

ISSN 1561-8323 (Print)  
ISSN 2524-2431 (Online)

**ИНФОРМАТИКА**  
**INFORMATICS**

УДК 004  
<https://doi.org/10.29235/1561-8323-2024-68-2-105-111>

Поступило в редакцию 06.02.2024  
Received 06.02.2024

**Б. А. Залесский, В. А. Иванюкович**

*Объединенный институт проблем информатики Национальной академии наук Беларуси,  
Минск, Республика Беларусь*

**АЛГОРИТМ СОПРОВОЖДЕНИЯ ОБЪЕКТА,  
НАБЛЮДАЕМОГО ВИДЕОКАМЕРОЙ**

*(Представлено членом-корреспондентом А. В. Тузиковым)*

**Аннотация.** Разработан алгоритм отслеживания объекта, наблюдаемого на кадрах видеопотока. Особенность алгоритма заключается в автоматическом обнаружении и захвате объекта одного из заранее заданных типов, его дальнейшем надежном сопровождении, быстром повторном захвате сопровождаемого объекта в случае срыва сопровождения, захвате другого объекта нужного типа при исчезновении сопровождаемого объекта. Обнаружение объекта интереса на кадрах видео осуществляется с помощью нейронной сети-детектора, а сопровождение – разработанным алгоритмом.

**Ключевые слова:** видеопоток, обнаружение объекта, отслеживание объекта, нейронная сеть-детектор, трекер

**Для цитирования.** Залесский, Б. А. Алгоритм сопровождения объекта, наблюдаемого видеокамерой / Б. А. Залесский, В. А. Иванюкович // Докл. Нац. акад. наук Беларуси. – 2024. – Т. 68, № 2. – С. 105–111. <https://doi.org/10.29235/1561-8323-2024-68-2-105-111>

**Boris A. Zalesky, Vladimir A. Ivanyukovich**

*United Institute of Informatics Problems of the National Academy of Sciences of Belarus,  
Minsk, Republic of Belarus*

**ALGORITHM FOR TRACKING AN OBJECT OBSERVED BY A VIDEO CAMERA**

*(Communicated by Corresponding Member Aleksandr V. Tuzikov)*

**Abstract.** An algorithm for tracking an object observed on video frames is presented. The specific feature of the constructed algorithm is the automatic detection and capture of an object of one of predetermined types, its further reliable tracking, the rapid re-capture of the tracked object in the case of a failure of tracking, the capture of another object of desired type if the tracked object disappears. An object of interest on video frames is detected using a neural network detector, whereas tracking is performed by the developed algorithm.

**Keywords:** video, object detection, object tracking, neural network detector, tracker

**For citation.** Zalesky B. A., Ivanyukovich V. A. Algorithm for tracking an object observed by a video camera. *Doklady Natsional'noi akademii nauk Belarusi = Doklady of the National Academy of Sciences of Belarus*, 2024, vol. 68, no. 2, pp. 105–111 (in Russian). <https://doi.org/10.29235/1561-8323-2024-68-2-105-111>

**Введение.** Задачи сопровождения объектов, наблюдаемых на кадрах видеопотока, в настоящее время являются одними из важнейших задач компьютерного зрения. Об этом, в частности, свидетельствует большое число научных работ, опубликованных по данной тематике в последние годы [1–3], появившиеся многочисленные программные реализации разработанных новых алгоритмов отслеживания [4; 5], в том числе в широко известной библиотеке программ Torch-

vision<sup>1</sup>, а также примеры их применения для решения практических задач. Новые результаты позволили существенно повысить надежность и точность решения задачи. Часть из них успешно применяется на практике, например, для контроля безопасности и недопустимой активности, контроля движения транспорта, навигации роботов и автономного транспорта, медицинской диагностики, анализа спортивных игр и т. д.

Рассматривается задача сопровождения объекта одного из заранее заданных типов, наблюдаемого на кадрах видеопотока, в следующей формулировке: требуется в автоматическом режиме обнаружить на кадре объект одного из заранее заданных типов, захватить и сопровождать его до момента исчезновения. В случае срыва сопровождения выполнить повторное обнаружение и захват сопровождаемого ранее или другого объекта этого же или другого заданного заранее типа.

Данная задача не является новой в компьютерном зрении. Многие алгоритмы, предназначенные для ее решения, например, одни из лучших на данный момент трекеров нескольких объектов (multi object trackers) DeepSort и ByteTrack могут быть разделены на два блока. Один из блоков обеспечивает автоматическое обнаружение объектов одного из наперед заданных типов, а второй – обеспечивает сопровождение обнаруженных объектов. В первом блоке для обнаружения объектов интереса в основном используются нейронные сети-детекторы. Во втором блоке выполняется сопровождение объекта на основе анализа его признаков и траектории движения на кадрах.

Предлагаемый алгоритм также состоит из упомянутых блоков. В первом блоке используется нейронная сеть-детектор для обнаружения объекта интереса, а во втором выполняется его сопровождение. Новизна алгоритма заключается в более сложном взаимодействии блоков обнаружения и слежения, а также в непосредственной реализации блока слежения, основанной на анализе оптического потока.

Реализация алгоритма в виде двух блоков позволяет адаптировать его для разных вычислителей, начиная от миниатюрных, имеющих ARM-архитектуру и небольшой нейросетевой процессор, заканчивая более мощными, в том числе современными персональными компьютерами, вычислительных мощностей которых вполне достаточно для выполнения разработанного алгоритма в режиме реального времени.

Для автоматического обнаружения объектов могут использоваться разные детекторы, выполняющиеся на вычислителе в режиме реального времени, однако очевидно, что более точные и надежные из них обеспечат большую точность и надежность сопровождения, поэтому в построенном алгоритме для обнаружения объектов использовались модели nano, small, medium, large и extralarge лучшего на момент написания статьи нейросетевого детектора YOLOv8<sup>2</sup>. Более сложные модели лучше обнаруживают объекты, но медленнее вычисляются, поэтому выбралась наиболее сложная модель, подходящая по времени выполнения для используемых CPU и GPU. На персональном компьютере PC i7-6700 CPU 3.40GHz с видеокартой Nvidia RTX 3060 использовалась модель YOLOv8 large.

С целью тестирования разработанный алгоритм применялся для решения актуальной задачи автоматического обнаружения и сопровождения находящихся в воздухе коптеров, наблюдаемых видеокамерой с земной поверхности. Для этого было сформировано и размечено четыре набора данных.

Для обучения нейронной сети YOLOv8 использовался первый набор данных, состоящий изначально из 1502 изображений. Этот набор был аугментирован с использованием масштабирования и поворотов. Полученные после аугментации 22237 изображений были разбиты случайным образом на тренировочную, валидационную и тестовую выборки. Тренировочная и валидационная выборки использовались для обучения модели YOLOv8 large. Тестовая выборка использовалась для оценки характеристик обученной модели.

<sup>1</sup> PyTorch [Electronic resource]. – Mode of access: <https://pytorch.org/vision/stable/index.html>. – Date of access: 05.01.2024.

<sup>2</sup> Документация Ultralytics YOLOv8 [Electronic resource]. – Mode of access: <https://docs.ultralytics.com/ru/>. – Date of access: 05.01.2024.

Оставшиеся три набора данных использовались для тестирования и оценки разработанного алгоритма.

**Алгоритм сопровождения объекта** можно представить в виде двух блоков. Первый блок выполняет автоматическое обнаружение объектов заданного типа. Второй – обеспечивает сопровождение одного из обнаруженных объектов. Особенность алгоритма заключается во взаимодействии блоков обнаружения и сопровождения. В начале выполнения алгоритма осуществляется обнаружение объекта интереса на кадрах выбранным детектором (в нашем случае нейронной сетью YOLOv8) до тех пор, пока объект не будет обнаружен. Сразу после обнаружения объекта осуществляется его захват и запускается блок сопровождения захваченного объекта с помощью оптического потока на заранее заданном количестве последовательных кадров (при реальном применении алгоритма количество кадров изменялось от 0 до 10. Если произошел срыв сопровождения, алгоритм перезапускается с его начала). После этого вновь запускается детектор и выполняется проверка наличия прямоугольной области одного из обнаруженных объектов заданного типа вблизи прямоугольной области вероятного нахождения сопровождаемого объекта. Если упомянутые области пересекаются, опять запускается сопровождение объекта оптическим потоком. В противном случае продолжается сопровождение объекта оптическим потоком на небольшом количестве кадров. При этом на каждом кадре запускается детектор объектов и проверяется пересечение прямоугольной области, ограничивающей сопровождаемый оптическим потоком объект, с прямоугольной областью, ограничивающей объект заданного типа, найденный детектором. Если на одном из кадров произошло пересечение упомянутых областей, выполняется блок сопровождения оптическим потоком без использования детектора, иначе алгоритм запускается с самого начала.

Напомним о том, что обнаружение объектов в алгоритме может выполняться с помощью произвольных детекторов, однако в нашем случае при разработке, тестировании и реальном применении использовались различные модели нового нейросетевого детектора объектов YOLOv8.

В алгоритме используется поиск объекта по областям кадров. Части этих областей, выходящие за пределы кадра, не учитываются при поиске, об этом не будет упоминаться в описании алгоритма. Введем следующие обозначения:  $t$  – номер текущего кадра;  $I_t$  – текущий кадр;  $B$  – прямоугольник, ограничивающий найденный объект (bounding box);  $I_{\text{search}}$  – изображение области поиска объекта с помощью YOLO;  $H$  – массив, содержащий последние последовательные успешно найденные ограничивающие прямоугольники  $B$  сопровождаемого объекта в хронологическом порядке (элемент массива с индексом 0 – самый старый);  $d$  – максимально допустимая длина массива  $H$  (размерность  $\dim(H) \leq d$ );  $q$  – периодичность обнаружения объекта детектором YOLO, измеренная в кадрах;  $m_{\text{LK}}$  – счетчик числа успешных предсказаний положения объекта оптическим потоком Лукаса–Канаде после последнего успешного обнаружения, выполненного YOLO ( $m_{\text{LK}} \leq q$ );  $m_{\text{nodet}}$  – счетчик числа кадров после неудачного обнаружения объекта детектором, на которых положение объекта предсказывается оптическим потоком;  $b$  – максимально допустимое значение переменной  $m_{\text{nodet}}$ ;  $B_{\text{prev}}$  – прямоугольник  $B$ , ограничивающий объект, полученный на предыдущем кадре (если он был получен). До нахождения ограничивающего прямоугольника  $B$  на текущем кадре  $B_{\text{prev}}$  является последним элементом массива  $H$  (если массив не пустой);  $B_{\text{search}}$  – прямоугольник, задающий область поиска, которая используется для обнаружения объекта в случае его потери;  $S$  – история величин сдвигов объекта между кадрами, заданная массивом, содержащим расстояния между центрами последовательных найденных прямоугольников  $B$ , ограничивающих сопровождаемый объект, из массива  $H$ ;  $P_0$  – массив координат характерных точек  $p_0$  на предыдущем кадре, используемых для вычисления оптического потока на кадре;  $P_1$  – массив точек  $p_0$  на текущем кадре, найденных оптическим потоком, которые являются образами точек  $p_0$ ;  $\Delta_{\text{max}}$  – максимально допустимое расстояние между центрами двух последовательных прямоугольников  $B$ , ограничивающих сопровождаемый объект, при отслеживании объекта. Также максимально допустимая величина сдвига оптического потока в точках  $p_0$ .

Далее приведено пошаговое описание построенного алгоритма отслеживания объекта. Фигурные скобки используются в пошаговом описании после условий для выделения всех действий, которые должны быть выполнены только в случае, когда эти условия выполняются.

Шаг 1. Задать начальные значения  $t = 0$ ,  $m_{LK} = 0$ ,  $B_{prev} = \emptyset$ ,  $H = \emptyset$ ,  $S = \emptyset$ .

Шаг 2. Читать текущий кадр  $I_t$ . Задать область детекции  $I_{search} = I_t$ .

Шаг 3. Если  $I_t = \emptyset$ , STOP.

Шаг 4. Если отслеживаемая траектория отсутствует ( $H = \emptyset$ ), перейти на шаг 8.

Шаг 5. Если счетчик  $m_{LK}$  меньше  $q$ , перейти на шаг 18.

Шаг 6. Если счетчик  $m_{nodet}$  больше 0, {на текущем кадре  $I_t$  выделить прямоугольную область, границы которой были определены на шаге 17 для предыдущего кадра в виде  $B_{search}$  и присвоить  $I_{search}$ . Перейти на шаг 8}.

Шаг 7. Если площадь  $B_{prev}$  меньше 5000, на текущем кадре  $I_t$  выделить прямоугольную область поиска с центром, совпадающим с центром  $B_{prev}$  и сторонами, превышающими на 200 пикселей стороны  $B_{prev}$ . Присвоить выделенную область  $I_{search}$ .

Шаг 8. Положить  $m_{LK} = 0$ . Выполнить обнаружение объектов на области  $I_{search}$ . Выбрать ограничивающие прямоугольники, найденные Yolo, с уровнем доверия  $\text{conf} \geq 0.1$ .

Шаг 9. Если количество ограничивающих прямоугольников, найденных на шаге 8 равно 0, перейти на шаг 14.

Шаг 10. Если в настоящее время объект не отслеживается, т. е.  $\dim(H) = 0$ , {занести в массив  $H$  один из найденных ограничивающих прямоугольников  $B$  с наибольшим уровнем доверия  $\text{conf}$ . Присвоить  $m_{nodet} = 0$ . Перейти на шаг 2}.

Шаг 11. Среди найденных ограничивающих прямоугольников  $B$  выбрать один, удовлетворяющий условию

$$B' = \arg \max_B (IoU(B, B_{prev})).$$

Шаг 12. Если  $IoU(B', B_{prev}) = 0$ , перейти на шаг 15.

Шаг 13. Добавить  $B'$  в массив  $H$ , а в историю величин сдвигов  $S$  добавить сдвиг между найденным  $B'$  и  $B_{prev}$ . Присвоить  $m_{nodet} = 0$ . Перейти на шаг 25.

Шаг 14. Если объект не сопровождается, т. е.  $H = \emptyset$ , {присвоить  $t := t + 1$ , перейти на шаг 2}.

Шаг 15. Если  $m_{nodet} = b$ , {удалить массивы  $H$  и  $S$ , присвоить  $t := t + 1$ , перейти на шаг 2}.

Шаг 16. Если счетчик  $m_{nodet} = 0$ , задать прямоугольную область поиска  $B_{search} = B_{prev}$ .

Шаг 17. Увеличить длины сторон прямоугольника  $B_{search}$  на 100 пикселей в каждую сторону. Положить  $m_{nodet} := m_{nodet} + 1$ , счетчик  $m_{LK} := q - 1$ .

Шаг 18. Присвоить  $m_{LK} := m_{LK} + 1$ . Задать  $\Delta_{max} = 0.5 \text{diag}(B_{prev})$ . Найти множество  $P_0$  характерных точек  $p_0$  в области предыдущего кадра, ограниченной прямоугольником  $B_{prev}$ , для вычисления оптического потока Лукаса–Канаде.

Шаг 19. Если размерность массива  $P_0$  меньше 4, {удалить массивы  $H$  и  $S$ , увеличить значение счетчика кадров  $t := t + 1$ , перейти на шаг 2}.

Шаг 20. Найти оптический поток в точках  $p_0$  между предыдущим и текущим кадрами. Удалить из  $P_0$  и  $P_1$  те пары точек, сдвиг оптического потока в которых превышает  $\Delta_{max}$ .

Шаг 21. Если число оставшихся пар соответствующих точек  $p_0$  и  $p_1$  меньше 4, {удалить массивы  $H$  и  $S$ , увеличить значение счетчика кадров, положив  $t := t + 1$ , перейти на шаг 2}.

Шаг 22. Вычислить проективное преобразование на основе найденных оптическим потоком пар соответствующих точек  $p_0$  и  $p_1$ . Найти вектор сдвига  $\Delta$  центра прямоугольника  $B_{prev}$  с помощью вычисленного проективного преобразования.

Шаг 23. Если величина вектора сдвига  $\Delta$  больше  $\Delta_{max}$  и среднего значения элементов массива  $S$  (если он не пустой), {удалить массивы  $H$  и  $S$ , увеличить значение счетчика кадров, положив  $t := t + 1$ , перейти на шаг 2}.

Шаг 24. Сдвинуть  $B_{prev}$  на полученный сдвиг  $\Delta$  и добавить координаты сдвинутого прямоугольника в конец массива  $H$ . Добавить величину сдвига  $\Delta$  в конец массива  $S$ .

Шаг 25. Если размер  $H$  превышает допустимое значение  $d$ , {удалить первые элементы массивов истории ограничивающих прямоугольников  $H$  и истории сдвигов  $S$ }. Перейти на шаг 2.

Ниже на рис. 1 приведена подробная блок-схема алгоритма.

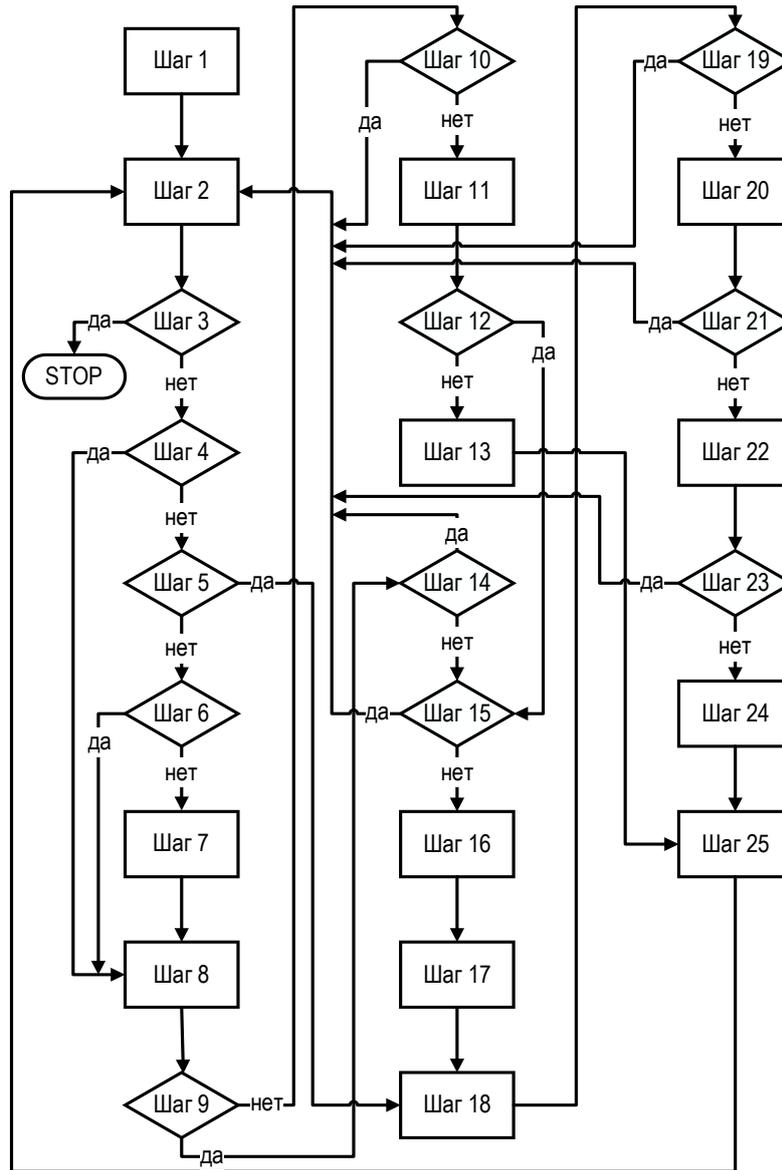


Рис. 1. Блок-схема алгоритма

Fig. 1. Algorithm flowchart

**Результаты и их обсуждение.** Для исследования характеристик алгоритма (тестирования) использовались три набора данных, созданных и размеченных авторами статьи (так как в открытом доступе отсутствуют наборы данных объектов данного типа). Два из них содержат 1264 и 5861 кадр с коптерами, находящимися в небе, снятых видеокамерой с земной поверхности. Кадры видео представляют собой RGB-изображения размером  $1920 \times 1080$  пикселей.

На кадрах первого, меньшего, набора данных присутствуют два квадрокоптера. Оба они обнаруживаются детектором. Один из них автоматически захватывается построенным алгоритмом и надежно сопровождается вне зависимости от того, где находится второй квадрокоптер. На кадрах второго, большего, набора данных присутствует один квадрокоптер. Размер квадрокоптера на всех кадрах превосходит 117 пикселей (рис. 2).

Ниже в таблице приведены результаты тестирования. Напомним, что параметр  $q$  задает периодичность использования детектора, параметр  $d$  – длину массива истории сопровождения на предыдущих кадрах, а  $b$  – количество кадров, на которых объект сопровождается оптическим потоком после его пропуска детектором.



Рис. 2. Пример работы алгоритма

Fig. 2. Example of the algorithm

### Результаты на первом и втором тестовых наборах видео

#### Results on the first and second test sets of videos

Видео Video	Кол-во кадров Number of frames	$IoU$ (Average)	Precision (%)	Recall (%)
Два квадрокоптера	1264	0,7415	100	100
Один квадрокоптер	5861	0,5728	99,67	99,67
Объединенное	7125	0,6027	99,76	99,76

Характеристика  $IoU$  (Intersection over Union) определяется двумя прямоугольниками, ограничивающими сопровождаемый объект, один из которых,  $R_1$ , найден вручную экспертом (так называемый ground truth прямоугольник), а второй,  $R_2$ , вычислен алгоритмом. Она задается равенством  $IoU = |R_1 \cap R_2| / |R_1 \cup R_2|$ . Среднее  $IoU$  вычисляется по всем кадрам. Значение параметра  $\Delta$  равно средней длине вектора разности центров прямоугольников  $R_1$  и  $R_2$ .

Результаты, приведенные в таблице, получены при  $q = 3$ ,  $d = 5$ ,  $b = 5$ .

Алгоритм также был протестирован на третьем наборе данных, составленном из 21-го короткого видео с различными коптерами (не только квадрокоптерами), состоящего из 7489 кадров. На этом наборе данных были получены следующие средние значения характеристик:  $IoU = 0,7471$ , Precision = 100 % (т. е. коптеры были обнаружены и локализованы корректно на всех кадрах), Recall = 100 %. Относительная погрешность  $\delta$  оценки центра  $c_1$  ground truth ограничивающего прямоугольника  $R_1$  определялась по формуле  $\delta = \|c_1 - c_2\| / \text{diam}(R_1)$ , в которой  $c_2$  – центр найденного алгоритмом ограничивающего объекта прямоугольника  $R_2$ . Относительная погрешность определения центра  $\delta = 0,09465$ .

**Заключение.** В работе представлен алгоритм сопровождения объекта, наблюдаемого на кадрах видео. Построенный алгоритм позволяет автоматически обнаружить и захватить объект одного из заранее заданных типов, наблюдаемых на видео. Для обнаружения и захвата объекта может использоваться подходящий детектор объектов. В представленном алгоритме для этого использовался нейросетевой детектор объектов YOLOv8, хотя допустимо использование и других детекторов. Дальнейшее сопровождение объекта выполняется предложенным алгоритмом сопровождения.

Тестирование построенного алгоритма показало возможность его применения для уверенного автоматического обнаружения объектов (размера, превосходящего 117 пикселей на изображениях размером  $1920 \times 1080$  пикселей) одного из заранее заданных типов и надежного сопровождения одного из обнаруженных объектов с повторным захватом в случае его потери, перекрытия или временного отсутствия на кадрах видео.

В дальнейшем предполагается усовершенствовать алгоритм для реализации сопровождения объектов меньших размеров.

**Благодарности.** Авторы благодарят Д. В. Морозова и А. Ю. Сивенкова за любезно предоставленные видео с квадрокоптером, которые были использованы для обучения и тестирования алгоритма.

**Acknowledgments.** The authors thank D. V. Morozov and A. Yu. Sivenkov for kindly providing videos with a quadcopter, which were used for training and testing the algorithm.

### Список использованных источников

1. Object Tracking [Electronic resource]. – Mode of access: <https://paperswithcode.com/task/object-tracking>. – Date of access: 05.01.2024.
2. Proceedings of Conference on Computer Vision and Pattern Recognition 2023. CVPR 2023. Vancouver Jun 18 2023 [Electronic resource]. – Mode of access: <https://openaccess.thecvf.com/CVPR2023>. – Date of access: 05.01.2024.
3. Visual object tracking: A survey / F. Chen [et al.] // *Computer Vision and Image Understanding*. – 2022. – Vol. 222. – Art. 103508. <https://doi.org/10.1016/j.cviu.2022.103508>
4. Wojke N. Simple online and realtime tracking with a deep association metric / N. Wojke, A. Bewley, D. Paulus // 2017 IEEE International Conference on Image Processing (ICIP), 17–20 September 2017. – Beijing, China, 2017. – P. 3645–3649. <https://doi.org/10.1109/ICIP.2017.8296962>
5. ByteTrack: Multi-Object Tracking by Associating Every Detection Box / Yi Zhang [et al.] // *Computer Vision – ECCV 2022. ECCV 2022* / eds.: S. Avidan [et al.]. – Springer, Cham, 2022. – P. 1–21. – (Lecture Notes in Computer Science, vol. 13682). [https://doi.org/10.1007/978-3-031-20047-2\\_1](https://doi.org/10.1007/978-3-031-20047-2_1)

### References

1. *Object Tracking* (2024). Available at: <https://paperswithcode.com/task/object-tracking> (accessed 05 January 2024).
2. *Proceedings of Conference on Computer Vision and Pattern Recognition 2023. CVPR 2023. Vancouver Jun 18 2023*. Available at: <https://openaccess.thecvf.com/CVPR2023> (accessed 05 January 2024).
3. Chen F., Wang X., Zhao Y., Lv Sh., Niu X. Visual object tracking: A survey. *Computer Vision and Image Understanding*, 2022, vol. 222, art. 103508. <https://doi.org/10.1016/j.cviu.2022.103508>
4. Wojke N., Bewley A., Paulus D. Simple online and realtime tracking with a deep association metric. *2017 IEEE International Conference on Image Processing (ICIP), 17–20 September 2017*, Beijing, China, 2017, pp. 3645–3649. <https://doi.org/10.1109/ICIP.2017.8296962>
5. Zhang Y., Sun P., Jiang Y., Yu D., Weng F., Yuan Z., Luo P., Liu W., Wang X. ByteTrack: Multi-Object Tracking by Associating Every Detection Box. Avidan S., Brostow G., Cissé M., Farinella G. M., Hassner T. (eds.). *Computer Vision – ECCV 2022. ECCV 2022. Lecture Notes in Computer Science, vol. 13682*. Springer, Cham, 2022, pp. 1–21. [https://doi.org/10.1007/978-3-031-20047-2\\_1](https://doi.org/10.1007/978-3-031-20047-2_1)

### Информация об авторах

*Залесский Борис Андреевич* – д-р физ.-мат. наук, заведующий лабораторией. Объединенный институт проблем информатики НАН Беларуси (ул. Сурганова, 6, 220012, Минск, Республика Беларусь). E-mail: [zalesky@newman.bas-net.by](mailto:zalesky@newman.bas-net.by).

*Иванюкович Владимир Александрович* – мл. науч. сотрудник. Объединенный институт проблем информатики НАН Беларуси (ул. Сурганова, 6, 220012, Минск, Республика Беларусь).

### Information about the authors

*Zalesky Boris A.* – D. Sc. (Physics and Mathematics), Head of the Laboratory. United Institute of Informatics Problems of the National Academy of Sciences of Belarus (6, Sarganov Str., 220012, Minsk, Republic of Belarus). E-mail: [zalesky@newman.bas-net.by](mailto:zalesky@newman.bas-net.by).

*Ivanyukovich Vladimir A.* – Junior Researcher. United Institute of Informatics Problems of the National Academy of Sciences of Belarus (6, Sarganov Str., 220012, Minsk, Republic of Belarus).