$
модель влияния:	$p(u_t   s_t^r)$
модель динамики робота:	$p(s_t^r   s_{t-1}^r, a_t)$

(3.4)

В зависимости от того, на что обусловлены априорное и апостериорное распределения вектора влияния, значение влияния в модели меняется. Представляются удобно трактуемыми и практически реализуемыми четыре варианта, представленные на Рисунке 2.

Первый вариант содержит следующие апостериорные распределения:

вывод объектного состояния:	$q(s_t^o   s_{t-1}^o, o_t^o)$
вывод вектора влияния:	$q(u_t   s_t^o)$
вывод состояния робота:	$q(s_t^r   s_{t-1}^r, a_t, u_t, o_t^r)$

(3.5)

В этом случае генеративный процесс и процесс вывода четко противопоставлены друг другу, протекая в обратных направлениях. Тогда как данный вариант выглядит достаточно разумно, у него присутствует ряд проблем:

- Апостериорное распределение для вектора влияния  $u_t$  плохо сопоставимо с генеративным. При проходе алгоритма снизу-вверх,  $u_t$  обусловлен только на состояние робота  $s_t^r$ , при этом состояние объекта может быть любым. Получив наблюдение изменившегося состояния объекта, попытки вывести, каким образом на объект повлиял манипулятор, выглядят странно, так как это подразумевало бы, что либо в  $s_t^o$  содержится информация о манипуляторе, либо в  $s_t^r$  содержится информация об объекте, чего хотелось бы избежать.
- Апостериорное состояние манипулятор  $s_t^r$  обусловлено на вектор влияния  $u_t$ . Данный факт сложно проинтерпретировать, также, учитывая тот факт, что  $u_t$  обусловлен на  $s_t^o$ , в состояние робота протекала бы информация от объекта, чего также хотелось бы избежать.

Второй вариант содержит процесс вывода, одинаково направленный с процессом генерации, формально записывающийся следующим образом:

$$\begin{aligned}
 &\text{вывод состояния робота:} & q(s_t^r \mid s_{t-1}^r, a_t, o_t^r) \\
 &\text{вывод вектора влияния:} & q(u_t \mid s_t^r) \\
 &\text{вывод объектного состояния:} & q(s_t^o \mid s_{t-1}^o, u_t, o_t^o)
 \end{aligned} \tag{3.6}$$

Тогда как такая архитектура выглядит странной для вывода вектора влияния, она может быть уместной для объектного состояния, поскольку в данном случае модели вывода и генерации объектной части представляют собой полную аналогию соответствующим моделям базового алгоритма. Подобное соответствие допускает трактовку  $u_t$  как уже упомянутого абстрактного действия для объектной части алгоритма.

В третьем варианте представлен фактор влияния, не обусловленный ничем на стадии вывода:

$$\begin{aligned}
 &\text{вывод состояния робота:} & q(s_t^r \mid s_{t-1}^r, a_t, u_t, o_t^r) \\
 &\text{вывод объектного состояния:} & q(s_t^o \mid s_{t-1}^o, u_t, o_t^o)
 \end{aligned} \tag{3.7}$$

Вектор влияния полностью теряет смысл, вложенный в него изначально.

В последнем варианте апостериорное распределение вектора влияния обусловлено и на объектное состояние, и на состояние манипулятора:

$$\begin{aligned}
 &\text{вывод объектного состояния:} & q(s_t^o \mid s_{t-1}^o, o_t^o) \\
 &\text{вывод состояния робота:} & q(s_t^r \mid s_{t-1}^r, a_t, o_t^r) \\
 &\text{вывод вектора влияния:} & q(u_t \mid s_t^r, s_t^o)
 \end{aligned} \tag{3.8}$$

Такой выбор апостериорного распределения мотивирует  $u_t$  содержать информацию, полезную для предсказания состояния объекта  $s_t^o$ , при этом используя информацию, содержащуюся в  $s_t^r$ . Таким образом, решается аналог обратной задачи динамики: по получившемуся состоянию  $s_t^o$  и воздействовавшему на него состоянию  $s_t^r$  мы хотим получить действие  $u_t$ .

Тогда как первый и четвертый варианты определения апостериорных распределений выглядят наиболее подходящими для использования в алгоритмах, в них присутствует недостаток, связанный с отсутствием информации об объекте в генеративном распределении  $p(u_t \mid s_t^r)$ . В зависимости от того, каким образом решается эта проблема, значение вектора влияния также меняется.

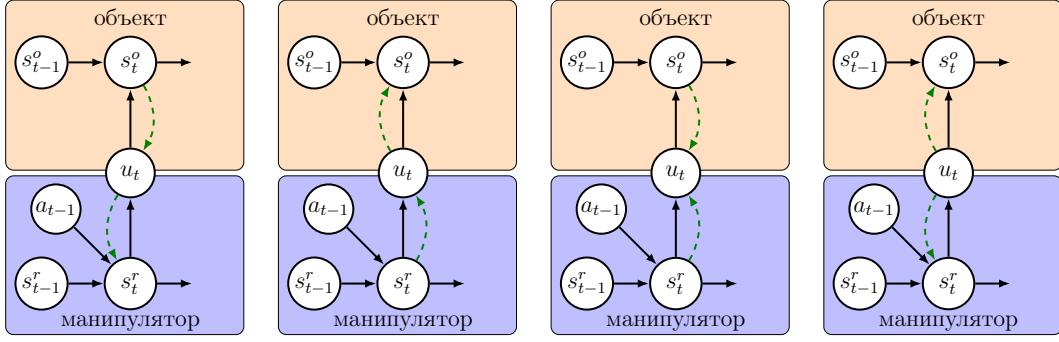


Рис. 2: Возможные виды зависимостей апостериорных распределений в модели неявного влияния

### 3.2.1 Однозадачный вектор влияния

Проблема в отсутствии информации об объекте появляется из-за разницы генеративного и апостериорного распределений для вектора влияния. Для того, чтобы генеративное распределение не слишком сильно отличалось от апостериорного, вектору влияния необходим дополнительный источник информации относительно того, где может находиться объект. То есть, вектор влияния должен зависеть не только от состояния манипулятора, но и от состояния объекта, в той или иной мере, чтобы более точно моделировать взаимодействие в генеративном процессе.

В некоторых средах, в частности в средах с малым количеством степеней свободы объекта, контекст задачи  $c \in \mathcal{C}$  может достаточно точно определять положение объекта в тот или иной момент времени. В предположении доступа к истинным значениям этих параметров для каждой задачи в такой среде, можно обусловить  $u_t$  на  $c$ , получая в результате следующее генеративное распределение для вектора влияния:

$$p(u_t | s_t^r, c) \quad (3.9)$$

Получаемый вектор влияния является обусловленным на контекст задачи, что дает возможность генерировать более точные траектории для каждой конкретной задачи при помощи модели среды. Введение такого распределения также мотивирует апостериорное распределение любого вида из упомянутых ранее выделять информацию из наблюдений, соответствующую отдельным параметрам задач. Также введение подобного внешнего фактора позволяет не вводить информации об объекте в состояние манипулятора. Получается архитектура, в которой модель перехода состояний манипулятора не учитывает информацию об объекте, но при этом функция перехода состояний объекта зависит от влияния манипулятора на объект в конкретной задаче.

Для усиления обучающего сигнала также предлагается подавать в модель функции награды только вектор влияния. Предсказание награды только по вектору влияния позволяет неявно интегрировать в него информацию об отношении манипулятора и робота. Без этого изменения вектор влияния может испытывать проблемы с тренировкой, поскольку единственный функционал, предоставляющий прямой обучающий сигнал для него - это дивергенция Кульбака-Лейблера между генеративным и апостериорным распределениями  $u_t$ .

Уместность введения подобного генеративного распределения полностью зависит от среды, в которой находится агент. При наличии возможности в течение одной задачи получать объектные наблюдения, также возможные для получения в других задачах, данный подход теряет свои преимущества, поскольку алгоритм больше не может определить примерное состояние объекта по информации о решаемой им задаче и состоянию манипулятора.

### 3.2.2 Многозадачный вектор влияния

Вместо того, чтобы пытаться придать вектору влияния  $u_t$  значение влияния манипулятора на объект в конкретном состоянии, можно рассмотреть вектор  $u_t$  как сжатое представление состояния  $s_t^r$ , содержащее информацию о потенциальных взаимодействиях на объекты в любых состояниях. Этого можно достигнуть, отказавшись от внедрения контекста задачи  $c$  в генеративное распределение для  $u_t$ . Тогда как для модели вывода 3.5 выглядит невозможным отказаться от контекста задачи, в 3.8 это выглядит реализуемым, поскольку генеративное и апостериорное распределение отличаются не настолько сильно. Пытаясь совместить генеративное и апостериорное распределение, мы пытаемся подстроиться к нашему набору задач, выучивая такое представление вектора влияния, которое было бы полезно вне зависимости от конкретной задачи. Однако эта модель испытывает больше проблем с тренировкой, поскольку предсказать награду по не зависящему от задачи или положения объекта вектору влияния  $u_t$  не представляется возможным.

## 3.3 Архитектуры моделей

Тогда как апостериорное распределение объектной части рассматривается в виде  $q(s_t^o | s_{t-1}^o, o_t^o)$ , такой вид распределения может проявлять себя не лучшим образом во время обучения алгоритма. В отличие от манипулятора, случайные действия, про-

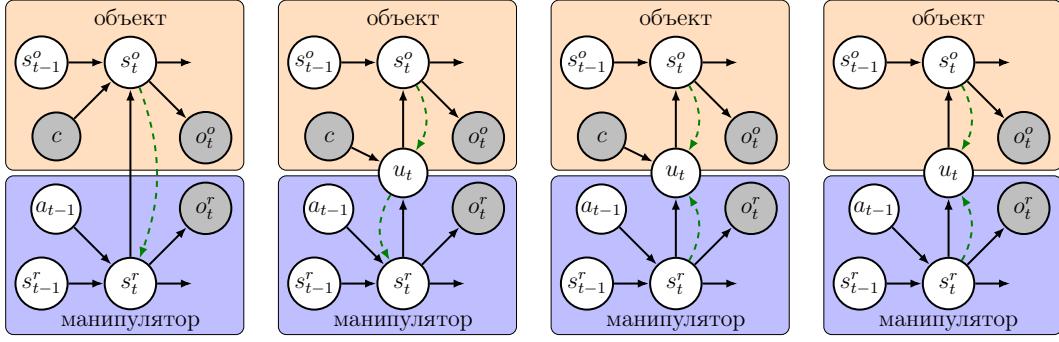


Рис. 3: Предлагаемые архитектуры

изводимые агентом в начале обучения, могут не затрагивать объект, тем самым не меняя его состояние в течение эпизода. Это может крайне негативно сказываться на обучении рекуррентной нейронной сети, подстраивающейся под предсказание одного и того же состояния. Для преодоления этого эффекта и одновременно улучшения его обобщающей способности предлагается упростить апостериорное распределение, приведя его к виду  $q(s_t^o | s_{t-1}^o)$ . Таким образом обеспечивается более четкий обучающий сигнал при изменении изображения, подающегося на вход.

Также, благодаря разделению скрытой переменной, появляется возможность разделить признаки, подающиеся в актора, критика и модель функции награды для предоставления более четкого сигнала при обучении тех или иных компонент алгоритма. В частности, вектор влияния лучше всего обучается при подаче в него только  $u_t$ , либо  $u_t$  и  $s_t^o$ .

Проверенные в ходе исследования модели алгоритмов изображены графически на рис. 3. Для удобства описания, алгоритмы, использующие факторизованные модели мира, приведены под названием СЕМА (Cause-Effect Modeling Agent). Таким образом, наиболее уместными выглядят следующие модели:

- **СЕМА-Direct:** Модель мира прямого влияния 3.2,  $\pi(a_t | s_t^r, s_t^o, c)$ ,  $\nu(s_t^r, s_t^o)$ ,  $r(s_t^o)$
- **СЕМА-TopDown:** Модель мира косвенного влияния 3.5, однозадачный вектор влияния,  $\pi(a_t | s_t^r, s_t^o, u_t)$ ,  $\nu(s_t^r, s_t^o, u_t)$ ,  $r(u_t)$
- **СЕМА-Influence:** Модель мира косвенного влияния 3.8, однозадачный вектор влияния,  $\pi(a_t | s_t^r, s_t^o, u_t)$ ,  $\nu(s_t^r, s_t^o, u_t)$ ,  $r(u_t)$
- **СЕМА-Generalize:** Модель мира косвенного влияния 3.8, многозадачный вектор влияния,  $\pi(a_t | s_t^r, s_t^o, u_t, c)$ ,  $\nu(s_t^r, s_t^o)$ ,  $r(s_t^o, u_t)$

## 4 Вычислительные эксперименты

Тестирование моделей мира проводилось в двух бенчмарках обучения с подкреплением, MetaWorld [13] и CasualWorld [1]. Оба пакета предоставляют инструментарий для разработки и тестирования сред и алгоритмов обучения с подкреплениями, решающих задачу роботизированного контроля.

### 4.1 Эксперименты в наборе сред MetaWorld

Библиотека сред MetaWorld [13] предоставляет широкие возможности для тестирования и сравнения алгоритмов мета-обучения и мультизадачного обучения и состоит из 50 задач, различающихся объектами, присутствующими в среде и целями, которые алгоритм должен достичь. Предлагаемые в работе архитектуры моделей мира в первую очередь рассчитаны на обобщение между задачами, имеющими одинаковую семантическую структуру, поэтому проверка алгоритма на всем наборе представляется неуместной. Также, некоторые из задач являются достаточно сложными даже для базового алгоритма, что представляло бы трудности в анализе обобщающей способности.

Для удобного тестирования предложенного подхода была создана новая среда, предлагаемое название которой RotatedDrawerOpen. В ней содержится два объекта - шкаф с выдвижным ящиком и робот. В конкретной задаче позиция шкафа закреплена, однако между задачами она меняется. Примеры изображений различных задач среды представлены на рисунке 4.

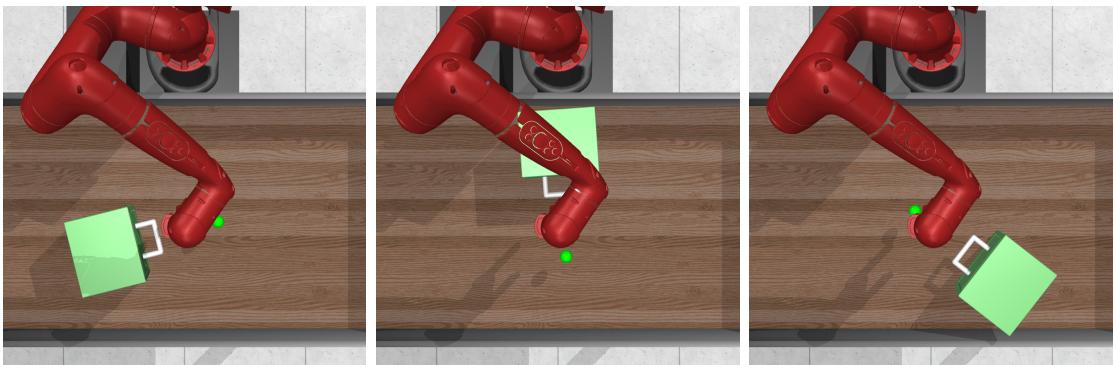


Рис. 4: Примеры задач в среде RotatedDrawerOpen.

Рассмотрим полярную систему координат с центром в середине сцены, в ней центр ящика имеет координаты  $(R, \alpha_\tau, h)$ . Для всех задач значения координат  $R$  и

$h$  одинаковы и не меняются ни в течения эпизода, ни между задачами. Множество задач, таким образом, совпадает с множеством значений  $\alpha_\tau$  и равно  $[0, 2\pi) \in \mathbb{R}$ . Контекст  $c$  в экспериментах в этой среде определяется следующим образом:

$$c = \alpha_\tau, \quad \forall \tau \in \mathcal{T} \quad (4.1)$$

Множество задач  $\mathcal{T}$  разделено на  $\mathcal{T}_{\text{train}}$  и  $\mathcal{T}_{\text{test}}$  следующим образом:

$$\begin{aligned} \mathcal{T}_{\text{train}} &= \{\tau \mid \alpha_\tau \in \left[0, \frac{\pi}{4}\right) \cup \left[\frac{3\pi}{4}, \frac{5\pi}{4}\right) \cup \left[\frac{7\pi}{4}, 2\pi\right)\}; \\ \mathcal{T}_{\text{test}} &= \{\tau \mid \alpha_\tau \in \left[\frac{\pi}{4}, \frac{3\pi}{4}\right) \cup \left[\frac{5\pi}{4}, \frac{7\pi}{4}\right)\}. \end{aligned} \quad (4.2)$$

Разделение такого вида выбрано для уменьшения эффекта ухудшения работы энкодера на изображениях с еще не виденной позицией шкафа. Действия  $a \in \mathbb{R}^4$  представляют собой координаты следующего положения руки робота и число, регулирующее сжатие его пальцев.

На шаге  $t$  агент получает из среды в качестве наблюдения робота  $o_t$  и наблюдения объекта  $o'_t$  сегментированные изображения робота и объекта соответственно. Изображения являются тензорами размера  $64 \times 64 \times 3$ , камера для всех задач зафиксирована в одном положении сверху сцены. Из-за подобного расположения камеры, сегментационная маска шкафа достаточно часто перекрыта изображением робота, в особенности в процессе выдвижения ящика. Тогда как подобная ситуация вписывается в постановку задачи и должна решаться алгоритмом за счет рекуррентного вывода  $q(s_{t+1}^o \mid s_t^o, u_t, o_{t+1}^o)$ , визуальные дефекты маски объекта могут мешать обучению объектной части модели мира, в особенности если вывод не рекуррентен  $q(s_{t+1}^o \mid o_{t+1}^o)$ . Чтобы исключить из анализа этот эффект, среда предоставляет в качестве объектного наблюдения  $o_t^o$  сегментационную маску объекта, полученную из виртуальной копии среды без робота. Функция награды не является разреженной и полностью соответствует функции награды из среды DrawerOpen пакета MetaWorld.

Подобная среда была выбрана для экспериментирования по нескольким причинам:

- Каждая отдельная задача в среде является достаточно простой для базового алгоритма, что позволяет сфокусироваться на улучшении обобщающей способности без необходимости проводить долгий процесс обучения и модификации алгоритма под задачу.

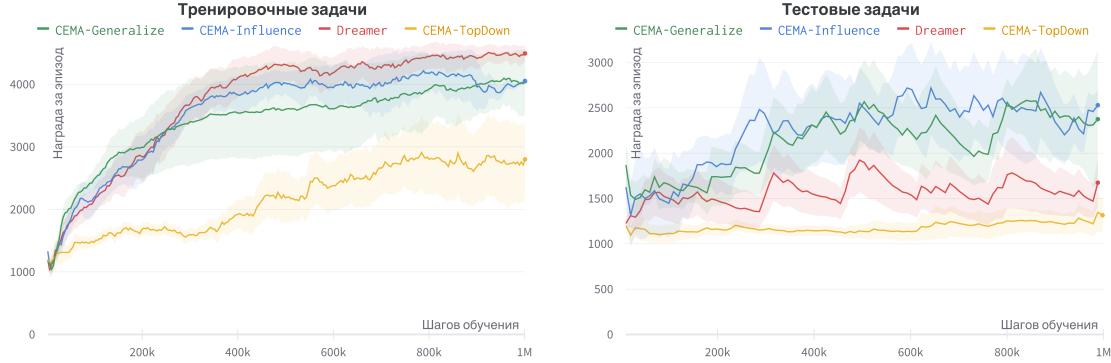


Рис. 5: Графики тренировочного процесса в среде RotatedDrawerOpen.

- Все задачи в среде имеют одну и ту же структуру, зависящую от единственного параметра вариации, что делает полученные результаты более визуализируемыми и трактуемыми.
- Структура задач в среде позволяет алгоритму получить достаточно информации из единственной камеры с закрепленным положением, что избавляет агента от необходимости обрабатывать сложные 3-D композиции положений объекта.

На рис. 5 представлены результаты обучения алгоритмов в среде RotatedDrawerOpen. Из результатов можно заключить, что опасения, высказанные насчет варианта алгоритма СЕМА-TopDown, были справедливы - даже на тренировочных задачах алгоритм показывает себя хуже остальных. Все предложенные варианты проигрывают базовому алгоритму по скорости обучения на тренировочных задачах. Однако важно заметить, что для вариантов СЕМА-Influence и СЕМА-Generalize этот проигрыш не является серьезным - награды в районе  $\sim 3000$  указывают на то, что алгоритм научился решать задачу, из чего следует что предложенные алгоритмы проигрывают только в скорости решения задач в конкретном эпизоде и при этом алгоритмы выходят по результатам обучения на схожую среднюю награду за эпизод. Однако на тестовых задачах, два упомянутых алгоритма уверенно превосходят базовый, хоть их результаты и не являются устойчивыми.

## 4.2 Эксперименты в наборе сред CasualWorld

В наборе сред CasualWorld[1] представлен удобный инструментарий для оценки обобщающих способностей алгоритмов обучения с подкреплением. Главным преимуществом по сравнению с другими библиотеками для тестирования является удобный

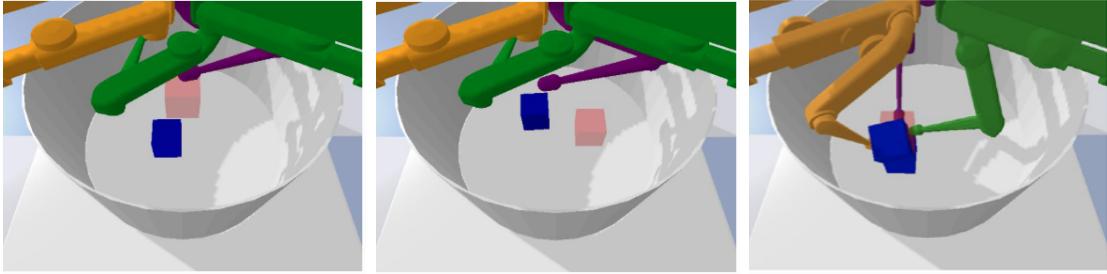


Рис. 6: Примеры наблюдений в среде Pushing библиотеки CasualWorld.

интерфейс для параметрического изменения характеристик среды, как визуальных, так и физических. Аналогично MetaWorld, в CasualWorld содержится набор заранее сконструированных сред и задач в них, а также заранее заданные наборы параметров варьирования задач для тренировки и тестирования алгоритмов. Сами задачи являются более сложными по сравнению с MetaWorld из-за большей размерности пространства действий, обусловленной конструкцией манипулятора. Эксперименты в данной среде преследовали цель проверки работы алгоритма в более близких условиях к реальным.

В качестве среды для тестирования была выбрана Pushing. В наборе задач в этой среде от агента требуется совместить незакреплённый параллелепипед с целью такого же размера, присутствующей на изображении с камер. Действия являются 9-мерными векторами  $a_t \in \mathbb{R}^9$  и представляют собой координаты трёх пальцев. Для получения наблюдений используются две камеры, положение и ориентация которых закреплены для всех задач. Каждая камера предоставляет маску сегментации для манипулятора и объекта, после чего изображения фильтруются, конкатенируются и подаются на вход как  $o_t^r$  и  $o_t^o$  соответственно. Каждое получившееся наблюдение является тензором  $64 \times 64 \times 6$ . Примеры отдельных изображений среды представлены на рис. 6.

Встроенная функция награды является относительной, то есть зависит от предыдущего состояния среды  $\mathcal{R}(s_t, s_{t+1})$ . Поскольку модель функции награды имеет вид  $q(r_t | s_t)$ , для базового алгоритма задача является достаточно сложной, что выражается в отсутствии прироста показателей качества модели достаточно долгое время при обучении. Для решения данной проблемы функция награды среды была изменена. Тогда как механика расчета наград, генерируемых средой, была оставлена нетронутой, агент наблюдает не исходные награды, а сумму всех наград с начала эпизода. Подобный дизайн награды помогает сделать её менее зависимой от  $s_t$ , хо-

Таблица 1: Определение параметров вариации задач в CasualWorld.

Параметр вариации	Тренировочные задачи	Тестовые задачи
Положение объекта	от $[0.0, -\pi, 0.425]$	до $[0.11, -\pi, 0.425]$
	до $[0.11, \pi, 0.425]$	до $[0.15, \pi, 0.425]$
Ориентация объекта	от $[0, 0, -\pi]$	от $[0, 0, -\pi]$
	до $[0, 0, \pi]$	до $[0, 0, \pi]$
Размеры объекта	от $[0.075, 0.075, 0.085]$	от $[0.095, 0.095, 0.085]$
	до $[0.095, 0.095, 0.085]$	до $[0.115, 0.115, 0.085]$
Масса объекта	от 0.015	от 0.045
	до 0.045	до 0.10

ти и оставляет зависимость от начального состояния среды  $s_0$ . Все представленные результаты для данной среды получены при помощи замены исходных наград  $r_t$  на аккумулированные  $\hat{r}_t$  в процессе обучения.

В экспериментах используются следующие вариации параметров среды:

- Начальное положение объекта  $(R_\tau, \alpha_\tau, \frac{h_\tau}{2})$
- Начальная ориентация объекта  $(R_\tau, \alpha_\tau, \frac{h_\tau}{2})$
- Линейные размеры объекта  $(l_\tau, w_\tau, h_\tau)$
- Масса объекта  $m_\tau$

Подробное описание параметров вариации представлено в Таблице 1.

На рис. 7 представлены результаты обучения алгоритмов в среде CasualWorld. После результатов, полученных на эксперименте в RotatedDrawerWorld, было принято решение исключить более слабый алгоритм СЕМА-TopDown из числа рассматриваемых по причине большей сложности CasualWorld. Как можно заключить из графиков, все испытанные методы уверенно превосходят базовый алгоритм на тренировочных задачах. На тестовых задачах вариант СЕМА-Generalize показывает себя хуже, что может быть проявлением усложнившегося взаимодействия с объектом - вектору влияния слишком сложно предсказать все варианты того, как манипулятор может влиять на объект в любой момент времени.

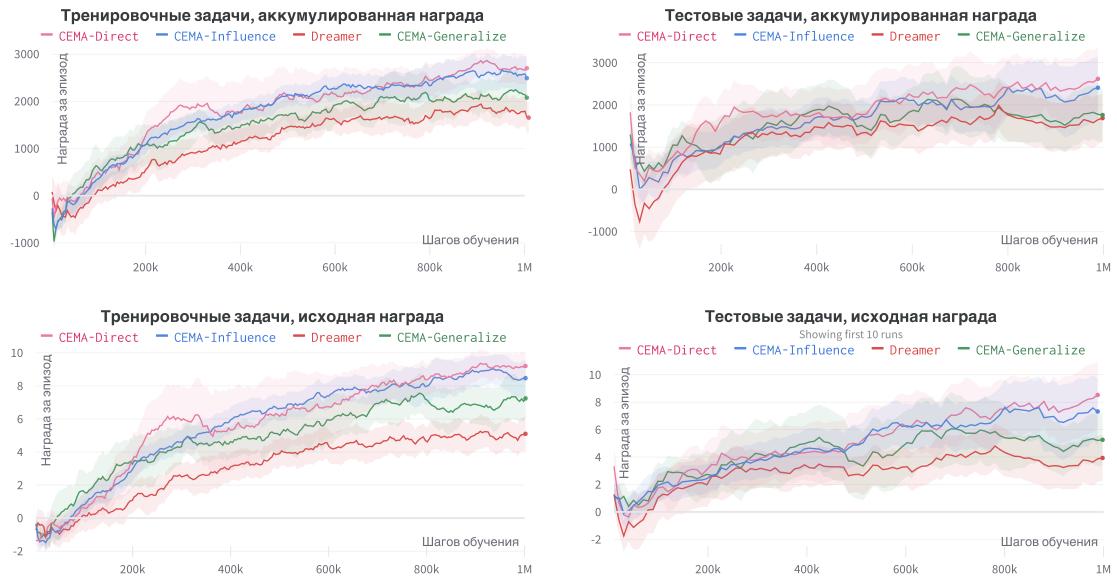


Рис. 7: Графики тренировочного процесса в среде Pushing библиотеки CasualWorld.

## 5 Заключение

В настоящей работе были предложены и описаны объектно-центричные модели мира в обучении с подкреплением для алгоритмов модельного подхода, решающих задачу визуального роботизированного контроля. За счет предположения о работе только с манипулятором и объектом, стало возможным внести в структуру в явном виде предположения о причинно-следственных связях в среде. Благодаря подобному моделированию, а также благодаря модификациям, свойственным только объектно-центричным моделям мира, было достигнуто улучшение обобщающей способности базового алгоритма.

Предложенное направление исследований представляется весьма перспективным, поскольку применение объектно-центричного подхода в других областях науки уже доказало его состоятельность. В настоящей работе была показана возможность добиться улучшений обобщающей способности алгоритмов, но при этом присутствовал ряд существенных ограничений, ограничивающих применение алгоритма для реальных задач.

Таким образом, для дальнейших исследований можно выделить следующие направления:

1. избавиться от необходимости использовать истинные сегментационные маски в алгоритме;
2. исследовать возможности интеллектуальной активации вектора влияния в моменты взаимодействия манипулятора с объектом;
3. рассмотреть случай с большим количеством объектов и/или манипуляторов;
4. рассмотреть возможные применения объектно-центричного подхода для улучшения иных качеств алгоритмов обучения с подкреплением.

## Список литературы

- [1] Causalworld: A robotic manipulation benchmark for causal structure and transfer learning / O. Ahmed, F. Träuble, A. Goyal et al. // *arXiv preprint arXiv:2010.04296*. — 2020.
- [2] Deep variational information bottleneck / A. A. Alemi, I. Fischer, J. V. Dillon, K. Murphy // *arXiv preprint arXiv:1612.00410*. — 2016.
- [3] Dream to control: Learning behaviors by latent imagination / D. Hafner, T. Lillicrap, J. Ba, M. Norouzi // *arXiv preprint arXiv:1912.01603*. — 2019.
- [4] Entity abstraction in visual model-based reinforcement learning / R. Veerapaneni, J. D. Co-Reyes, M. Chang et al. // Conference on Robot Learning / PMLR. — 2020. — Pp. 1439–1456.
- [5] Gatsbi: Generative agent-centric spatio-temporal object interaction / C.-H. Min, J. Bae, J. Lee, Y. M. Kim // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. — 2021. — Pp. 3074–3083.
- [6] Ha D., Schmidhuber J. Recurrent world models facilitate policy evolution // *Advances in neural information processing systems*. — 2018. — Vol. 31.
- [7] Hallak A., Di Castro D., Mannor S. Contextual markov decision processes // *arXiv preprint arXiv:1502.02259*. — 2015.
- [8] Kingma D. P., Welling M. Auto-encoding variational bayes // *arXiv preprint arXiv:1312.6114*. — 2013.
- [9] Kipf T., van der Pol E., Welling M. Contrastive learning of structured world models // *arXiv preprint arXiv:1911.12247*. — 2019.
- [10] Learning latent dynamics for planning from pixels / D. Hafner, T. Lillicrap, I. Fischer et al. // International conference on machine learning / PMLR. — 2019. — Pp. 2555–2565.
- [11] Learning object-centric transformation for video prediction / X. Chen, W. Wang, J. Wang, W. Li // Proceedings of the 25th ACM international conference on Multimedia. — 2017. — Pp. 1503–1512.

- [12] Mastering atari with discrete world models / D. Hafner, T. Lillicrap, M. Norouzi, J. Ba // *arXiv preprint arXiv:2010.02193*. — 2020.
- [13] Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning / T. Yu, D. Quillen, Z. He et al. // Conference on robot learning / PMLR. — 2020. — Pp. 1094–1100.
- [14] Multi-object representation learning with iterative variational inference / K. Greff, R. L. Kaufman, R. Kabra et al. // International Conference on Machine Learning / PMLR. — 2019. — Pp. 2424–2433.
- [15] Object-centric learning with slot attention / F. Locatello, D. Weissenborn, T. Unterthiner et al. // *Advances in Neural Information Processing Systems*. — 2020. — Vol. 33. — Pp. 11525–11538.
- [16] Roll: Visual self-supervised reinforcement learning with object reasoning / Y. Wang, G. N. Narasimhan, X. Lin et al. // *arXiv preprint arXiv:2011.06777*. — 2020.
- [17] Schmeckpeper K., Georgakis G., Daniilidis K. Object-centric video prediction without annotation // 2021 IEEE International Conference on Robotics and Automation (ICRA) / IEEE. — 2021. — Pp. 13604–13610.
- [18] Sutton R. S., Barto A. G. Reinforcement learning: An introduction. — MIT press, 2018.
- [19] Tishby N., Pereira F. C., Bialek W. The information bottleneck method // *arXiv preprint physics/0004057*. — 2000.
- [20] A tutorial on energy-based learning / Y. LeCun, S. Chopra, R. Hadsell et al. // *Predicting structured data*. — 2006. — Vol. 1, no. 0.