УДК 681.5.08
DOI 10.47367/0021-3497_2024_5_197

ОСОБЕННОСТИ ЦИФРОВОЙ ОБРАБОТКИ ОБЪЕКТОВ ПРИ ОПРЕДЕЛЕНИИ КОНТУРОВ НАТУРАЛЬНЫХ КОЖ

PECULIARITIES OF DIGITAL OBJECT PROCESSING IN DETERMINING CONTOURS OF NATURAL LEATHERS

Е.А. РЫЖКОВА, А.А. КАЗНАЧЕЕВА, А.М. КАЗАНЦЕВА

E.A. RYZHKOVA, A.A. KAZNACHEEVA, A.M. KAZANTSEVA

(Российский государственный университет им. А. Н. Косыгина (Технологии. Дизайн. Искусство))

(The Kosygin State University of Russia)

E-mail: kazantseva-am@rguk.ru

Статья посвящена вопросам, связанным с цифровой обработкой контуров натуральных кож при их распознавании. Анализируются виды цифровой обработки, варианты фильтрации, возможности применения фильтров для выделения объектов и определения контуров натуральных кож. Рассматриваются два вида фильтрации: анизотропная и рекуррентная. Детально разбирается вариант применения алгоритма Кенни для определения контура натуральной кожи. Также рассматривается компьютерная программа, написанная на языке Python, которая позволяет определять контур произвольной кожи, используя функцию cv.Canny, входящую в библиотеку OpenCV. Другой способ распознавания контуров связан с применением стандартных функций библиотеки OpenCV. В статье приводятся варианты использования обоих способов.

The article is devoted to issues related to digital processing of natural leather contours during their recognition. The types of digital processing, filtering options, and the possibilities of using filters to highlight objects and determine the contours of natural leather are analyzed. Two types of filtering are considered: anisotropic and recurrent. The option of using the Kenny algorithm to determine the contour of natural leather is analyzed in detail. A computer program written in Python is also considered, which allows determining the contour of arbitrary leather using the cv.Canny function included in the OpenCV library. Another method of contour recognition is associated with the use of standard functions of the OpenCV library. The article provides options for using both methods.

Ключевые слова: искусственный интеллект, техническое зрение, определение контуров, натуральная кожа, фильтрация изображения, метод Кенни, OpenCV.

Keywords: artificial intelligence, machine vision, contouring, genuine leather, image filtering, Canny method, OpenCV.

В современном мире, когда искусственный интеллект все больше входит в нашу жизнь, задача автоматического распознавания объектов по изображениям становится все более актуальной. Несмотря на то, что алгоритмов, решающих эту задачу, очень много, для ряда объектов, в том числе для определения контуров и пороков натуральных кож, процесс автоматизации распознавания по-прежнему актуален.

Существует два варианта постановки задачи автоматического распознавания объектов [1].

Первый вариант – решение задачи классификации. В нашем случае это определение того, что объект является натуральной кожей, и к этому же варианту относится задача определения типов дефектов этой кожи. В обоих случаях только информацией о геометрических параметрах объектов ограничиться не получится. Это позволяет отнести такие задачи к сложным задачам комплексного типа, так как признаки, по которым можно отнести тот же самый дефект к какому-либо типу, могут носить абстрактный характер. Но любая абстракция визуально реализуется в виде наличия или отсутствия у объектов специфических особенностей, которые и необходимо выделить. Такие задачи на современном этапе решаются с использованием нейросетевых технологий.

Второй вариант постановки задачи автоматического распознавания объектов – это их идентификация. В этом случае задача распознавания заключается в отождествлении изображения с одним из ранее известных. Даже с учетом того, что распознаваемые объекты имеют разную геометрическую форму, для решения поставленной задачи достаточно визуальной информации о них. Полнота и точность информации об объекте определяется способом передачи информации и разрешающей способностью средств получения изображения. При этом высококачественное изображение самого объекта не всегда требуется. Достаточно выделение некоторых его признаков, при которых обеспечивается заданная надежность распознавания. Так, для определения контура натуральной кожи нам не важна структура самой кожи, но нам важен признак, по которому мы определим границу между раскройным столом и краем кожи. В геометрической интерпретации каждому изображению соответствует точка n-мерного выборочного пространства, где n – число признаков. Таким образом, исследование задачи идентификации объектов сводится к анализу информации о наличии или отсутствии каждого из n признаков. При этом мы получаем многоальтернативную задачу с использованием большого числа признаков. Такие задачи наиболее

успешно решаются с применением современных компьютерных методов на основе искусственного интеллекта.

Несмотря на возможности искусственного интеллекта подготовка изображения к распознаванию имеет очень важное значение. И один из этапов такой подготовки – фильтрация [2]. Необходимость учета влияния помех связана с возможностью появления в кадре, кроме требуемого изображения, посторонних объектов, а также наличием шумов, возникающих при передаче изображения на обработку.

Вся методика учета помех должна быть адекватна реальной ситуации, принятой концепции распознавания, используемым для этого признакам и т.п. Поэтому целесообразно учитывать помехи уже на уровне абсолютного описания изображения [3]. При этом процесс введения шумов при моделировании алгоритмов фильтрации может заключаться в следующем. Анализируется каждый элемент матрицы изображения, и с заранее заданной вероятностью $p_п$ его состояние меняется на противоположное. Такой метод введения помех удобен для реализации на компьютере и позволяет описать уровень шумов только одним параметром – величиной $p_п$, выраженной в процентах. Качество фильтрации при этом определяется коэффициентом

$$\rho = \frac{M_п}{M_Ф}, \qquad (1)$$

где $M_п$ – число элементов в кадре, искаженном помехами, не совпадающих с соответствующими элементами эталонного изображения; $M_Ф$– число элементов отфильтрованного изображения, не совпадающих с соответствующими элементами эталонного изображения.

Рассмотрим два вида фильтрации – анизотропную и рекуррентную.

Дискретная интерпретация анизотропной фильтрации приводит изображение к соотношению

$$\widetilde{a_{i,j}} = \Lambda\left[\sum_{\nu=-\frac{N_a}{2}}^{\frac{N_a}{2}}\sum_{\xi=-\frac{N_a}{2}}^{\frac{N_a}{2}} a_{i+\nu,j+\xi}\,\omega_{\nu,\xi} - \eta\right], \qquad (2),$$

где $\widetilde{a_{i,j}}$ – элемент матрицы отфильтрованного изображения, находящийся на пересечении i-й строки и j-го столбца; $a_{i+\nu,j+\xi}$ – элемент матрицы изображения, искаженного помехами, расположенный на пересечении $(i+\nu)$-й строки и $j+\xi$-го столбца; $\omega_{\nu,\xi}$ – элемент апертуры, представляющий собой матрицу N_a x N_a, находящийся на пересечении ν-й строки и ξ-го столбца, причем размер матрицы является, как правило, нечетным числом; η – порог фильтрации, являющийся константой; Λ – пороговая функция:

$$\Lambda = \{0, \text{если x} < 0\ 1, \text{если x} \geq 0. \qquad (3)$$

Для полной фильтрации матрицу изображения надо симметрично дополнить элементами, равными нулю, так, чтобы ее результирующий размер оказался равным $(N+N_a)$x$(N+N_a)$ элементов, где N – размер изображения в элементах.

Анизотропная фильтрация ослабляет влияние отдельных пятен, не относящихся к силуэту объекта, и пробелов в силуэте. Она обеспечивает эффективную фильтрацию изображений, искаженных нормальным шумом [4].

Качество фильтрации возрастает с увеличением размера N_a, однако пропорционально квадрату N_a растет и время, затраченное на фильтрацию. Обычно размер апертуры выбирается 5×5 элементов. Это обеспечивает хорошее качество и приемлемое время.

Элементы $\omega_{\nu,\xi}$ апертуры определяются обычно исходя из нормального двумерного некорректированного кругового распределения, максимум которого совпадает с центром апертуры [5]. Такое распределение можно охарактеризовать только средним квадратичным отклонением σ_a. Это распределение усекается и аппроксимируется так, чтобы веса были нормированы:

$$\sum_{\nu=-\frac{N_a}{2}}^{\frac{N_a}{2}}\sum_{\xi=-\frac{N_a}{2}}^{\frac{N_a}{2}} \omega_{\nu,\xi}. \qquad (4)$$

Чем меньше σ_a, тем больший вес придается центральному элементу апертуры. Так,

при $\sigma_a \leq 0{,}3$ вес центрального элемента $\omega_1 = 1$, веса периферийных элементов равны нулю и эффект фильтрации отсутствует. При $\sigma_a \to \infty$ и отсутствии нормирования происходит полное стирание изображения. Апертуры с $\sigma_a < 1{,}0$ считаются узкими, а апертуры с $\sigma_a \geq 1{,}0$ – широкими.

Теоретический анализ анизотропной фильтрации для определения оптимального значения порога фильтрации η можно выполнить только для конкретного изображения, но из общих соображений ясно, что при аддитивном нормальном шуме значение η должно находиться вблизи 0,5.

Эффективность сглаживания может быть повышена при использовании рекуррентного фильтра. В этом случае в процессе фильтрации участвуют не только элементы исходного изображения, но и элементы уже сглаженного изображения. В этом случае соотношение (2) принимает вид:

$$\widetilde{a_{i,j}}^{(p)} = \Lambda\left[\sum_{\nu=-\frac{N_a}{2}}^{\frac{N_a}{2}} \sum_{\xi=-\frac{N_a}{2}}^{\frac{N_a}{2}} a^*_{i+\nu,j+\xi}\omega_{\nu,\xi} - \eta\right], \quad (5)$$

где

$$a^*_{i+\nu,j+\xi} = \begin{cases} a_{i+\nu,j+\xi}, \text{если } \left[\left(\nu = 1,2,\dots,\frac{N_a}{2}; \xi = -\frac{N_a}{2},\dots,\frac{N_a}{2}\right), \left(\nu = 0; \xi = 0,1,..,\frac{N_a}{2}\right)\right], \\ \widetilde{a_{i,j}}^{(p)}, \text{если } \left[\left(\nu = -1,-2,\dots,-\frac{N_a}{2}; \xi = -\frac{N_a}{2},\dots,\frac{N_a}{2}\right), \left(\nu = 0; \xi = -0,-1,..,-\frac{N_a}{2}\right)\right]. \end{cases} \quad (6)$$

Рекуррентная фильтрация имеет преимущества в том случае, когда уровень шумов не превышает 15…20%. Другим достоинством этого фильтра является то, что такое сглаживание более чувствительно к порогу η.

Таким образом, при интенсивности помех, не превышающей 15…20%, целесообразно использовать либо анизотропный фильтр с узкой апертурой, либо рекуррентный фильтр с той же апертурой, в противном случае следует применять анизотропный фильтр с широкой апертурой. Если же на фильтрацию можно выделить дополнительное время, то независимо от ожидаемого уровня шумов лучшее сглаживание обеспечит фильтр с автоматической регулировкой апертуры.

На основе этих двух типов фильтров разработано большое количество различных видов фильтров. Один из таких фильтров, позволяющий определить контур натуральной кожи, разработан на основе метода Кенни (Canny), который является одним из эффективных для обработки изображений [6]. Суть метода заключается в извлечении полезной структурной информации из *n*-го количества объектов, что значительно сокращает объем обрабатываемых данных, и основывается на трех критериях: 1) повышение отношения сигнал/шум; 2) правильное определение положения границы; 3) единственный отклик на одну границу.

Процесс обнаружения по методу Кенни можно разбить на 5 этапов [7]:

Этап I. Применение фильтра Гаусса. Устраняются шумы на изображении, что в свою очередь предотвращает ложное обнаружение. Чтобы сгладить изображение, ядро фильтра сворачивается вместе с изображением.

Этап II. Нахождение градиентов интенсивности изображения. На данном этапе находятся градиенты интенсивности изображения при помощи оператора Собеля. Этот оператор определяет векторы градиентов яркости в каждой точке изображения, которые в свою очередь помогут определить углы или направления краев, так как направление градиента всегда перпендикулярно краю.

Этап III. Применение порогового значения величины градиента или подавление границы отсечки. На изображении ищутся места с наиболее резким изменением значения интенсивности. Обычно поиск проводится матрицей 3x3, таким образом, центральный пиксель сравнивается с крае-

выми. Если разница между двумя «диаметрально» противоположными пикселями образует максимум, тогда центральный пиксель считается краем, в противном случае обращается в ноль.

Этап IV. Применение двойного порога. Далее необходимо отсечь некоторые граничные пиксели, которые вызваны шумом и изменением цвета. В данном случае, если значение градиента граничного пикселя превышает верхнее пороговое значение, такой пиксель помечается как пиксель с сильным краем, в свою очередь, если значение лежит между высоким и нижним, такой пиксель помечается как слабый. Если же значение пикселя лежит ниже нижней границы, тогда такой пиксель подавляется, то есть приравнивается к 0.

Этап V. Отслеживание краев с помощью гистерезиса (подавление всех ребер, которые являются слабыми и не связаны с остальными краями). Здесь отсекаются лишние линии следующим образом: если значение линии лежит ниже минимального значения или находится строго между максимальным и минимальным значением, тогда линия подавляется; если линия находится выше максимального значения, такая линия оставляется и точно считается краем; если часть линии находится между максимальным и минимальным значением, но при этом другая часть линии находится выше максимального, тогда линия оставляется, считаясь продолжением точной линии.

Для решения задачи обнаружения контура кожи применена функция cv.Canny. Она входит в библиотеку OpenCV и принимает следующие аргументы: исходное изображение, минимальное и максимальное значения порогов, размер ядра для фильтра Собеля (по умолчанию равен 3). Последний аргумент определяет уравнение для нахождения величины градиента [8]. Реализация программного кода выполнена на языке Python.

Код начинается с импорта библиотек opencv и numpy:

```
import numpy as np
import cv2 as cv
```

Далее определяется функция nothing, которая применяется для создания ползунков:

```
def nothing ():
pass
```

Затем создается окно под именем result и ползунки для определения максимального и минимального порогов. В функции cv.createTrackbar последним аргументом является nothing, который будет выполняться каждый раз при изменении значения ползунка. Так как в текущем случае потребность в каких-либо действиях отсутствует, аргумент nothing просто пропускает действие:

```
cv.namedWindow('result')
cv.createTrackbar('minVal', 'result', 0, 255, nothing)
cv.createTrackbar('maxVal', 'result', 255, 255, nothing)
```

На следующем шаге создается цикл while для ожидания нажатия кнопки завершения процесса. В теле цикла выполняются следующие действия. Сначала производится считывание изображения и преобразование его в градации серого функцией cv.imread. Также проверяется действительное открытие изображения, в случае отсутствия указанного файла выдается ошибка:

```
while True:
img= cv.imread('C:/NN/Images /Leather_03.jpg', cv.IMREAD_GRAYSCALE)
assert img is not None, "file could not be read, check with os.path.exists()"
```

Далее двум переменным присваиваются значения, выставленные ползунками:

```
minVal = cv.getTrackbarPos('minVal', 'result')
maxVal = cv.getTrackbarPos('maxVal', 'result')
```

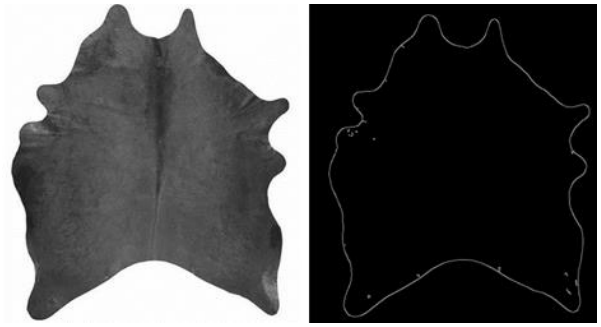
Затем применяется функция cv.Canny:

```
edges = cv.Canny(img, minVal, maxVal, True)
```

В конце алгоритма выводятся окна Image, отображающее начальную картинку, и result, отображающее два ползунка и картинку с полученными краями, также выполняется проверка нажатия клавиши Esc для выхода из цикла и завершения программы:

```
cv.imshow('Image', img)
cv.imshow('result', edges)
k = cv.waitKey(5)
if k == 27:
cv.destroyAllWindows()
```

На рис. 1 представлено первоначальное изображение кожи, а также полученное изображение с наложением найденного контура.



а) б)

Рис. 1

Еще одним способом распознавания контура объекта является применение стандартных функций findContour() и drawContour() библиотеки OpenCV [9]. На первом этапе выполнения программы импортируются библиотеки sys, numpy, cv2 и класс settings. Как уже было отмечено выше, контуром называется кривая, соединяющая непрерывную последовательность точек, имеющих одинаковый цвет или яркость [10, 11]. Таким образом, следует отсечь ненужные цвета на изображении. Для этого рекомендуется производить преобразование изображения из модели RGB (Red Green Blue) в HSV (Hue Saturation Value) для более точного нахождения границ с помощью функции cv.cvtColor(). RGB – это цветовая модель, описывающая способ кодирования цвета или цветовоспроизведения с помощью трех цветов. HSV – это модель, где координатами цвета выступают: hue – цветовой тон, saturation – насыщенность и value – яркость. Для более четкого распознавания рациональнее использовать модель HSV, так как она построена не только на совмещении цвета.

Преобразование изображения происходит по следующему алгоритму [12].

Производится инициализация объекта settings, где задается подбор максимального и минимального значений порогов:

```
get_values = prefer.settings(fn)
h_values = get_values.values()
```

Затем создаются окна с изображением и настройками:

```
cv.namedWindow("result")
cv.namedWindow("settings")
```

Далее определяются шесть ползунков для настройки максимального и минимального значений порогов, а затем объявляется переменная, содержащая массив следующих значений:

```
cv.createTrackbar('h1', 'settings', 0, 255, nothing)
cv.createTrackbar('s1', 'settings', 0, 255, nothing)
cv.createTrackbar('v1', 'settings', 0, 255, nothing)
cv.createTrackbar('h2', 'settings', 255, 255, nothing)
cv.createTrackbar('s2', 'settings', 255, 255, nothing)
cv.createTrackbar('v2', 'settings', 255, 255, nothing)
crange = [0, 0, 0, 0, 0, 0]
```

В результате сформирован цикл для настройки пороговых значений, который будет выполняться до нажатия клавиши Esc. В цикле выполняется считывание изображения, преобразование его в HSV формат, затем считываются значения ползунков:

```
img = cv.imread(fn)
hsv = cv.cvtColor(img, cv.COLOR_BGR2HSV)
h1 = cv.getTrackbarPos('h1', 'settings')
s1 = cv.getTrackbarPos('s1', 'settings')
v1 = cv.getTrackbarPos('v1', 'settings')
h2 = cv.getTrackbarPos('h2', 'settings')
s2 = cv.getTrackbarPos('s2', 'settings')
v2 = cv.getTrackbarPos('v2', 'settings')
```

На следующем этапе формируется максимальный и минимальный порог изображения:

```
h_min = np.array((h1, s1, v1), np.uint8)
h_max = np.array((h2, s2, v2), np.uint8)
```

Накладывается фильтр с помощью функции cv.inRange(), где в качестве аргументов указываются цветовая модель и выставленные максимум и минимум порогов, а также выводится полученное изображение в окне result:

```
thresh = cv.inRange(hsv, h_min, h_max)
cv.imshow('result', thresh)
```

Производится проверка нажатия клавиши Esc, и, если она нажата, уничтожаются окна, затем возвращаются значения максимума и минимума порогов:

```
        ch = cv.waitKey(5)
        if ch == 27:
            hv_min = h_min
            hv_max = h_max
            break
    cv.destroyAllWindows()
    return (hv_min, hv_max)
```

Чтобы выбрать максимум и минимум порогов, следует придерживаться нескольких правил [13]. В первую очередь максимальный порог должен быть больше минимального, иначе контур или объект должен быть белого цвета, а все остальное черного. Пример настройки изображения показан на рис. 2. Каждый ползунок создается для выбора значения соответствующей координаты цветовой модели: h – hue, s – saturation, v – value. Для удобства подбора максимального и минимального порогов ползунки распределены так: h1, s1, v1 – отвечают за значения минимального порога, h2, s2, v2 – за значения максимального. Три нижних ползунка двигаются справа налево, а верхние слева направо.

а) б)

Рис. 2

В main преобразуем изображение в цветовую модель HSV, где применяем фильтр с полученными значениями максимума и минимума порогов:

```
    hsv = cv.cvtColor(img, cv.COLOR_BGR2HSV)
    thresh = cv.inRange(hsv, h_values[0], h_values[1])
```

Далее выполняем поиск контуров функцией cv.findContours(), записываем их в переменные с самим изображением контура, а также иерархию контуров:

```
    contours0, hierarchy = cv.findContours(thresh.copy(), cv.RETR_TREE, cv.CHAIN_APPROX_SIMPLE)
```

Затем для вывода изображения с наложением контура, а также разбивки его по иерархии воспользуемся тремя функциями update(), update_index() и update_layer(v):

```
    index = 0
    layer = 0
    def update():
        vis = img.copy()
        cv.drawContours(vis, contours0, index, (255, 0, 0), 2, cv.LINE_AA, hierarchy, layer)
        cv.imshow('contours', vis)
    def update_index(v):
        global index
        index = v - 1
        update()
    def update_layer(v):
        global layer
        layer = v
        update()
```

Рис. 3

Далее обновляются индекс и слои, создаются ползунки для их задания, а затем программа переходит в режим ожидания закрытия окна. После выполнения настроек появится окно с первоначальным изображением, где, двигая ползунки layers и contour, можно увидеть все найденные контуры. Пример изображен на рис. 3.

ВЫВОДЫ

Выполнив код программы и произведя необходимые манипуляции, получили контуры образца кожи. Распознавание контуров при помощи метода Кенни и диапазона порогов выполнено корректно. В дальнейшем планируется протестировать использование автоматического подбора порогов, а также выполнение операций по сегментированию изображения.

ЛИТЕРАТУРА

1. *Гонсалес Р., Вудс Р.* Цифровая обработка изображений. М.: Техносфера, 2005. 1007 с.

2. *Анисимов Б.В.* Распознавание и цифровая обработка изображений. М.: Высш. школа, 1983. 295 с.

3. *Nafstad O., Gronstol H.* Variation in the Level of Grain Defect Light Flecks and Sports on Cattle Hides // Acta vet. scand. 2001, 42, P. 91…98.

4. *Mwondu J.* Training manual on improved production and preservation techniques of hides and skins. – URL: researchgate.net/publication/324844427.

5. *Крейман Г.* Биологическое и компьютерное зрение. М.: ДМК-Пресс, 2022. 314 с. – ISBN 978-5-93-700100-9.

6. *Гарсия Г.Б.* Обработка изображений с помощью OpenCV. М.: ДМК-Пресс, 2016. 111 с. – ISBN 978-5-97-060387-1.

7. *Солем Я.* Программирование компьютерного зрения на языке Python. М.: ДМК-Пресс, 2016. 312 с. – ISBN 978-5-97-060200-3.

8. *Калачев Н.М., Казначеева А.А., Рыжкова Е.А.* Технологический процесс раскроя натуральных кож применительно к задачам автоматизации контроля качества // Сборник научных трудов кафедры автоматики и промышленной электроники Российского государственного университета им. А.Н. Косыгина. М.: РГУ им. А.Н. Косыгина, 2023. С. 19…24.

9. *Клетте Р.* Компьютерное зрение. Теория и алгоритмы. М.: ДМК-Пресс, 2019. 506 с. – ISBN 978-5-97-060702-2.

10. *Жиляков Е.Г., Черноморец А.А.* Оптимальная фильтрация изображений на основе частотных представлений // Вопросы радиоэлектроники. Сер. ЭВТ. 2008. Вып.1. С. 118…132.

11. *Murashko F.V., Ryzhkova E.A., Vlasenko O.M.* Search for an object in an image by image difference method to find contours of a natural leather blank in pattern cutting processs // Fibre Chemistry. 2018. Т. 50. № 1. С. 38…41.

12. *Murashko F.V., Ryzhkova E.A., Vlasenko O.M.* Development of a system for calibration of a machine vision complex based on the use of colors bands // Fibre Chemistry. 2018. Т. 49. № 6. С. 394…399.

REFERENCES

1. *Gonzalez R., Woods R.* Digital image processing. Moscow: Technosphere, 2005. 1007 p.

2. *Anisimov B.V.* Recognition and digital image processing. Moscow: Higher School, 1983. 295 p.

3. *Nafstad O., Gronstol H.* Changes in the level of grain defects, light spots and spots on cattle hides // Acta vet. scand. 2001, 42, P. 91…98.

4. *Mvondu J.* Training manual on improved production and preservation techniques of hides and skins. – URL: researchgate.net/publication/324844427.

5. *Kreiman G.* Biological and computer vision. Moscow: DMK-Press, 2022. 314 p. – ISBN 978-5-93-700100-9.

6. *Garcia G.B.* Image processing using OpenCV. Moscow: DMK-Press, 2016. 111 p. – ISBN 978-5-97-060387-1.

7. *Solem Ya.* Computer-level programming in Python. Moscow: DMK-Press, 2016. 312 p. – ISBN 978-5-97-060200-3.

8. *Kalachev N.M., Kaznacheeva A.A., Ryzhkova E.A.* Technological process of cutting natural skins in relation to the tasks of automation of quality control // Collection of scientific papers of the Department of Automation and Industrial Electronics of the Kosygin Russian State University. M.: RSU im. A.N. Kosygina, 2023. P. 19…24.

9. *Klette R.* Computer vision. Theory and algorithms. Moscow: DMK-Press, 2019. 506 p. – ISBN 978-5-97-060702-2.

10. *Zhilyakov, E.G., Chernomorets, A.A.* Optimal image filtering on the basis of frequency representations // Voprosy radioelectroniki. Ser. EVT. 2008. Issue 1. P. 118...132.

11. *Murashko F.V., Ryzhkova E.A., Vlasenko O.M.* Search for an object in an image using the image difference method to determine the contours of a billet made of genuine leather in cutting processes // Fiber chemistry. 2018. Vol. 50. No. 1. P. 38…41.

12. *Murashko F.V., Ryzhkova E.A., Vlasenko O.M.* Development of a calibration system for a machine vision complex based on the use of color stripes // Fiber chemistry. 2018. Vol. 49. No. 6. P. 394…399.

Рекомендована кафедрой автоматики и промышленной электроники РГУ им. А.Н. Косыгина. Поступила 23.05.24.