

**Отчёт о выполнении тестового задания  
«DINO-Tracker: Taming DINO for Self-Supervised Point  
Tracking in a Single Video»**

Выполнил: Дюжев В. Д.

Санкт-Петербург, 2024

# ОГЛАВЛЕНИЕ

<b>Введение</b>	<b>1</b>
1 Краткое описание . . . . .	1
2 Постановка задачи . . . . .	1
3 Структура отчета . . . . .	1
<b>Принцип работы</b>	<b>2</b>
1 Идея . . . . .	2
2 Архитектура . . . . .	2
2.1 DINoV2-ViT . . . . .	3
2.2 DELTA-DINO . . . . .	3
2.3 CNN-Refiner . . . . .	3
2.4 Денормализация . . . . .	5
3 Обучение . . . . .	5
3.1 Обучающие наборы . . . . .	5
3.2 Функции потерь . . . . .	5
3.3 Гиперпараметры (In progress) . . . . .	7
<b>Запуск предобученной модели</b>	<b>8</b>
1 Базовая визуализация . . . . .	8
2 Метрики качества . . . . .	9
3 Производительность . . . . .	11
4 Мульти-трекинг . . . . .	12
<b>Запуск на собственных данных (In progress)</b>	<b>13</b>
<b>Техническая информации (In progress)</b>	<b>14</b>
<b>Выводы (In progress)</b>	<b>15</b>

# Введение

## Краткое описание

Dino-Tracker — метод трекинга, созданный на основе модели DINOv2-ViT, которая используется как базовая для получения качественных обобщений изображений, содержащих важную семантическую информацию. Он позволяет добиться высокой точности, при этом относится к области self-supervised learning, что расширяет возможности его использования (т.к. нет необходимости разметки данных). Перед применением предполагается обучение на конкретном видео.

## Постановка задачи

Дана последовательность кадров  $\{I^t\}_{t=1}^T$ , где  $T$  — длина видео. Задача состоит в обучении модели-трекера  $\Pi$ , принимающего на вход целевую точку  $x_q$  (query point) и предсказывающего набор  $\{\hat{x}^t\}_{t=1}^T$  — траекторию данной точки (положения на всех кадрах). Целевая точка при этом задается положением и кадром (далее — начальный кадр). Таким образом итоговым результатом является возможность предсказания траектории любой точки любого кадра на протяжении всего видео.

## Структура отчета

Данный отчет состоит из 5 секций:

1. **Принцип работы** — описание архитектуры нейросетевой модели и алгоритмов постобработки, а также процесса обучения.
2. **Запуск предобученной модели** — содержит базовые результаты экспериментов с демонстрацией основных возможностей метода на предложенных в оригинальном репозитории данных.
3. **Запуск на собственных данных** — результаты обучения модели на собранных вручную реальных данных.
4. **Техническая информация** — описание процесса взаимодействия с ПО, дополнения к оригинальному репозиторию.
5. **Выводы** — подведение итогов, выдвижение гипотез для улучшения метода.

# Принцип работы

## Идея

Как говорилось ранее, в основе метода лежит предобученная модель DINOv2-ViT (визуальный трансформер) для формирования признаков, содержащих семантическую информацию (далее будем называть эту модель экстрактором признаков). В статье утверждается, что такие признаки, однако, имеют недостаточное содержание временной информации (для связей внутри видео). В качестве решения данной проблемы предлагается обучать дополнительную модель DELTA-DINO, формирующую добавки к выходу экстрактора признаков для формирования их уточнений. Уточненные признаки могут рассматриваться как обобщения для целой траектории, что в дальнейшем позволит сопоставлять точки на разных кадрах.

## Архитектура

В простейшем случае входом сети является целевая точка  $x_q$ , ее начальный кадр  $I^k$  и кадр  $I^t$  для которого необходимо произвести предсказание положения  $\hat{x}^t$ . Обозначим выход модели DINOv2 как  $\Phi_{DINO}(I)$ , а выход DELTA-DINO как  $\Phi_{\Delta}(I)$ . Тогда в качестве уточненных признаков будем иметь:

$$\Phi(I) = \Phi_{DINO}(I) + \Phi_{\Delta}(I). \quad (1)$$

После получения представления для обоих кадров ( $\Phi(I^k)$  и  $\Phi(I^t)$ ) производится процесс сопоставления. Целевой точке  $x_q$  на кадре  $I^k$  ставится в соответствие признак  $\varphi_q$  из  $\Phi(I^k)$  путем билинейной интерполяции (т.к. размеры карты признаков уменьшены — результат токенизации в DINOv2-ViT).

Далее происходит построение карты корреляции  $S$  между признаками  $\Phi(I^t)$  и  $\varphi_q$  путем расчета косинусного расстояния:

$$S(p) = \frac{\varphi_q^T \Phi^t(p)}{\|\varphi_q\| \|\Phi^t(p)\|}, \quad (2)$$

где  $p$  — обозначение положения на карте.

Для получения итоговой карты распределения вероятности  $H$  находящаяся целевой точки карта корреляции подается на вход сверточной сети (CNN-Refiner) с функцией Softmax на выходе.

Предсказание положения формируется как взвешенная сумма координат точек на карте распределения в окрестности точки (радиуса  $R$ ) с наи-

большей вероятностью ( $x_{p_{\max}}$ ):

$$\hat{x}^t = \frac{\sum_{p \in \Omega} H(p) \cdot x_p}{\sum_{p \in \Omega} H(p)}, \quad (3)$$

где  $\Omega = \{p : \|x_p - x_{p_{\max}}\|_2 \leq R\}$ . Итоговым предсказанием является  $\Pi(x_q, t) = \hat{x}^t$ , и траектория для  $x_q - \mathcal{T}_q = \{\hat{x}^t : \hat{x}^t = \Pi(x_q, t), t = 1 \dots T\}$ .

Ниже приведены подробности реализации описанного алгоритма.

## 2.1 DINOv2-ViT

DINOv2-ViT — предобученный визуальный трансформер (используется модель ViT/14), широко применяемый как базовая модель экстракции признаков в области компьютерного зрения. Являясь частью DINO-Tracker, его веса не участвуют в обучении. Единственным изменением, отличающим используемый экстрактор от оригинального является уменьшенный размер шага токенизации (7 вместо 14) для повышения разрешения карты признаков. Для изображений 480x854 карта признаков имеет размерность 1024x67x121 (1024 — размерность пространства признаков).

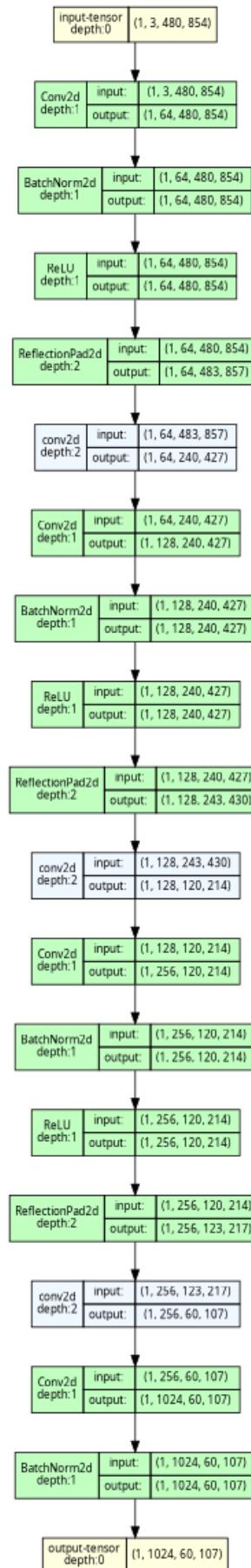
## 2.2 DELTA-DINO

Рассмотрим подробнее устройство модели DELTA-DINO. Эта сверточная сеть состоит только из 4 слоев со следующим преобразованием каналов:  $[3 \rightarrow 64 \rightarrow 128 \rightarrow 256 \rightarrow 1024]$ . Каждый слой, кроме последнего представляет собой комбинацию Conv2d (kernel: 5, stride: 1, reflection padding: 2), BatchNorm2d, ReLU, BlurPool. Последний слой: Conv2d (kernel: 5, stride: 1, reflection padding: 4, dilation: 2), BatchNorm2d. На рисунке 1 представлена визуализация данной модели (размерности расчитаны с учетом входного изображения 480x854).

Важно отметить, что размерность выхода DELTA-DINO (1024x60x107) не совпадает с исходной размерность карты признаков DINOv2-ViT. Для расчета уточненных признаков необходимо совершить приведение к размерности признаков исходных. Данная задача решается методом линейной интерполяции координатной сеткой.

## 2.3 CNN-Refiner

Сверточная нейронная сеть на выходе имеет структуру: Conv2d (kernel: 3, padding: 1), ReLU, Conv2d (kernel: 3, padding: 1). Преобразование каналов:  $[1 \rightarrow 16 \rightarrow 1]$ . Таким образом, размерность карты распределения остается такой же как у карты корреляции.



*Puc. 1. Архитектура DELTA-DINO (визуализация — torchview).*

## 2.4 Денормализация

Как было сказано в прошлом пункте, выходная карта распределения имеет размерность, совпадающую с картой корреляции и картой признаков. Для работы алгоритма в исходной постановке необходимо вернуть предсказанные положения в размерность входных изображений. Для этого производится линейная денормализация предсказанного положения. Именно для обеспечения большей точности и «гладкости» данной процедуры сопоставления при расчете  $\hat{x}^t$  применяется взвешенная сумма.

## Обучение

Обучение DINO-Tracker происходит в парадигме self-supervised learning на конкретном видео. Это подразумевает нахождение и использование закономерностей в данных для выполнения задачи. Предобработка данных представляет из себя расчет оптического потока (ОП) в видео, а также нахождение семантически близких точек в последовательности кадров (на основе признаков DINOV2-ViT).

### 3.1 Обучающие наборы

Набор соответствующих друг другу точек, найденный на основе расчета оптического потока  $\Omega_{\text{flow}}$  состоит из близких по кадрам точек (т.к. надежность оптического потока теряется со временем).

Для сэмплинга набора семантически близких точек  $\Omega_{\text{DINO-bb}}$  используется принцип ближайших соседей:

$$NN(\varphi_{\text{DINO}}^i, \Phi_{\text{DINO}}(I^j)) = \varphi_{\text{DINO}}^j \wedge NN(\varphi_{\text{DINO}}^j, \Phi_{\text{DINO}}(I^i)) = \varphi_{\text{DINO}}^i, \quad (4)$$

где  $NN(\varphi, \Phi)$  — обозначение ближайшего соседа  $\varphi$  в карте  $\Phi$ ,  $I^i, I^j$  — выбранные кадры. В процессе обучения аналогичным образом формируется набор  $\Omega_{\text{rfn-bb}}$  на основе уточненных признаков.

### 3.2 Функции потерь

Задача оптимизации ставится для агрегированной функции потерь, состоящей из нескольких компонентов:

$$\mathcal{L} = \mathcal{L}_{\text{flow}} + \lambda_1 \mathcal{L}_{\text{dino-bb}} + \lambda_2 \mathcal{L}_{\text{rfn-bb}} + \lambda_3 \mathcal{L}_{\text{rfn-cc}} + \lambda_4 \mathcal{L}_{\text{prior}}, \quad (5)$$

где  $\{\lambda_i\}$  — весовые коэффициенты,  $\mathcal{L}_{\text{flow}}$  — функция потерь ОП,  $\mathcal{L}_{\text{dino-bb}}$  и  $\mathcal{L}_{\text{rfn-bb}}$  — функции потерь семантически схожих точек (на основе сырых и уточненных признаков),  $\mathcal{L}_{\text{rfn-cc}}$  — функция потерь циклической согласованности,  $\mathcal{L}_{\text{prior}}$  — функция потерь априорной информации.

В некоторых последующих формулах применяется функция потерь Хубера:

$$L_H(x, y) = \begin{cases} 0.5\|x - y\|^2, & \|x - y\| < 1 \\ \|x - y\| - 0.5, & \|x - y\| \geq 1 \end{cases} \quad (6)$$

### Потери ОП

Функция представляет собой вычисление суммы ошибок для прямого и обратоного потока:

$$\mathcal{L}_{\text{flow}} = \sum_{(x^i, x^j) \in \Omega_{\text{flow}}} L_H(\Pi(x^i, j), x^j) + L_H(\Pi(x^j, i), x^i) \quad (7)$$

### Потери семантически схожих точек

Ставится цель увеличения корреляции между уточненными признаками семантически схожих точек и уменьшения корреляции с остальными. Для этого расчитывается «contrastive loss» (широко применяется в задачах self-supervised learning) между парами из  $\Omega_{\text{dino-bb}}$ :

$$l(\varphi^i, \varphi^j) = -\log \frac{\exp(\text{cos-sim}(\varphi^i, \varphi^j)/\tau)}{\sum_p \exp(\text{cos-sim}(\varphi^i, \Phi^j(p))/\tau)}, \quad (8)$$

где  $\tau$  — параметр «температуры», регулирующий гладкость.

Итоговая функция потерь имеет вид:

$$\mathcal{L}_{\text{dino-bb}} = \frac{1}{|\Omega_{\text{dino-bb}}|} \sum_{(\varphi^i, \varphi^j) \in \Omega_{\text{dino-bb}}} \frac{1}{2} w_{\text{dino-bb}}^{ij} (l(\varphi^i, \varphi^j) + l(\varphi^j, \varphi^i)), \quad (9)$$

где  $\{w_{\text{dino-bb}}^{ij}\}$  — веса, рассчитываемые на основе величины веренности в сходстве.

$\mathcal{L}_{\text{rfn-bb}}$  определяется аналогичным образом.

### Потери циклической согласованности

Важным свойством трекера должна быть циклическая согласованность, а именно: если  $x^j = \Pi(x^i, j)$ , то  $x^i \approx \Pi(x^j, i)$ . Функция потерь формулируется с учетом этого:

$$\mathcal{L}_{\text{rfn-cc}} = \sum_{(x^i, x^j) \in \Omega_{\text{rfn-cc}}} \frac{1}{2} w_{\text{rfn-cc}}^{ij} (L_H(\Pi(x^i, j), x^j) + L_H(\Pi(x^j, i), x^i)), \quad (10)$$

где  $\{w_{\text{rfn-cc}}^{ij}\}$  — веса, рассчитанные на основе ошибки циклической согласованности.

### Потери априорной информации

Признаки полученные напрямую от DINOv2-ViT обладают высокой обобщающей способностью и содержат важную семантическую априорную информацию. Для ее сохранения вводится функция потерь, поощряющая сохранение нормы и ориентации уточненных признаков (для каждого кадра).

$$\mathcal{L}_{\text{prior}} = \frac{1}{H' \cdot W'} \cdot \sum_p \left| 1 - \frac{\|\Phi(I)[p]\|_2}{\|\Phi_{\text{DINO}}(I)[p]\|_2} \right| + |1 - \text{cos-sim}(\Phi(I)[p], \Phi_{\text{DINO}}(I)[p])|, \quad (11)$$

где  $H', W'$  — размеры карты признаков.

### 3.3 Гиперпараметры (In progress)

## Запуск предобученной модели

В данной секции представлены результаты использования модели DINO-Tracker на последовательностях из датасета Tapvid-Davis-480. Данные доступны по ссылке, указанной в репозитории.

### Базовая визуализация

В качестве примеров для демонстрации работоспособности базовой модели DINO-Tracker (предсказаний нейросетевой части, без постобработки) были выбраны видео-последовательности с достаточно простым профилем движения, без явных окклузий. На рисунках 2-4 изображены по 3 кадра (первый, центральный и последний) с демонстрацией промежуточных траекторий для целевых точек (начальным кадром во всех случаях является первый).



Рис. 2. Траектории DINO-Tracker. Tapvid-Davis-480/29.



Рис. 3. Траектории DINO-Tracker. Tapvid-Davis-480/26.



Рис. 4. Траектории DINO-Tracker. Tapvid-Davis-480/21.

Можем заметить, что полученные траектории имеют естественный вид. Базовый метод может быть успешно применен и к трекингу в более сложных окружениях (рисунок 5), однако важно чтобы целевая точка находилась в достаточно уникальной части изображения и не подвергалась окклюзиям. Например, на рисунке 6 видно, что целевая точка сопоставляется неправильно с определенного момента времени (из-за появления схожего объекта и окклюзии первоначальной цели). Данные ограничения помогают во многом снять постобработка с предсказанием окклюзий, о которой будет сказано в дальнейшем.



Рис. 5. Траектории DINO-Tracker. Tapvid-Davis-480/13.

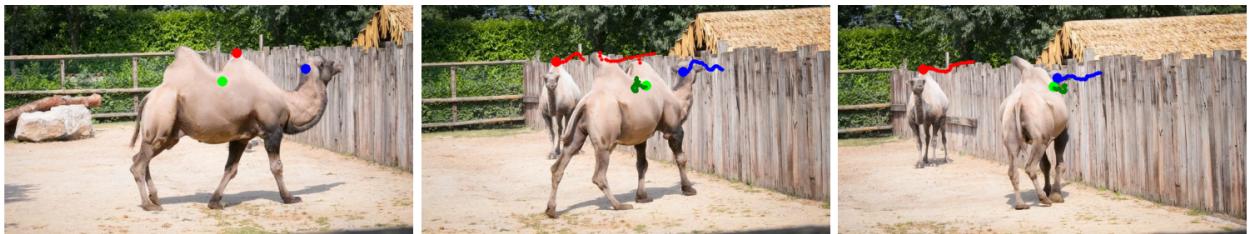


Рис. 6. Траектории DINO-Tracker. Tapvid-Davis-480/15.

## Метрики качества

Оценка качества трекинга производится на датасете Tapvid-Davis-480 согласно инструкции, указанной в репозитории (также есть возможность провести оценку на других предложенных наборах данных, однако ввиду длительности процесса был выбран один из них).

Для оценки используются метрики, предложенные в оригинальной статье «TAP-Vid: A Benchmark for Tracking Any Point in a Video»:

1. occlusion accuracy ( $OA$ ) — отношение количества правильно предсказанных окклюзий в последовательности к длине последовательности (стандартная accuracy классификации);
2. position accuracy ( $< \delta^x$ ) — отношение количества правильно предсказанных положений видимых точек (без окклюзий) последовательности к количеству видимых точек, где правильными считаются пред-

сказания, лежащие в заданной окрестности (в пикселях) действительных значений;

3. jaccard — объединенная метрика для оценки качества предсказаний окклюзий и положений, рассчитываемая как отношение количества правильно предсказанных положений видимых точек с соответствующим правильным предсказанием окклюзий последовательности к количеству видимых точек + количеству неправильно классифицированных невидимых точек или точек с неправильными предсказаниями положений, предсказанных видимыми.

Важно отметить, что метрики  $< \delta^x$  и jaccard рассчитываются для всех изображений последовательности (кроме начальных для соответствующих ключевых точек) с учетом приведения к размеру 256x256. В качестве итоговых метрик выступают  $OA$ , average jaccard ( $AJ$ ) и average position accuracy ( $< \delta_{avg}^x$ ), где последние две рассчитываются как средние значения jaccard и  $< \delta^x$  для окрестностей в 1, 2, 4, 8 и 16 пикселей.

<b>id</b>	<b><math>OA</math></b>	<b><math>AJ</math></b>	<b><math>&lt; \delta_{avg}^x</math></b>	<b>id</b>	<b><math>OA</math></b>	<b><math>AJ</math></b>	<b><math>&lt; \delta_{avg}^x</math></b>
25	0.86407	0.52461	0.71564	23	0.88914	0.70477	0.8492
7	0.84293	0.62454	0.82839	14	0.95139	0.64647	0.75761
20	0.83599	0.5811	0.77156	13	0.89146	0.59985	0.72462
1	0.83536	0.55336	0.77867	29	0.96237	0.78638	0.87435
26	0.92519	0.72271	0.84825	17	0.89871	0.61162	0.75912
5	0.87887	0.60303	0.8446	4	0.87391	0.66864	0.88173
2	0.69653	0.37687	0.72344	11	0.79149	0.57035	0.7935
19	0.82501	0.47515	0.70126	16	1.0	0.94087	0.96645
6	0.85056	0.63005	0.80656	10	0.94926	0.76182	0.86853
22	0.79057	0.54181	0.76219	3	0.91635	0.75579	0.8585
8	0.88787	0.69386	0.84002	21	0.98371	0.84777	0.90917
28	0.93551	0.73576	0.84587	24	0.94805	0.8512	0.9379
0	0.99963	0.64083	0.72042	15	0.96577	0.80356	0.87399
9	0.69041	0.29614	0.5251	18	0.81178	0.52368	0.71774
27	0.87957	0.62347	0.7801	12	0.99365	0.88775	0.93579
<b><i>avg</i></b>	<b><i>0.8855</i></b>	<b><i>0.65279</i></b>	<b><i>0.80668</i></b>	—	—	—	—

Таблица 1. Метрики качества. Датасет Tapvid-Davis-480.

В статье завлены следующие данные для рассматриваемого датасета:  $OA = 0.881$ ,  $AJ = 0.646$ ,  $< \delta_{avg}^x = 0.804$ . Эксперимент подтверждает достижение таких значений.

## Производительность

Эксперименты производятся на устройстве Lenovo Legion 7 (AMD Ryzen 9 5900HX, NVIDIA GeForce RTX 3080 120W 16Gb, 32 Gb RAM).

Для получения данных о производительности в инференсе метод был применен ко всем последовательностям из датасета. Эксперимент проводился только для нейросетевой части DINO-Tracker. Дальнейшее предсказание окклюзий сильно зависит от контекста изображения. На вход модели подавалась одна ключевая точка, соответствующая центру первого кадра. Принятый размер батча: 1. Ниже приведен график, иллюстрирующий зависимость времени обработки видео от количества кадров в нем.

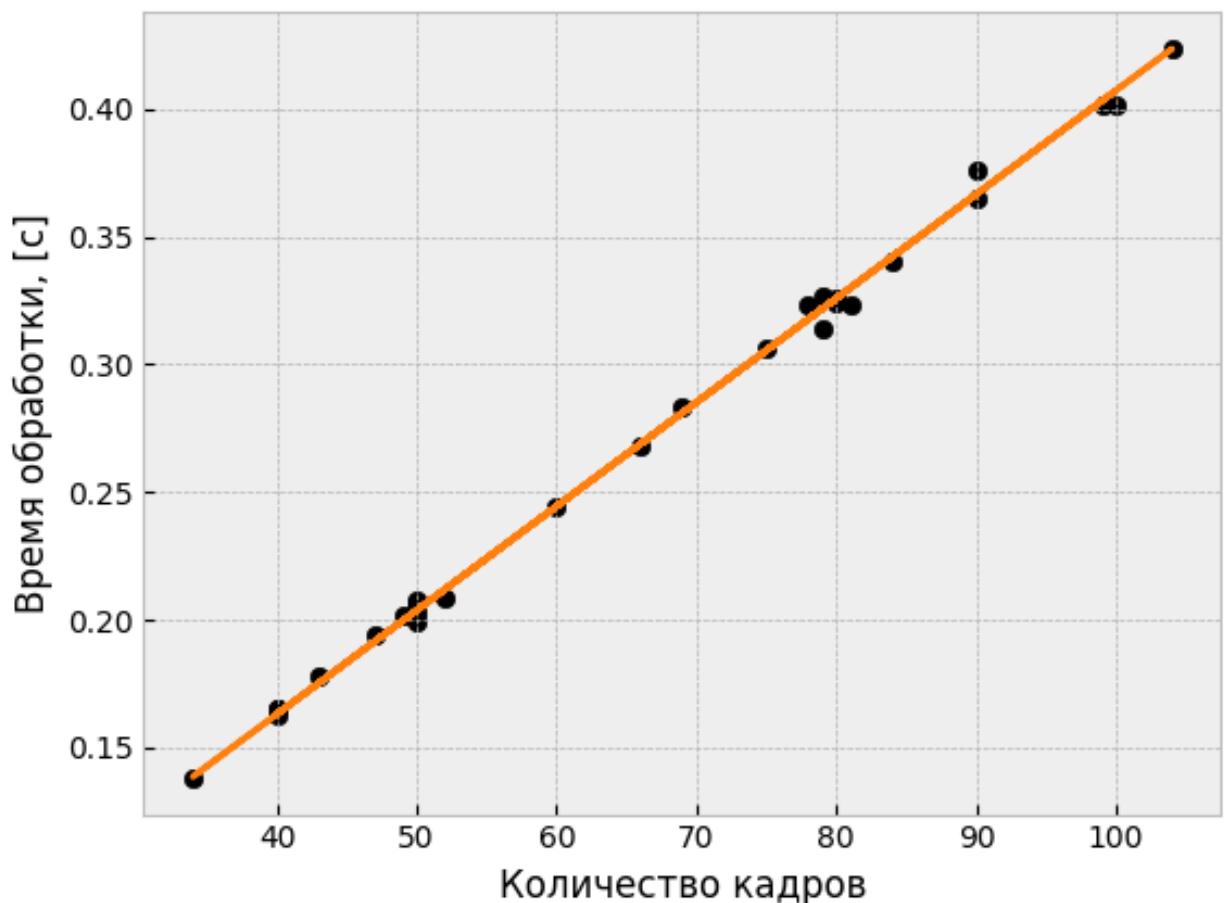


Рис. 7. Зависимость времени инференса нейросетевой части DINO-Tracker от количества кадров.

Алгоритм обработки видео последовательно предсказывает положение целевой точки на кадрах. Данные подтверждают, что зависимость является линейной. Аппроксимировав ее моделью линейной регрессии без смещения получим, что на предсказание позиции целевой точки в одном кадре тратится в среднем 0.0041 [с].

В процессе инференса в среднем затрачивалось 14.5 Гб видеопамяти.

## Мульти-трекинг

Задача множественного трекинга объектов подразумевает наличие механизма сопоставление объектов траекториями движения (ассоциация данных). Базовый метод DINO-Tracker не использует данную процедуру. Как видно на примерах, представленных выше (рисунки 5, 6) в такой конфигурации он может быть применен при отсутствии явных окклюзий и достаточном количестве семантически уникальных точек на объектах.

При использовании постобработки (предсказания окклюзий) область применения метода для задач множественного трекинга можно существенно расширить. По сути, в алгоритме обработки окклюзий частично применяется подход ассоциации — проверяется схожесть траекторий на разных участках видео согласно установленному критерию. На рисунке 8 показаны положения целевых точек (1 и 30 кадр), изначально расположенных на объектах.



*Рис. 8. Трекинг точек на нескольких объектах. Tapvid-Davis-480/8.*

Можно предположить, что произведя изначальную детекцию объектов и выбрав наборы целевых точек, соответствующие им производить трекинг объектов как среднее положение всех видимых соответствующих ему точек. Однако такой подход требует усовершенствования для обработки новых объектов появляющихся на видео (что в итоге возвращает к задаче верхнеуровневой ассоциации данных). Кроме того, даже с учетом обработки окклюзий семантически близкие точки могут неверно сопоставляться.

## Запуск на собственных данных (In progress)

## Техническая информации (In progress)

## Выводы (In progress)