Нейросетевая система технического зрения для детектирования и трекинга канцелярских товаров

А. Место [1], *Х. Чан* [1], *С.А.К. Диане* [1,2]

[1] *МИРЭА – Российский технологический университет, Москва, Россия*
[2] *Институт проблем управления им. В. А. Трапезникова РАН, Москва, Россия*

Аннотация **– Информационная поддержка решений по управлению автономными манипуляционными роботами, устанавливаемыми на конвейерных линиях в целях сортировки транспортируемых объектов, требует применения современных средств анализа изображений. В статье предложен подход к подготовке обучающего множества и настройке сверточной нейронной сети YOLOv8n для детектирования и нейронной сети SORT/DeepSORT для трекинга канцелярских товаров. Исследована обобщающая способность и устойчивость разработанной системы технического зрения к зашумлениям изображения. Предлагается схема внедрения нейросетевой модели в контур системы управления автономным манипуляционным роботом.**

Ключевые слова **– система технического зрения, сверточная нейронная сеть, автономный манипуляционный робот, детектирование объектов, визуальный трекинг, YOLOv8, SORT, DeepSORT.**

ВВЕДЕНИЕ

В условиях современной промышленной автоматизации системы технического зрения являются неотъемлемой частью роботизированных производственных линий, в которых визуальная информация используется для управления технологическими процессами, контроля качества и координации действий промышленных манипуляторов [1].

Одним из распространённых сценариев применения таких систем является задача обнаружения, локализации и сопровождения объектов, перемещающихся по конвейерной ленте.

Для подобных приложений ключевыми требованиями являются высокая точность обнаружения и минимальное время обработки данных. Алгоритм должен функционировать в условиях ограниченных вычислительных ресурсов, поскольку на промышленных объектах зачастую используются системы управления, основанные на центральных процессорах без специализированных графических ускорителей.

В рамках настоящей работы в качестве объектов наблюдения рассматриваются канцелярские товары (ластики и тюбики клея), расположенные на рабочей поверхности, имитирующей конвейер. Это позволяет воспроизвести реальные условия промышленной сортировки и апробировать разработанные алгоритмы детектирования и трекинга в квази-производственной обстановке. На Рис. 1 показана система “робот Дельта” типичная для операций взятия и установки на конвейерных линиях.

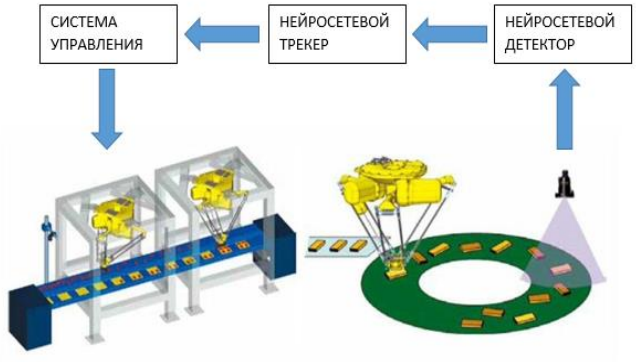

Рис. 1. Схема внедрения нейросетевой модели в контур системы управления автономным манипуляционным роботом

I. ПОДГОТОВКА ОБУЧАЮЩИХ ДАННЫХ

Качество обучающей выборки является определяющим фактором эффективности нейросетевого детектора [2]. В рамках данной работы авторы сформировали обучающие данные для двух взаимодополняющих компонентов системы.

А. Данные для детектора

Исходные изображения получены RGB-камерой в условиях, приближённых к производственным: различные фоновые поверхности (конвейерная лента, рабочий стол), вариации освещения, частичные перекрытия объектов. Разрешение – 1664×1664

пикселя. Определены два класса: Класс 0 – Цилиндр, Класс 1 – Треугольник.

Аннотирование выполнялось вручную в программе LabelImg в формате YOLO (class_id, center_x, center_y, width, height – нормализованные координаты) [3]. Всего размечено 213 изображений, 1270 объектов. Пример разметки показан на Рис. 2.

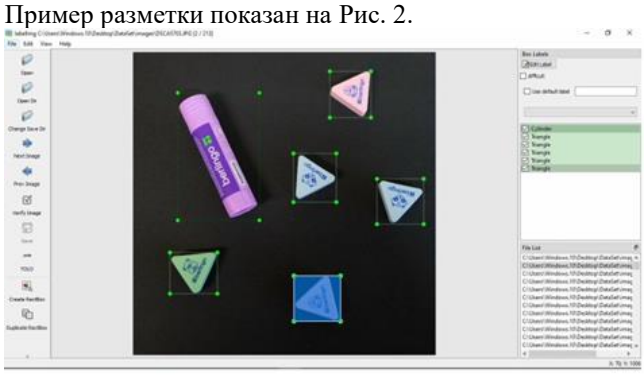

Рис. 2. Пример разметки изображений в программе LabelImg

Статистическая информация об обучающей выборке представлена на Рис. 3.

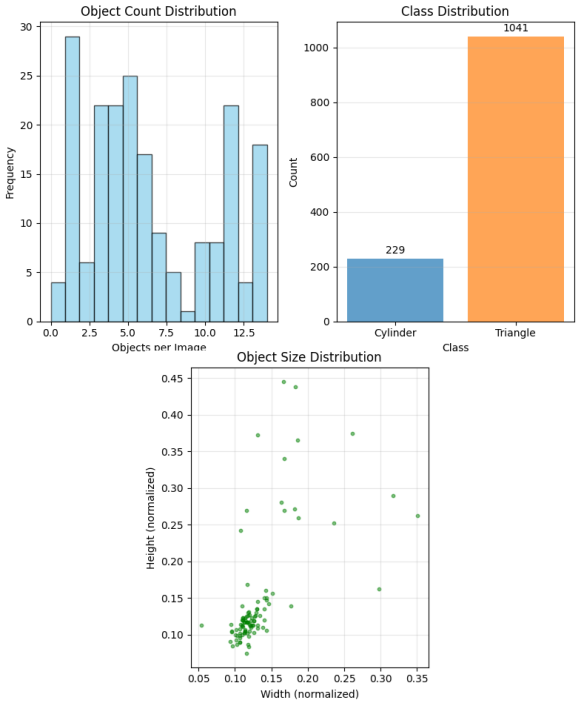

Рис. 3. Распределение количества объектов, классов и размеров в обучающей выборке

Б. Данные для трекера

Для обучения компонента ReID формировался набор данных из видеопоследовательностей: объекты вручную обрезались по ограничивающим прямоугольникам и организовывались в иерархическую структуру ReID (train/[id]/[frame].jpg). Аннотирование YOLO-данных выполнялось в Label Studio. Дополнительно подготовлены видео-последовательности с тремя сценариями: без перекрытий, с частичными перекрытиями, с кратковременным полным перекрытием.

В. Сравнительный анализ методологий

Табл. I представляет сравнение методологий подготовки данных двух компонентов системы технического зрения – детектора и трекера.

ТАБЛИЦА I
СРАВНЕНИЕ МЕТОДОЛОГИЙ ПОДГОТОВКИ ОБУЧАЮЩИХ ДАННЫХ

Параметр	Детектор	Трекер/ReID
Инструмент разметки	LabelImg	Label Studio; ручная нарезка кадров
Формат аннотаций	YOLO (.txt)	Папочная структура ReID
Кол-во изображений	213 (1270 объектов)	Видеопоследовательности
Классы	Цилиндр, Треугольник	Ластик, Клей
Аугментация	Mosaic, HSV, Flip, Scale	Аугментация при обучении
Разрешение	1664×1664 пкс	640×640 пкс (входное)

II. ОБУЧЕНИЕ НЕЙРОСЕТЕВОГО ДЕТЕКТОРА

В качестве архитектуры нейросетевого детектора объектов выбрана модель YOLOv8n – наиболее лёгкая версия семейства YOLOv8. Выбор обусловлен необходимостью работы в условиях ограниченных вычислительных ресурсов и возможностью инференса на CPU [4]. Сравнение архитектур приведено в Табл. II.

ТАБЛИЦА II
СРАВНЕНИЕ АРХИТЕКТУР НЕЙРОСЕТЕВЫХ ДЕТЕКТОРОВ

Метод	Тип	Скорость	Точность	CPU
Faster R-CNN	2-stage	Низкая	Высокая	Нет
SSD	1-stage	Высокая	Средняя	Частично
YOLOv8n (выбран)	1-stage	Очень высокая	Высокая	Да (OpenVINO)
YOLOv9s	1-stage	Высокая	Высокая	Частично

Обучение YOLO осуществляется путём минимизации составной функции потерь, включающей несколько компонентов:

$$L = \lambda_{box} \cdot L_{box} + \lambda_{obj} \cdot L_{obj} + \lambda_{cls} \cdot L_{cls}, \quad (1)$$

где L – функции потерь; L_{box} – ошибка локализации (например, на основе IoU); L_{obj} – ошибка предсказания наличия объекта; L_{cls} – ошибка классификации; λ_{box} – коэффициент взвешивания для ошибки локализации; λ_{obj} – коэффициент взвешивания для ошибки предсказания наличия объекта; λ_{cls} – коэффициенты для ошибки классификации.

Обучение нейронной сети осуществлялось в среде Google Colab с GPU. Использовались библиотека Ultralytics YOLOv8, фреймворк PyTorch, инструмент OpenVINO Toolkit для оптимизации инференса на CPU. После завершения модель экспортировалась в формат OpenVINO IR [5]. Графики обучения модели приведены на Рис. 4.

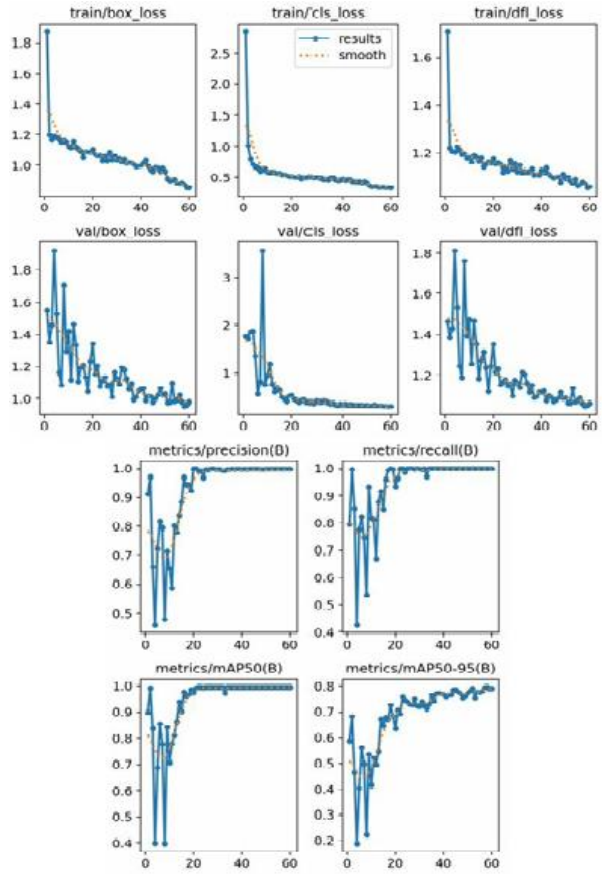

Рис. 4. Графики обучения нейросетевой модели

При обучении нейронной сети YOLO (включая YOLOv8) для оценки качества локализации использовались метрики, основанные на IoU (Intersection over Union):

$$IoU = \frac{Area(B_{pred} \cap B_{gt})}{Area(B_{pred} \cup B_{gt})} \quad (2)$$

где B_{pred} – предсказанный ограничивающий прямоугольник; B_{gt} – истинный ограничивающий прямоугольник.

Кривые обучения демонстрируют хорошую сходимость: функция потерь стабильно убывает и выходит на плато к 120-й эпохе; mAP@0.5 достиг 0.997 (99.7%); признаков переобучения не наблюдается. Параметры конфигурации приведены в Табл. III.

ТАБЛИЦА III
ПАРАМЕТРЫ КОНФИГУРАЦИИ ОБУЧЕНИЯ ДЕТЕКТОРА

Параметр	Значение	Обоснование
Архитектура	YOLOv8n	Лёгкая модель под CPU
Размер входа	1024×1024 пкс (обучение); 640×640 пкс (инференс)	Обучение при высоком разрешении; оптимизированный инференс на CPU
Число эпох	120	Достаточная сходимость
Размер батча	8	Google Colab (GPU, AdamW)
Предоб. веса	yolov8n.pt	Transfer Learning (COCO)
Формат экспорта	OpenVINO IR	CPU-инференс
Порог уверенности (класс 0)	0,15	Повышение полноты цилиндров
Порог уверенности (класс 1)	0,30	Снижение ложных срабатываний
Порог NMS (IoU)	0,60	Подавление дублей bbox

III. НАСТРОЙКА АЛГОРИТМА ТРЕКИНГА

A. Выбор метода трекинга

Задача визуального трекинга требует сохранения идентичности каждого объекта на протяжении всего видеопотока. Анализ методов (см. Табл. IV) показал, что оптимальным решением является подход tracking-by-detection на основе SORT и DeepSORT [6]. Блок-схема разработанной системы обнаружения и многообъектного трекинга представлена на Рис. 5.

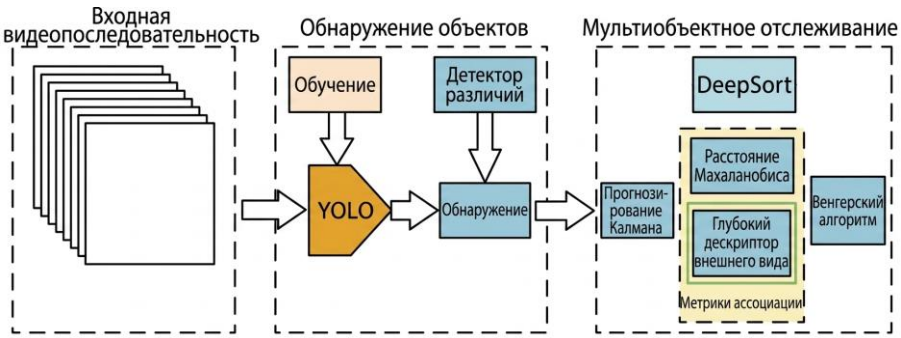

Рис. 5. Блок-схема системы обнаружения и многообъектного трекинга

ТАБЛИЦА IV
СРАВНЕНИЕ МЕТОДОВ ВИЗУАЛЬНОГО ТРЕКИНГА

Метод	Ключевая идея	Преимущества	Недостатки
KCF	Корреляция HOG	Высокая скорость	1 объект
Opt. Flow	Точки потока	Движение поверхности	Однотонные объекты
SORT (выбран)	Kalman+Hungarian	Скорость, мульти-объект	Зависит от детектора
DeepSORT (выбран)	SORT+Appearance	Устойчивость к перекрытиям	Медленнее SORT
Siamese	CNN-шаблоны	Высокая точность	Вычислительно затратен

Б. Алгоритм SORT и фильтр Калмана

SORT (Simple Online and Realtime Tracking) объединяет фильтр Калмана для предсказания положения объектов между кадрами и алгоритм Венгера для оптимального сопоставления детекций с треками [7]. Вектор состояния объекта:

$$x = [u, v, s, r, \dot{u}, \dot{v}, \dot{s}]^T, \quad (3)$$

где u, v – координаты центра объекта; s – площадь; r –соотношение сторон bbox; $\dot{u}$, $\dot{v}$, $\dot{s}$ – скорости изменения соответствующих параметров; T – операция транспонирования.

Этап прогноза:

$$\hat{x}_{t/t-1} = F\hat{x}_{t-1/t-1}, \quad (4)$$

$$P_{t/t-1} = FP_{t-1/t-1}F^T + Q, \quad (5)$$

где F – матрица перехода состояния, описывающая динамику движения (например, модель прямолинейного движения с постоянной скоростью); P – матрица ковариации; Q – ковариационная матрица шума процесса; T – символ транспонирования; ^ – символ оценивания значения, t_1/t_2 – индекс атрибуции к моменту времени t_1 на шаге времени t_2. Этот шаг позволяет предсказать, где объект должен находиться на следующем кадре, даже если наблюдений нет.

Этап обновления:

$$K_t = P_{t/t-1}H^T(HP_{t-1/t-1}H^T + R)^{-1}, \quad (6)$$

$$\hat{x}_{t/t} = \hat{x}_{t/t-1} + K_t(z_t - H\hat{x}_{t/t-1}), \quad (7)$$

$$P_{t/t} = (I - K_tH)P_{t/t-1}, \quad (8)$$

где K_t – матрица Калмана, определяющая, насколько корректировать прогноз с учётом новых измерений; H – матрица наблюдения, связывающая измерения детектора с состоянием объекта; R – ковариационная матрица шума измерений; z_t – измерения детектора (координаты центра, площадь, соотношение сторон); I – единичная матрица; T – символ транспонирования.

Метрика сопоставления треков и детекций: $1 - IoU$, минимизируемая алгоритмом Венгера за время $O(n^3)$.

В. Алгоритм DeepSORT и компонент ReID

DeepSORT дополняет SORT appearance-признаками (embedding-вектор d=128), извлекаемыми нейросетью ReID на основе MobileNetV2 [8, 9]. Комбинированная метрика:

$$C = \lambda \cdot d_{motion} + (1-\lambda) \cdot d_{appearance}, \quad (9)$$

где C – комбинированная метрика, учитывающая расстояние по движению и внешнему виду; λ – коэффициент весомости, регулирующий вклад каждого компонента; d_{motion} – расстояние Махаланобиса; $d_{appearance}$ – косинусное расстояние.

Это снижает число ошибок идентификации (ID-switch) при перекрытиях объектов.

Г. Параметрическая оптимизация

Эффективность алгоритмов SORT и DeepSORT в значительной степени определяется корректным выбором их параметров. В ходе экспериментальных исследований была проведена параметрическая оптимизация с целью обеспечения баланса между стабильностью треков, устойчивостью к перекрытиям и вычислительной эффективностью системы. Оптимизированные значения параметров приведены в Табл. V.

ТАБЛИЦА V
ОПТИМИЗИРОВАННЫЕ ПАРАМЕТРЫ SORT / DEEPSORT

Параметр	Значение	Пояснение
iou_threshold	0,3	Порог IoU для сопоставления треков
max_age	45 кадров	Число кадров до удаления трека
min_hits	2	Мин. подтверждений нового трека
conf_threshold	0,4	Порог уверенности детектора
R_scale_(s,r)	×10	Шум измерений (площадь/стороны)
P_scale_velocity	×1000	Неопределённость скорости при старте
Q_scale_velocity	×0,01	Сглаживание траектории

IV. АНАЛИЗ ЭФФЕКТИВНОСТИ СИСТЕМЫ

А. Точность нейросетевого детектора

Оценка нейросетевого детектора YOLOv8n проводилась при CPU-инференсе с оптимизацией OpenVINO. Результаты по классам приведены в Табл. VI.

ТАБЛИЦА VI
РЕЗУЛЬТАТЫ ДЕТЕКТИРОВАНИЯ YOLOV8N ПО КЛАССАМ

Класс	TP	FP	FN	Recall	Precision
Цилиндр (0)	229	5	0	1,000	0,980
Треугольник (1)	1041	12	0	1,000	0,989

Получена стопроцентная полнота обнаружения для обоих классов. Пространственная точность: Mean IoU = 0,960, AP@0,5 (Цилиндр) = 0,996, AP@0,5 (Треугольник) = 0,999, mAP@0,5 = 0,997. Среднее время CPU-инференса – 485,75 мс/изображение (2,06 FPS). Полная задержка принятия решения – 795,72 мс. Пример обнаружения при размытии и изменённом фоне показан на Рис. 6.

Б. Оценка алгоритмов трекинга

Тестирование проводилось на персональном компьютере с поддержкой GPU (CUDA), с использованием Python, PyTorch, OpenCV и детектора YOLOv9s (mAP@0,5 = 0,99). Анализировались три сценария наблюдения объектов: без перекрытий, частичное перекрытие, полное кратковременное перекрытие. Результаты работы алгоритма SORT показаны на Рис. 7, результаты DeepSORT – на Рис. 8. Дополнительная информация о сравнении алгоритмов приведена в Табл. VII.

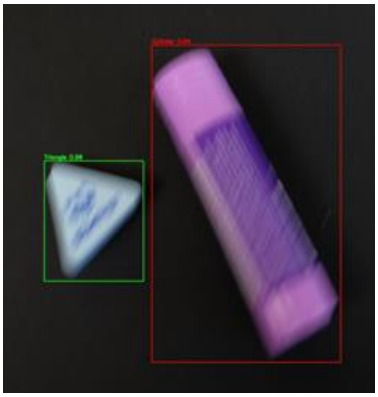

Рис. 6. Результаты обнаружения на обучающих данных с учётом размытия

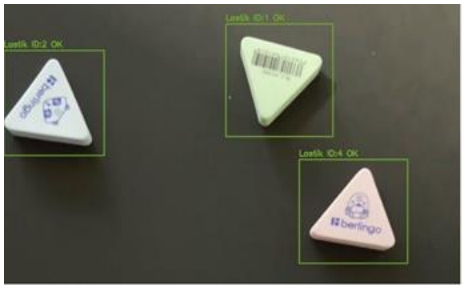

Рис. 7. Результаты работы алгоритма SORT

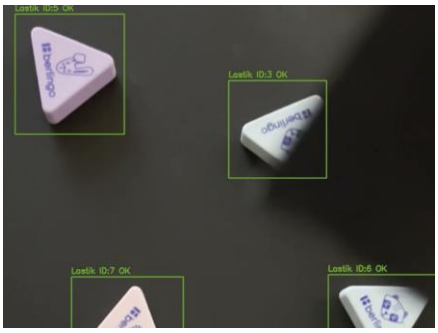

Рис. 8. Результаты работы алгоритма DeepSORT

ТАБЛИЦА VII
СРАВНЕНИЕ SORT И DEEPSORT

Критерий	SORT	DeepSORT
Скорость (FPS)	~13–15 FPS	~5–6 FPS
Точность (без перекрытий)	Высокая	Высокая
Устойчивость при перекрытии	Хорошая	Очень высокая
Восстановление ID	Удовл.	Высокое
ID-switch	Встречаются	Значительно меньше
Вычислительная нагрузка	Низкая	Высокая (ReID)

B. Устойчивость к зашумлению

Качественная оценка показала: модель стабильно работает при коэффициентах достоверности детекции 0,68–0,98. При наличии геометрически схожих элементов фона наблюдались единичные ложные срабатывания, устранимые расширением обучающей выборки. Алгоритм корректно работает при частичных перекрытиях, что подтверждает применимость в условиях высокой плотности объектов на конвейере.

ЗАКЛЮЧЕНИЕ

В работе разработана комплексная нейросетевая система технического зрения для детектирования и трекинга канцелярских товаров на конвейерной линии, объединяющая результаты двух независимых исследований. Первый автор разработал нейросетевой детектор на базе YOLOv8n с оптимизацией для CPU через OpenVINO; второй – алгоритмы визуального трекинга SORT и DeepSORT с обученным компонентом ReID на основе MobileNetV2.

Оба подхода решают принципиально разные, но взаимозависимые задачи и образуют единый замкнутый контур. Детектор отвечает за пространственную локализацию объектов на каждом отдельном кадре: обученная на 213 изображениях модель YOLOv8n достигла стопроцентной полноты обнаружения при mAP@0,5 = 0,997 и времени инференса менее 0,5 с на изображение при работе только на CPU. Трекер принимает эти покадровые детекции как входные данные и решает задачу сохранения идентичности объектов во времени: алгоритм SORT с фильтром Калмана и ассоциацией по IoU обеспечивает непрерывность траекторий при частичных и кратковременных полных перекрытиях, а обученная модель ReID в составе DeepSORT дополнительно снижает число ошибок смены идентификаторов (ID-switch) при сложных сценах.

Взаимодополняемость двух подходов проявляется на трёх уровнях. На уровне точности: детектор устраняет неопределённость положения объекта в каждом кадре, а трекер устраняет неопределённость принадлежности – он решает, какому из ранее известных объектов соответствует новая детекция. Без детектора трекер не имеет измерений для обновления фильтра Калмана; без трекера детектор выдаёт изолированные, несвязанные между кадрами ограничивающие рамки, непригодные для управления манипулятором. На уровне вычислительных ресурсов: детектор работает на CPU (2,06 FPS при инференсе через OpenVINO), что достаточно для конвейерных систем с нежёсткими требованиями к частоте кадров, тогда как SORT добавляет минимальные накладные расходы и обеспечивает трекинг в реальном времени (≥ 25 FPS); DeepSORT с ReID подключается при необходимости повышенной устойчивости идентификации, принося ограничение по скорости (~5–6 FPS), что приемлемо при наличии GPU. На уровне интеграции: выходные координаты (u, v) детектированных и отслеживаемых объектов могут быть переданы в алгоритм предиктивного управления (MPC) промышленного манипулятора после преобразования в систему координат рабочего пространства посредством матрицы проекции $T = K \cdot [R|t]$ откалиброванной камеры (конфигурация eye-in-hand), формируя замкнутый контур «зрение–управление».

Таким образом, вклад первого автора – высокоточный, оптимизированный под CPU детектор – является необходимым фундаментом, на котором работает вклад второго автора: устойчивый по времени трекер, способный сохранять идентичность объектов в условиях перекрытий и использовать траектории для управления захватным устройством. Ни один из компонентов не достигает целевой функциональности системы в отдельности; именно их совместное применение обеспечивает полный цикл обработки от сырого видеокадра до управляющего сигнала манипулятора. Вклад третьего автора – выбор методологической основы и общее руководство ходом нучной работы.

Дальнейшие исследования целесообразно направить: на расширение обучающей выборки для повышения устойчивости к вариациям фона; оптимизацию компонента ReID для работы на CPU с целью объединения точности DeepSORT и скорости SORT; верификацию интегрированной системы на реальном роботизированном комплексе с конвейерным транспортированием объектов.

ЛИТЕРАТУРА

[1] Gonzalez R.C., Woods R.E. Digital Image Processing, 2012. – 1104 c.
[2] Szeliski R. Computer Vision: Algorithms and Applications. – Springer, 2022. – 935 p.
[3] Tzutalin. LabelImg. URL: https://github.com/tzutalin/labelImg

[4] Ultralytics. YOLOv8 Documentation. URL: https://docs.ultralytics.com.
[5] Intel Corp. OpenVINO™ Toolkit. URL: https://docs.openvino.ai.
[6] Bewley A. et al. Simple Online and Realtime Tracking // IEEE ICIP. – 2016. – P. 3464–3468.
[7] FilterPy. Kalman Filters in Python. URL: https://filterpy.readthedocs.io.
[8] Wojke N. et al. DeepSORT // IEEE ICIP. – 2017. – P. 3645–3649.
[9] Sandler M. et al. MobileNetV2 // CVPR. – 2018. – P. 4510–4520.

Информация об авторах

Место Аммар – студент группы КРМО-11-24, Институт искусственного интеллекта, РТУ МИРЭА, Москва, Россия, mesto.ammar@mail.ru

Чан Минь Хоанг – студент группы КРМО-11-24, Институт искусственного интеллекта, РТУ МИРЭА, Москва, Россия, mh69titanic@gmail.com

Диане Секу Абдель Кадер – канд. техн. наук, старший научный сотрудник лаб. №90 ИПУ РАН, доцент кафедры проблем РТУ МИРЭА, Москва, Россия, diane1990@yandex.ru, ORCID: 0000-0002-8690-6422.

A neural network vision system for detecting and tracking office supplies

Ammar Mesto[1], Minh Hoang Tran[1], Sekou Diane[1,2]

[1]MIREA – Russian Technological University, Moscow, Russia
[2]Institute of Control Sciences RAS, Moscow, Russia

***Abstract* – Information support for solutions for control of autonomous manipulation robots installed on conveyor lines in order to sort transported objects requires the use of modern image analysis tools. The article proposes an approach to preparing a training set and configuring the YOLOv8n convolutional neural network for detection and the SORT/DeepSORT neural network for tracking office supplies. The generalizing ability and stability of the developed vision system to image noise are investigated. A scheme for the implementation of a neural network model in the contour of an autonomous manipulative robot control system is proposed.**

***Keywords* – vision system, convolutional neural network, autonomous manipulation robot, object detection, visual tracking, YOLOv8, SORT, DeepSORT.**

REFERENCES

[1] Gonzalez R.C., Woods R.E. Digital Image Processing, 2012. – 1104 c.
[2] Szeliski R. Computer Vision: Algorithms and Applications. – Springer, 2022. –935 p.
[3] Tzutalin. LabelImg. URL: https://github.com/tzutalin/labelImg.
[4] Ultralytics. YOLOv8 Documentation. URL: https://docs.ultralytics.com.
[5] Intel Corp. OpenVINO™ Toolkit. URL: https://docs.openvino.ai.
[6] Bewley A. et al. Simple Online and Realtime Tracking // IEEE ICIP. – 2016. – P. 3464–3468.
[7] FilterPy. Kalman Filters in Python. URL: https://filterpy.readthedocs.io.
[8] Wojke N. et al. DeepSORT // IEEE ICIP. – 2017. – P. 3645–3649.
[9] Sandler M. et al. MobileNetV2 // CVPR. – 2018. – P. 4510–4520.

Information about the authors

Ammar Mesto is a student of the KRMO-11-24 group, Institute of Artificial Intelligence, RTU MIREA, Moscow, Russia, mesto.ammar@mail.ru

Minh Hoang Tran is a student of the KRMO-11-24 group, Institute of Artificial Intelligence, RTU MIREA, Moscow, Russia, mh69titanic@gmail.com

Sekou Abdel Kader Diane is Cand. of Tech. Sciences, Senior Researcher at the lab. No. 90 ICS RAS, Associate Professor of the Department of Problems of RTU MIREA, Moscow, Russia, diane1990@yandex.ru, ORCID: 0000-0002-8690-6422.