УДК 004.93'1
DOI 10.35211/1990-5297-2024-9-292-36-43

Д. С. Князев, А. М. Макаров

РАЗРАБОТКА ПРОГРАММЫ СЛЕЖЕНИЯ И КЛАССИФИКАЦИИ ОБЪЕКТА НА РОБОТИЗИРОВАННОЙ КОНВЕЙЕРНОЙ ЛИНИИ

Волгоградский государственный технический университет
dimagold.dg53@gmail.com, amm34@mail.ru

Разработан алгоритм классификации и слежения за объектом. Алгоритм основан на использовании нейронной сети YOLOv5 для высокоточной классификации объектов в реальном времени. Разработана программная реализация алгоритма на базе языка программирования Python и библиотеки OpenCV. В ходе исследования была проведена отладка программы и оптимизация ее работы для повышения производительности и точности системы. Оценка технического решения показала, что разработанная система значительно улучшает точность и скорость обработки данных на конвейерной линии, а также обеспечивает адаптивность к изменениям в производственном процессе.

Ключевые слова: программа управления, алгоритм, нейронная сеть, слежение, классификация, компьютерное зрение, манипулятор.

D. S. Knyazev, A. M. Makarov

DEVELOPMENT OF AN OBJECT TRACKING AND CLASSIFICATION PROGRAM ON A ROBOTIC CONVEYOR LINE

Volgograd State Technical University

An algorithm for classifying and tracking the object has been developed. The algorithm is based on the use of the YOLOv5 neural network for high-precision classification of objects in real time. The developed software implementation of the algorithm is based on the Python programming language and the OpenCV library. During the work, the program was debugged and optimized to improve the performance and accuracy of the system. The evaluation of the technical solution showed that the developed system significantly improves the accuracy and speed of data processing on the conveyor line, as well as provides adaptability to changes in the production process.

Keywords: control program, algorithm, neural network, tracking, classification, computer vision, manipulator.

С развитием технологий автоматизации и робототехники требования к производственным процессам становятся все более жесткими. Современные предприятия стремятся минимизировать человеческий фактор, который может стать источником ошибок и замедлить производственные циклы. В этой связи особую актуальность приобретает разработка систем, способных автоматически распознавать, классифицировать и сортировать объекты на конвейерных линиях [1].

Использование манипуляторов с системой машинного зрения представляет собой один из наиболее эффективных подходов к решению этих задач. Такие системы могут адаптироваться к различным типам продукции, работать в непрерывном режиме и обеспечивать высокую точность распознавания объектов. Кроме того, они позволяют существенно снизить затраты на производство за счет сокращения количества брака и оптимизации рабочих процессов.

Несмотря на значительные достижения в области автоматизации, существующие решения все еще имеют ряд ограничений, связанных с точностью классификации объектов, скоростью обработки данных и адаптивностью к различным производственным условиям. Это обусловливает необходимость проведения дальнейших исследований и разработок в данной области, направленных на создание более совершенных и надежных систем слежения и классификации [2].

Техническое решение задачи сортировки объектов на конвейерной линии представлено с помощью структурной схемы (рис. 1). Система состоит из манипулятора, блока управления манипулятором, конвейера, компьютера, который будет обрабатывать информацию с камеры и передавать ее в блок управления, и камеры, закрепленной на манипуляторе. Сортировка осуществляется по типу изделия на конвейерной ленте. На входе предметы подаются в хаотичном порядке.

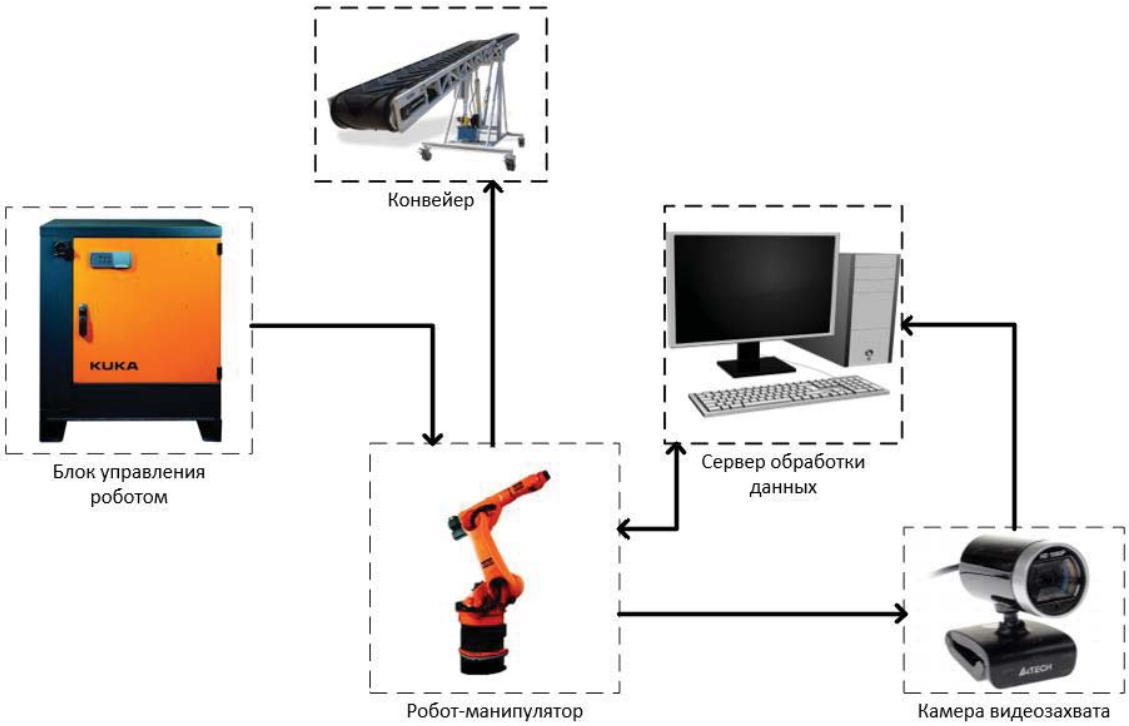

Рис. 1. Структурная комбинированная схема системы управления

Камера закреплена на захват манипулятора. Конструкция предусматривает установку на специальный фланец, который позволяет схвату полноценно функционировать с работающей камерой (рис. 2).

Рис. 2. Вариант крепления камеры к захвату манипулятора

Предлагаемая система способна классифицировать объекты на конвейерной ленте и производить сортировку в определенную группу изделий.

Целью проделанной работы была разработка алгоритма слежения для классификации объектов на конвейерной линии, а также его программная реализация.

Для классификации объекта в реальном времени требуется анализировать и осуществлять распознавание объекта на каждом кадре. Для захвата объекта после классификации необходимо преобразовать координаты объекта в систему координат манипулятора.

Рассмотрим две системы координат: систему координат OXYZ (манипулятор) с осями OX, OY, OZ и систему OUVW (камера) с осями OU, OV, OW (рис. 3).

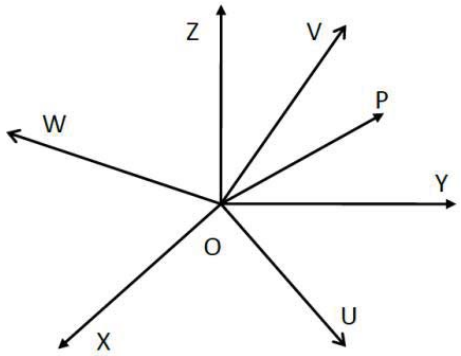

Рис. 3. Системы координат камеры и робота

Однородная матрица преобразования представляет собой матрицу размерностью 4x4, которая преобразует вектор, выраженный в однородных координатах, из одной системы отсчета в другую [3]. Однородная матрица преобразования может быть разбита на четыре подматрицы:

$$T = \begin{bmatrix} R_{3\times3} & P_{3\times1} \\ f_{1\times3} & s \end{bmatrix} = \begin{bmatrix} \text{Поворот} & \text{Сдвиг} \\ \text{Преобразование перспективы} & \text{Масштабирование} \end{bmatrix}.$$

Верхняя левая подматрица размерностью 3x3 представляет собой матрицу поворота; верхняя правая подматрица размерностью 3x1 является вектором положения начала координат повернутой системы отсчета относительно абсолютной; нижняя левая матрица размерностью 1x3 задает преобразование перспективы; четвертый диагональный элемент является глобальным масштабирующим множителем. Однородная матрица преобразования позволяет выявить геометрическую связь между связанной системой отсчета OUVW и абсолютной системой OXYZ.

В качестве алгоритма классификации используется нейронная сеть YOLOv5. YOLO (You Only Look Once) – это семейство моделей глубокого обучения, предназначенных для задач детекции объектов в реальном времени. YOLOv5 – одна из версий этой архитектуры, созданная для обеспечения высокой точности и скорости работы. YOLOv5 получила широкое признание благодаря своей эффективности и простоте использования [4].

Предпочтительнее всего было использовать данную архитектуру нейронной сети, потому что:

– в отличие от многих других нейронных сетей YOLO выполняет обнаружение объектов за один проход через сеть. Это означает, что входное изображение обрабатывается единожды, и на выходе сразу выдаются координаты и классы всех обнаруженных объектов;

– YOLOv5 имеет высокую скорость обработки данных, что делает ее подходящей для робототехнических систем.

Архитектура YOLOv5 состоит из трех основных компонентов: опорного (Backbone), промежуточного (Neck) и основного (Head) (рис. 4).

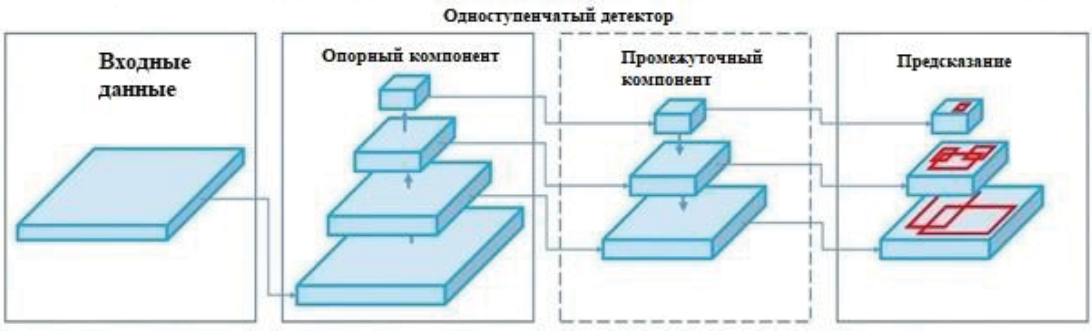

Рис. 4. Архитектура нейронной сети YOLOv5

Backbone-компонент – это часть сети, отвечающая за извлечение признаков из входного изображения. В YOLOv5 в качестве Backbone используется CSPDarknet53 – сверточная нейронная сеть, состоящая из 53 слоев, модифицированная версия. Основной операцией в сверточной сети является свертка [5]. Свертка применяет фильтр (или ядро) ко входному изобра-

жению для создания карты признаков. Формально, свертка одного канала изображения с фильтром определяется как

$$(I \cdot K)(x,y) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} I(x+i, y+j) \cdot K(i,j),$$

где $I(x,y)$ – значение пикселя входного изображения в позиции (x,y); $K(i,j)$ – значение элемента фильтра в позиции (i,j); m и n – размеры фильтра.

После свертки на выходные карты признаков применяется нелинейная функция активации для введения нелинейностей в модель, что позволяет ей выявлять сложные зависимости. Наиболее часто используемые функции активации – ReLU (RectifiedLinearUnit):

$$f(x) = \max(0, x).$$

График функции активации изображен на рис. 5.

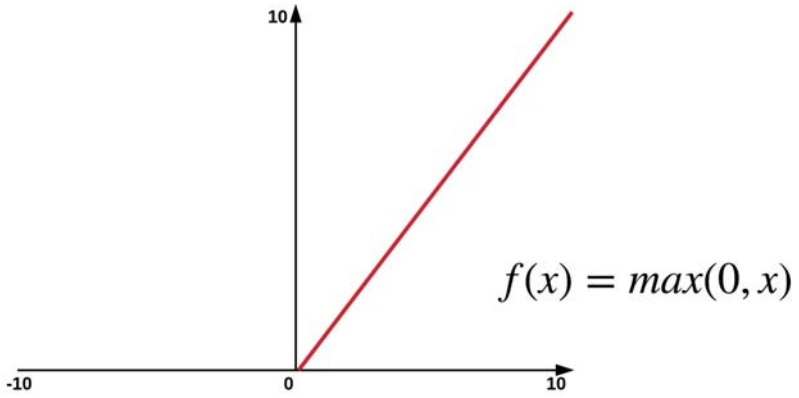

Рис. 5. Функция активации ReLU

Также в данную нейронную сеть добавляются CSP (Cross Stage Partial) блоки, которые улучшают представление признаков и снижают вычислительные затраты.

CSPDarknet53 включает в себя несколько слоев сверток и пропускных соединений, что позволяет эффективно захватывать иерархические признаки изображения.

Neck – это промежуточный компонент между Backbone и Head, который помогает объединять и обрабатывать признаки на различных масштабах. В YOLOv5 используется PANet (Path Aggregation Network) в качестве Neck, что позволяет улучшить передачу информации о признаках через сеть. PANet включает в себя структуры FPN (Feature Pyramid Network) и специальный агрегирующий путь, который помогает усилить информацию о мелких объектах и улучшить общую производительность сети.

Head отвечает за окончательное обнаружение объектов и предсказание их координат. В YOLOv5 Head состоит из нескольких слоев сверток, которые выводят три выходных вектора на разных масштабах, что позволяет обнаруживать объекты разного размера. Каждый выходной вектор содержит координаты ограничивающего прямоугольника вероятности классов и уверенность в наличии объекта.

Для эффективной классификации и захвата актуального положения объекта необходимо изменять параметры объекта в пространстве в реальном времени. Это обусловлено тем, что объекты на конвейерной линии находятся в постоянном движении, и их положение, ориентация и другие характеристики могут изменяться даже в момент захвата или подвода захвата манипулятора. Для обеспечения такой динамической обработки требуется не только классификация объектов, но и реализация алгоритма слежения «трекинга» объектов.

Трекинг объекта (object tracking) представляет собой процесс непрерывного отслеживания положения объекта в реальном времени [6]. Это включает в себя идентификацию и обновление координат объекта на основе последовательности изображений или видеокадров, получаемых от камеры (рис. 6).

Алгоритмы трекинга играют ключевую роль в системах машинного зрения, обеспечивая точное определение местоположения объекта в каждом кадре, что особенно важно для манипуляционных роботов, работающих с движущимися объектами.

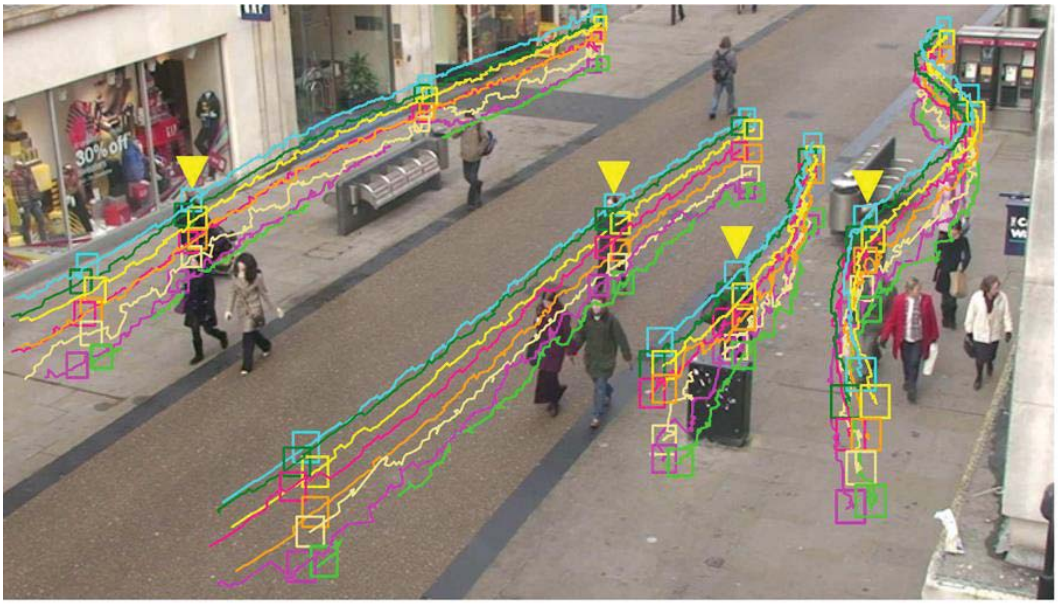

Рис. 6. Демонстрация работы алгоритма слежения за объектом

Основные этапы процесса трекинга объекта включают:

– определение начального положения объекта в первом кадре. Может быть выполнено с помощью методов классификации и сегментации, которые позволяют выделить объект среди остальных элементов сцены;

– использование моделей движения для предсказания положения объекта в последующих кадрах. Может включать в себя простые линейные модели или более сложные методы, такие как фильтр Калмана или фильтр частиц;

– сопоставление предсказанного положения объекта с фактическими данными, полученными от камеры, и коррекция координат. Может быть выполнено с использованием методов корреляции, оптического потока или других техник, обеспечивающих высокую точность;

– адаптацию модели трекинга в зависимости от изменений в освещении, форме или размере объекта, что позволяет системе оставаться устойчивой к различным внешним воздействиям.

Метод оптического потока представляет собой вычисление векторов движения для каждого пикселя изображения. Для расчета используется алгоритм Фарнебака для вычисления плотного оптического потока.

В основе дифференциального метода вычисления оптического потока Фарнебака лежит одно важное предположение. Предположим, что значения пикселей переходят из одного кадра в следующий кадр без изменений. Таким образом, делается допущение, что пиксели, относящиеся к одному и тому же объекту, могут сместиться в какую-либо сторону, но их значение останется неизменным. Глобальные условия освещения и освещенность самого движущегося объекта могут меняться от кадра к кадру, но это допущение достаточно хорошо работает на практике, особенно если речь идет о видеопотоке.

На математическом языке это предположение можно выразить следующим образом [7]:

$$I(x, y, t) = I\big(x + u_x, y + u_y, t + 1\big),$$

где I – это функция яркости пикселей от положения на кадре и времени. Таким образом, x и y – это координаты пикселя в плоскости кадра; вектор u – смещение, а t – номер кадра в последовательности.

Рис. 7. Два одномерных кадра

На примере двух одномерных кадров со смещением, где 1 пиксель в высоту и 20 пикселей в ширину (рис. 7), алгоритм Фарнебака выглядит следующим образом: каждый кадр представляет собой функцию $f_n(x)$, каждая функция относительно другой смещена (рис. 8).

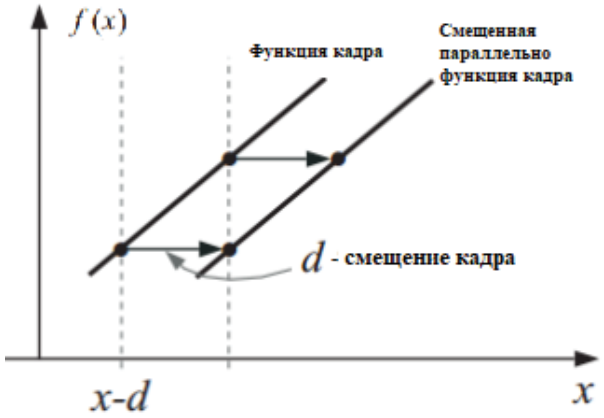

Рис. 8. Два одномерных кадра в виде функций

Функция $f_2(x)$ – смещенная параллельно функция $f_1(x)$. Таким образом, зависимость функции $f_2(x)$ имеет вид:

$$f_2(x) = f_1(x - d). \quad (1)$$

В общем виде функции $f_1(x)$и $f_2(x)$ выглядят следующим образом:

$$f_1(x) = I(x, y, t),$$
$$f_2(x) = I(x, y, t + 1).$$

Для выражения неизвестной величины d – искомое смещение, используется разложение в ряд Тейлора. Предполагая, что функция $f_1(x - d)$ хорошо аппроксимируется, можно отбросить младшие члены ряда. Тогда уравнение будет иметь вид:

$$f_1(x - d) = f_1(x) - df_1'(x). \quad (2)$$

Учитывая (1) и (2), формула искомого смещения d выглядит как

$$d = \frac{f_1(x) - f_2(x)}{f_1'(x)}.$$

Таким образом, алгоритм классификации и слежения за объектом следующий:

1. Определение пользователем объектов, которым будет присвоен приоритет при поиске по изображению.
2. Инициализация объекта видеозахвата.
3. Загрузка предварительно обученной модели YOLOv5.
4. Классификация объекта и расчет его координат контура.
5. Присвоение идентификационного номера классифицированному объекту и сохранение этого номера на протяжении всего видеопотока (слежение за объектом).
6. Обновление координат объекта до его захвата.

Для программной реализации алгоритма использовались язык программирования Python и библиотека OpenCV.

OpenCV (Open Source Computer Vision Library) – это библиотека компьютерного зрения с открытым исходным кодом, предназначенная для предоставления инструментов для обработки изображений и видео, а также для разработки приложений, связанных с машинным зрением и искусственным интеллектом [8].

Листинг основной программы представлен на рис. 9.

```
if __name__ == '__main__':
    with open("Resources/coco.names.txt") as file:
        classes = file.read().split("\n")

    # Определение классов, которым будет присвоен приоритет при поиске по изображению
    # Имена указаны в файле coco.names.txt

    video = input("Введите путь файла (URL): ")
    look_for = input("Метка класса: ").split(',')
    tracker = DeepSort(max_age=5000)
    # Delete spaces
    list_look_for = []
    for look in look_for:
        list_look_for.append(look.strip())

    classes_to_look_for = list_look_for

    # инициализация объекта видеозахвата
    video_cap = cv2.VideoCapture(video)

    # инициализация объекта для записи видео
    writer = cv2.VideoWriter("result.avi",
                             cv2.VideoWriter_fourcc(*"MJPG"), 20, (1920, 1080))

    # загрузка предварительно обученной модели YOLOv8m
    model = YOLO("Resources/yolov8m.pt")

    start_tracking_detection()
```

Рис. 9. Листинг основной программы машинного зрения

Для слежения и распознавания объектов была реализована функция для основной программы (рис. 10).

```
def object_tracking(frame_shot, array_detection):
    """
    трекинг объектов
    :param frame_shot: кадр видео
    :param array_detection: массив координат контуров распознанных объектов
    :return: массив координат объектов для маски
    """

    tracks = tracker.update_tracks(array_detection, frame=frame_shot)
    inc_track = 0
    tracker_list = []
    for track in tracks:
        if not track.is_confirmed() or inc_track > len(array_detection) - 1:
            continue

        # получение идентификатора трека и ограничивающей рамки
        track_id = track.track_id
        xmin, ymin, xmax, ymax = (int(array_detection[inc_track][0][0]),
                                  int(array_detection[inc_track][0][1]),
                                  (int(array_detection[inc_track][0][2]) +
                                   int(array_detection[inc_track][0][0])),
                                  (int(array_detection[inc_track][0][3])
                                   + int(array_detection[inc_track][0][1])))
        class_indx = int(array_detection[inc_track][2])
        score = round(array_detection[inc_track][1], 2)
```

```
        # создание ограничивающей рамки и запись
        # идентификатора объекта,
        # класса, точность прогнозирования
        cv2.rectangle(frame_shot, (xmin, ymin),
                      (xmax,
                       ymax), VIOLET, 10)
        cv2.rectangle(frame_shot, (xmin, ymin - 40),
                      (xmin + 350, ymin), VIOLET, -1)
        font_size = 1
        font = cv2.FONT_HERSHEY_SIMPLEX
        width = 3

        text = f'Id:{track_id} {((xmax - xmin)/2, (ymax - ymin)/2)}'
        cv2.putText(frame_shot, text,  org: (xmin, ymin - 10),
                    font, font_size, WHITE, width, cv2.LINE_AA)
        delta = 40
        tracker_list.append([[xmin, ymin],
                             [xmin, ymin - 40],
                             [xmin + 350, ymin - 40],
                             [xmin + 350, ymin],
                             [xmax, ymin],
                             [xmax, ymax],
                             # [xmin + 250, ymin],
                             [xmin, ymax],

                             ])
        inc_track += 1
```

Рис. 10. Функция слежения за объектом

С целью демонстрации работы алгоритма было загружено видео для классификации движущихся объектов (рис. 11).

Рис. 11. Видеоряд на входе алгоритма

На выходе изображение представляет контуры детектированных объектов, их уникальный идентификационный номер, который сохраняется на протяжении всего видеоряда, и присвоенных к ним координат центров, которые изменяются в реальном времени (рис. 12).

Рис. 12. Демонстрация алгоритма машинного зрения

Был разработан алгоритм классификации и слежения за объектом на основе библиотеки OpenCV. Проведена отладка программы и оптимизация ее работы, что позволило достичь высокой точности и скорости обработки данных. Использование манипулятора с системой машинного зрения, реализованной на базе нейронной сети YOLOv5, продемонстрировало точную классификацию и отслеживание объектов на конвейерной линии. Это решение позволяет адаптироваться к различным типам продукции и технологическим процессам, снижая долю ручного труда.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Особенности автоматизации на современных предприятиях – Студенческий научный форум. – Режим доступа : https://scienceforum.ru/2021/article/2018026266 (дата обращения: 05.05.2024).

2. Methods and models for substantiating application scenarios for the digitalization of manufacturing and business processes of network enterprises. [Электронный ресурс] / Y. F. Telnov, V. A. Kazakov, A. A. Bryzgalov, I. G. Fiodorov // Business Informatics. – 2023. – Режим доступа : https:// bi-journal.hse.ru/en/2023--4%20Vol.17/882347729.html, свободный (дата обращения: 05.05.2024).

3. Разработка управляющих программ промышленных роботов. Курс лекций. – Режим доступа : https:// www.bsuir.by/m/12_113415_1_70397.pdf (дата обращения 10.05.2024).

4. Практическое применение моделей YOLO и ResNet для обнаружения предметов на фотографиях. – Режим доступа : https://habr.com/ru/articles/761200/ (дата обращения 11.05.2024).

5. Object Detection and Tracking with OpenCV – Scaler Topics. – Режим доступа : https://www.scaler.com/ topics/object-detection-using-opencv-python/ (дата обращения: 23.01.2024).

6. *Форсайт, Д.* Компьютерное зрение. Современный подход / Д. Форсайт, Ж. Понж. – М.: Вильямс, 2004. – 960 с.

7. *Fleet, J. D.* Optical Flow Estimation / J. D. Fleet, Y. Weiss. – 2006. – Pp. 44–92.

8. *Шапиро, Л.* Компьютерное зрение : пер. с англ. / Л. Шапиро, Дж. Стокман. – М.: БИНОМ. Лаборатория знаний, 2013. – 752 с.