diversity

MDPI

Article

BugTracker: Software for Tracking and Measuring Arthropod Activity

Hajnalka Malakoff [1,2,3], Dávid Tőzsér [2,4], Béla Tóthmérész [2,3] and Tibor Magura [2,5,*]

[1] Juhász-Nagy Pall Doctoral School of Biology and Environmental Sciences, University of Debrecen, Egyetem Sq. 1, H-4032 Debrecen, Hungary; roffa.hajnalka.91@gmail.com
[2] Department of Ecology, Faculty of Science and Technology, University of Debrecen, Egyetem Sq. 1, H-4032 Debrecen, Hungary; tozser.david@windowslive.com (D.T.); tothmerb@gmail.com (B.T.)
[3] MTA-DE Biodiversity and Ecosystem Services Research Group, Egyetem Sq. 1, H-4032 Debrecen, Hungary
[4] Circular Economy Analysis Center, Hungarian University of Agriculture and Life Sciences, Páter Károly Str. 1, H-2100 Gödöllő, Hungary
[5] ELKH-DE Anthropocene Ecology Research Group, Egyetem Sq. 1, H-4032 Debrecen, Hungary
* Correspondence: magura.tibor@science.unideb.hu

Abstract: The automated video tracking of the activity/movement of an experimental organism is essential for reliable, repeatable quantitative analyses in behavioral ecology and also in other disciplines. There are only some open-access, open-source automated tracking software applications that can track unmarked organisms. Moreover, several of these software applications are substantially affected by brightness and differences in the lighting conditions of the video recording. Our Python-based software, called *BugTracker*, uses the latest innovations in computer vision technologies to solve these problems. By analyzing videos with considerably different lighting conditions with *BugTracker* and other available software, we demonstrate that our software could reliably track the studied organisms of any size and speed. Additionally, the results provide accurate measures of the organism's movements. *BugTracker* is the most reliable currently available, easy-to-use, and automated tracking software compatible with the Windows, Linux, and MacOS operating systems.

Keywords: behavior; activity; locomotion; open field test; video; tracking; Python programming language

Citation: Malakoff, H.; Tőzsér, D.; Tóthmérész, B.; Magura, T. *BugTracker*: Software for Tracking and Measuring Arthropod Activity. *Diversity* **2023**, *15*, 846. https://doi.org/10.3390/d15070846

Academic Editors: Paolo Solari, Giorgia Sollai, Roberto Massimo Crnjar, Anita Giglio, Piero G. Giulianini and Michael Wink

Received: 9 June 2023
Revised: 7 July 2023
Accepted: 8 July 2023
Published: 10 July 2023

1. Introduction

Global anthropogenic activities, such as agriculture, forestry, and urbanization, are leading to significant changes in environments and creating patchworks of modified land types [1]. Alterations in environmental parameters trigger pressures on organisms living there, modifying their life history (phenology, body condition, and productivity) and behavioral parameters (activity, exploratory, and boldness) [2,3]. Changes in behavior are possibly the fastest way to react to unfavorable changes in environmental conditions; thus, behavioral studies are nowadays widespread [4]. In many disciplines, including behavioral ecology, ethology, and evolutionary ecology, a key question concerns how to quantify the activity/movement of the studied organisms. However, movement is also a crucial parameter, and a proxy is frequently used to evaluate the effect of human-associated stressors, chemicals, and pharmaceuticals [5]. Therefore, precise and accurate methods for measuring movements using noninvasive techniques are highly requested [6]. One common way to study the movement of arthropods is with artificial laboratory test arenas using video recordings over short periods (minutes) [7].

The availability of video-making products and powerful personal computers that are capable of complex, frame-by-frame analyses is making it increasingly feasible to record and measure animal movements [8]. Thus, even with a small investment or by using a cell phone, we can make a video recording of their movements. The fast transfer of data enables

video recordings to be sent directly to a computer. Thus, we can even process these data in real time, or we can automate the processes and perform multiple analyses simultaneously with high precision and accuracy.

Automated monitoring systems provide a solution to either detect behaviors that occur only for a short period and then blend into each other or involve long periods of inactivity or possibly require continuous attention [9]. Reducing human resources can minimalize calculation and processing errors and promote task parallelization. Today, several software applications provide solutions for modeling and processing ecological problems and tasks. Although most of these pieces of software are parts of a prewritten standard process, there are always experiment-specific steps that pose a challenge to existing ones, possibly requiring the modification of the existing code in ways that are not supported by the code that has already been written and would use incompatible algorithms. Most of them are well suited to particular arenas and animals but are not capable of tracking arthropods of any species, shape, or size, regardless of the arena shape. Also, some software is proprietary, which means its creator prohibits users from freely sharing or modifying it, and its usage usually costs money. Based on these aforementioned issues, the prospect is given to open-source programs created with modern coding tools that enable the easy and quick development of solutions for a given task.

There are software applications that are capable of tracking one or multiple animals [10–13]. However, they use different methods for tracking than what we provide in this study. They detect an object based on contrast, basically segmenting focal objects from the background. Because of this, the examined area should be substantially brighter than the trackable animal. So, these programs are substantially affected by brightness and differences in lighting conditions in the area. Here, we propose an open-source tracking software, *BugTracker*, which uses a more robust and complex algorithm for tracking than the threshold- or contrast-based alternatives. Below, we summarize the functions of the software, offer examples of its applications, compare its characteristics and performance to other tracking software, and discuss its strengths and limitations.

2. Materials and Methods

2.1. Software

BugTracker was created to use a newer algorithm for tracking than existing solutions and to be optimized for testing a specified behavioral parameter, which can be used to facilitate the measurement and evaluation process and increase repeatability. It uses the CSRT (discriminative correlation filter with channel and spatial reliability) algorithm; thus, it is less affected by brightness and differences in the lighting conditions in the area.

BugTracker is a Python-based program that uses the open-source OpenCV software library (https://opencv.org, accessed on 11 April 2020) for computer vision algorithms to identify shapes and track objects. Python is a simple and easy-to-understand programming language for nontechnical users, and it is very popular in science [14,15]. Because of the increasing popularity and usage of the R programming language among ecologists, we also intend to create an R version of the software.

The goal of the OpenCV software library is to provide tools for solving computer vision problems. This includes a large variety of algorithms both for low-level image processing, like object recognition and tracking functions, and high-level algorithms, such as face recognition, pedestrian detection, feature matching, and tracking [16].

The software is designed to run multiple instances at once, but there is no GUI (graphical user interface) available yet for the configuration. It is easy to install and configure, even for nontechnical users. It does not require any programming knowledge to use it. *BugTracker* is available under GNU General Public License v3.0, with an installation and usage guide accessible on GitHub: https://github.com/Roffagalaxis/Bugtracker (accessed on 29 March 2023). We chose GitHub because of its popularity within and outside of academia, which increases the probability of it being discovered by people with programming knowledge. Making your research publicly accessible in this way leads to

higher citation counts and a larger impact [17]. If the complete research project is hosted on a free-to-use site like GitHub, it becomes more easily reproducible and transparent, as how the code and data changed during the journey from idea to publication can be reviewed. With a public repository, anyone can retrieve the previous status of the research at any time [18].

2.2. Defining the Testing Area

The first step in the usage of the *BugTracker* software is to identify the analyzed area. We can define a square area manually, or we can use automatic detection, which works with any shape if the area has a visible barrier.

Manual setting uses the following codes:

```
initarea = cv2.selectROI(framename, image, fromCenter = False, showCrosshair = True)
for i in range (0, area [2], step_ver):
cv2.line(image, (area [0] + i, area [1]), (area [0] + i, area [3] + area [1]), (0, 255, 0), 1)
for i in range (0, area [3], step_hor):
cv2.line(image, (area [0], area [1] + i), (area [2] + area [0], area [1] + i), (0, 255, 0), 1)
```

Algorithm for manual setting:

1. Stop the video and evoke the mouse control for the user to select the area.
2. Draw the vertical lines of the squares.
3. Draw the horizontal lines of the squares.
4. The user can later adjust these lines with the W, A, S, and D keys on the keyboard at any time while the video is running.

The automatic mode searches for object contours based on border following algorithms [19]. A border following algorithm is a core technique for processing digital images. It derives a series of coordinates from the boundary between an I-pixel and a connected component of an O-pixel (i.e., background). These techniques have been thoroughly studied because of their many uses in computer vision [20–27].

The automatic area detector method uses the following codes for detecting the test arena and squares:

```
gray = cv2.cvtColor (image, cv2.COLOR_BGR2GRAY)
blur_gray = cv2.GaussianBlur (gray, (5, 5), 0)
edges = cv2.Canny (blur_gray, 50, 150)
contours, hierarchy = cv2.findContours (edges, cv2.RETR_EXTERNAL,
cv2.CHAIN_A-PROX_NONE)
```

Algorithm for automatic area detection:

1. Convert image to grayscale format.
2. Apply binary thresholding.
3. Find contours based on the border following algorithms.

Using various videos, we examined the effect of light conditions on the automatic area detection efficiency for open field tests. We concluded that the ideal conditions for recording video are cold light from a vertical 90° angle or using a lightbox in order to create as little shadow as possible around the studied animal (Figure 1).

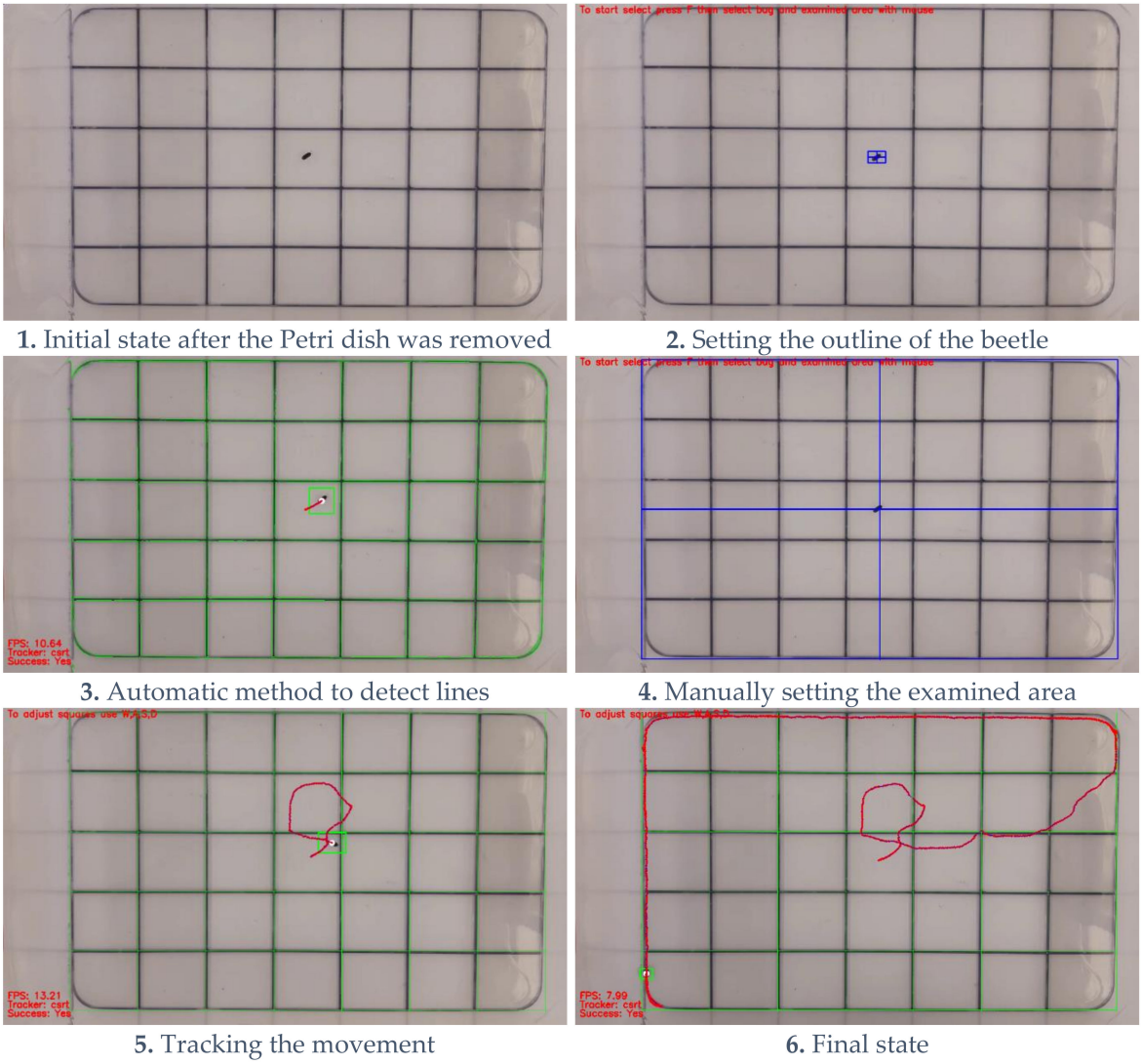

Figure 1. Steps of the process of arthropod tracking. For details of this particular open field test, see Section 2.4 (Examples of the Software's Applications).

2.3. Tracking the Defined Test Organism

After defining the test area, we need to focus on short-term and model-free object tracking, which is an open and actively studied problem [28–30]. Localizing a target in a continuously moving video requires significant computer resources. The results also depend on various factors like distinguishability from the background, lighting conditions, presence of shadows, fast object movement, disappearances, or deformations [31–34].

Currently available software solutions mostly use two types of detection techniques: static and dynamic background subtraction. The static one uses a reference frame or video, where only the background is visible without any animals. The dynamic technique works with moving or decaying average models, where detection counts on illumination or other gradual changes to the frame [35].

OpenCV includes the following built-in tracking algorithms:

- cv2.TrackerCSRT_create;
- cv2.TrackerKCF_create;
- cv2.TrackerMIL_create;
- cv2.legacy.TrackerBoosting_create;
- cv2.legacy.TrackerTLD_create;
- cv2.legacy.TrackerMedianFlow_create;

- cv2.legacy.TrackerMOSSE_create.

Based on a preliminary test, the built-in CSRT (discriminative correlation filter with channel and spatial reliability) tracker works with the best accuracy. The CSRT algorithm uses a spatial reliability map to find the examined object that is most suitable for tracking. The spatial reliability map is estimated using a graph labeling algorithm. This grants a significantly wider search region, the possibility to use complex backgrounds, and improved nonrectangular object tracking [36,37].

It was also chosen because it handles special cases like object crosses or disappearances, and there is no limit to the size of the examined individual, so it is also suitable for tracking the movements of both small and large animals.

The main functions of any of the above trackers are:

initTracker = cv2.selectROI (showedframe, image, fromCenter = False, showCrosshair = True)
tracker.init (sourceframe, initTracker)
(success, box) = tracker.update (sourceframe)

Algorithm to call tracking:

1. Stop the video and evoke the mouse control for the user to select the outline of the object they intend to track.
2. Start the tracking based on these boundaries.
3. Each frame looks for the object, and it reports back whether it finds it and the outline where it was found.

The user sets the outline of an animal to track, and the program will continue the tracking until the input video is finished.

2.4. Examples of the Software's Applications

BugTracker was optimized on videos created by Magura and his colleagues. They collected rove beetles along a rural–urban gradient, in and near Debrecen city (Eastern Hungary). Eight forest stands were selected along the gradient: four rural sites, and four in urban areas. The rural sites were located in a large forest of 1082 hectares in old (>130 years) forest stands, where English oak (*Quercus robur*) was the dominant species. All urban sites were isolated fragments of a once-continuous old forest (>130 years) dominated by English oak (*Quercus robur*). The rove beetles were sampled during their main activity periods using live pitfall traps. The traps were checked twice a week, from early April to late June. The collected beetles were transported to the laboratory, where standard conditions (24 °C, 40% relative humidity, and natural L:D cycle) were provided. In the laboratory, the beetles were left to rest with access to water but no food [38]. After the resting period, the activity of the beetles was individually tested in a novel environment, also called an open field test [39,40]. In the test arena, each beetle was covered with a Petri dish, and when it stopped moving, the Petri dish was lifted, and the movement of the beetle was recorded for the next 90 s with a GoPro HERO6 camera (CHDHX-601-FW) [38].

The current version of the *BugTracker* software was designed to automate the processing of this test in the best possible way, as this test is often used to evaluate exploratory behavior [41–43]. In the basic videos, the software worked with a preset test environment, which consisted of an open box (364 × 230 mm) whose bottom was divided into 35 equal-sized squares [38]. The software was created in such a way that the grid lines could be added to the grids already placed at the bottom of the box, but this part can be omitted using the software, as it can create its grid structure during the session.

2.5. Software Output

The current version of the software provides basic information for the user, including a graph of the trajectory of the bug's movement and the number of visited cells of the grid. After the video evaluation, we can print both the graph of the trajectory and the number of visited cells (see Figure 1). The number of visited cells is a vital parameter during an ecological study [38].

2.6. Comparing the Properties of BugTracker with Other Similar Software

We compared the properties of our software to the other, currently available alternatives to highlight the differences. The main properties of the programs are summarized in Table 1.

Table 1. Main properties of the programs used to compare their performances.

Software	Environment	Interface	Open Source	Method	Visualization Process	Object Crosses Handling
BugTracker	Python, OpenCV	Terminal-based/Command-based	Yes	CSRT	Yes	Yes, by tracker
Tracktor [10]	Python, OpenCV	Command-based	Yes	Dynamic background subtraction	Yes	No
ToxTrac [11]	C++, OpenCV	GUI	No	Dynamic background subtraction	No	Yes, by matching similarities of the trajectory
Pathtrackr [13]	R	Command-based	Yes	Static background subtraction	Yes	No

Environment—programming language and library used; **Interface**—type of interactive user interface; **Methods**—computer vision technology used in the software; **Visualization process**—whether the user can follow the tracking process with the software; **Object crosses handling**—whether the software can solve crosses of movement paths or disappearances.

2.7. Video Quality

For the comparison, we used video recordings with the following lighting and camera stability parameters: (1) video recording with ideal lighting conditions (next to the ceiling lamps, an additional external lighting source directed towards the test arena) and a fixed camera (using a tripod); (2) video recording with moderate lighting conditions (using only ceiling lamps without an additional external lighting source) and a fixed camera (using a tripod); and (3) video recording with moderate lighting conditions (using only ceiling lamps without an additional external lighting source) and a nonfixed, hand-held camera.

The outline of the insect was set after the Petri dish was lifted during the set-up of *ToxTrax* and *Bugtracker* for the best possible results. There is no similar setting in *Tracktor* and *Pathtrackr*.

3. Results

We compared the performance of our software to the currently available alternatives using the same videos with the various lighting and camera stability conditions. Based on our testing, none of the example video recordings could be calibrated well using *Tracktor*, so the software did not produce accurate tracking results for the videos. After proper calibration, *ToxTrac* worked precisely on the video recordings with the ideal and moderate lighting conditions and a fixed camera, but it failed on the hand-held video recording. *Pathtracker* was able to track the animal with the ideal lighting conditions with a fixed camera, but it could not continuously track the animal's movements under moderate lighting conditions with a fixed camera. In addition, the video recording with a hand-held camera under moderate lighting conditions could not be processed by this program at all. Our software, *BugTracker*, was able to precisely track the target animals during the whole video recordings with various lighting and camera stability conditions (Figure 2). The number of visited squares, a recognized measure of activity and exploration in arthropods [38], counted by the four animal-tracking software applications were also

markedly different for the video recordings under suboptimal parameters (i.e., moderate lighting conditions and/or a nonfixed, hand-held camera; Table 2).

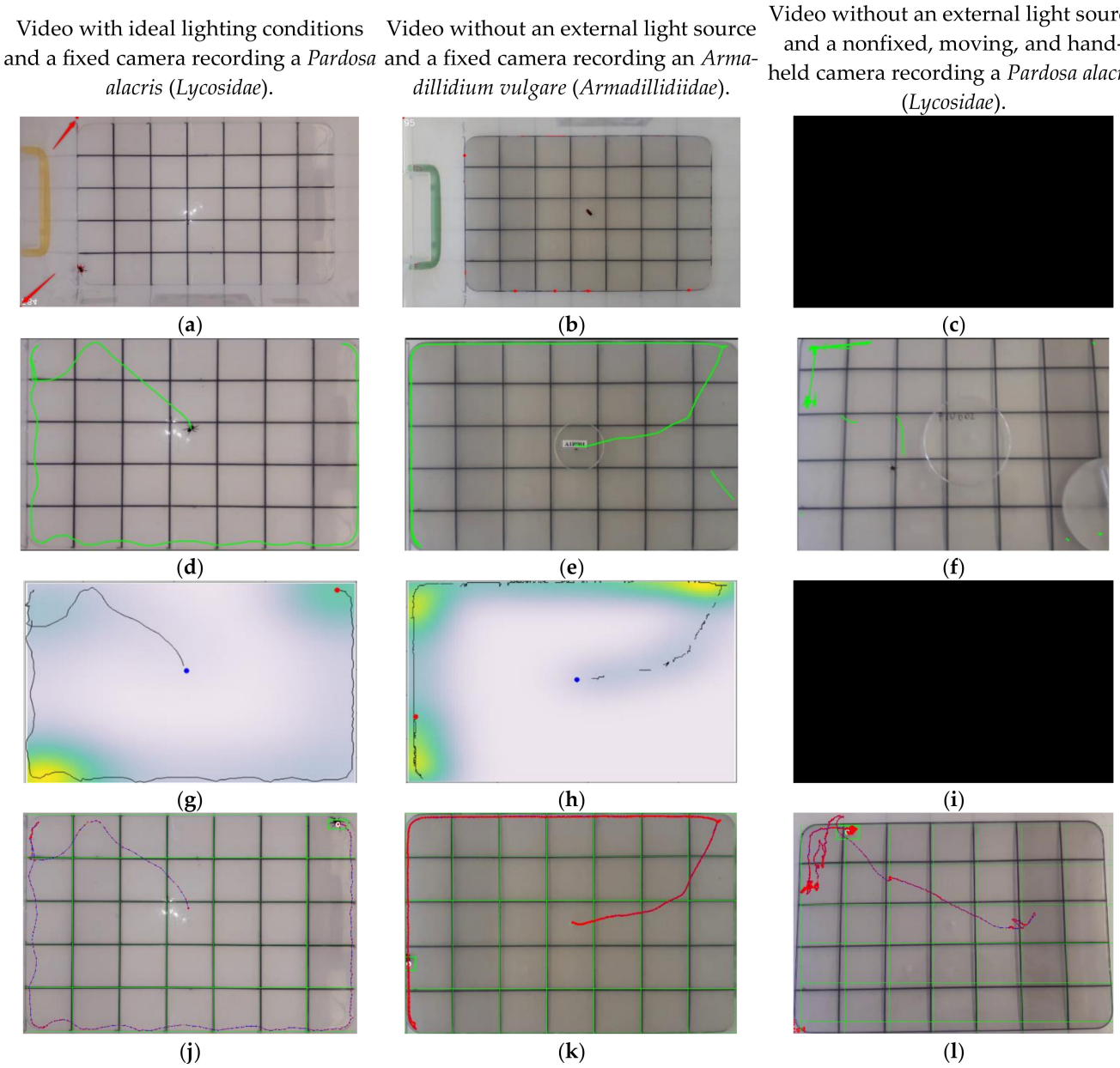

Figure 2. Performance of four tracking programs using the same video recordings with different lighting and camera stability conditions. Recorded movement paths: *Tracktor* (**a**–**c**); *ToxTrac* (**d**–**f**); *Pathtrackr* (**g**–**i**); *BugTracker* (**j**–**l**).

Table 2. The number of visited squares counted manually and with the four animal-tracking software applications using video recordings with different lighting and camera stability conditions. The values highlighted in red indicate that the application was unable to track the arthropod's movement. The values marked in orange indicate incorrect results, while the values highlighted in green indicate correct results for the visited square numbers.

Software	Video with Ideal Lighting Conditions and a Fixed Camera Recording the Movements of *Pardosa alacris* (*Lycosidae*)	Video without an External Light Source and a Fixed Camera Recording the Movements of *Armadillidium vulgare* (*Armadillidiidae*)	Video without an External Light Source and a Nonfixed, Moving, Hand-Held Camera Recording the Movements of *Pardosa alacris* (*Lycosidae*)
Manual counting	21	16	12
Tracktor [10]	0	0	0
ToxTrac [11]	21	17	7
Pathtrackr [13]	21	14	0
BugTracker	21	16	12

4. Discussion

Future Software Developments

An open-source software allows users and developers to use the knowledge of the software for their personalized experiments. The most important duty is to develop a proper GUI (graphical user interface) and an R version of the software to help the adaptation of the software among entomologists and other researchers.

The length of the trajectory, average acceleration, and speed are also useful information for researchers analyzing animal movements. We would like to include the measurement of these parameters in the software. Tracking multiple individuals at the same time in the same arena is also planned.

An improved version of our software can even be suitable for processing UAS camera recordings and for tracking location changes and behavioral patterns recorded on aerial photographs. Nowadays, the use of UAS in scientific research is increasing rapidly, as it can provide researchers with a completely new perspective.

Since the CSRT tracker does not need a constant background, it is not only suitable for tracking the movement of living beings but could be used for tracking vehicles. An advanced version of our software could also be suitable for following a live recording. Implementing this feature also means that we could track the movement of a species like *Silpha tristis (Silphidae)* for weeks in a closed terrarium. Even if the beetle hid somewhere, the tracker could continue tracking it when it came out of hiding. From these data, we could determine when and how much distance it moves during a day or week. We could examine whether it moves less when it becomes older.

5. Conclusions

With the use of the *BugTracker* software, human resources can be optimized during open field experiments. It can reduce calculation and processing errors caused by human factors and promote task parallelization. During studies, arthropods are not limited by species, shape, or size either; the movements of *Acinophus ammophilus (Carabidae)*, which is a large beetle (20–28 mm), or even small ones, such as *Armadillidium vulgare (Armadillidiidae)* with a 0.7–18 mm length, can be monitored effectively. We armed the software with the recent tracking algorithms, which provide better results under unfavorable conditions than the existing solutions and are optimized for testing a specified behavioral parameter, which can facilitate the evaluation. Since the code for this software is open and available on GitHub, it is easily accessible and can be used by anyone in the Python environment. Its development is quite broad because of the use of the Python programming language,

as it is simple and understandable, even for novice programmers, and later it can also be properly integrated into R, thus making it even more widely available to users.

Author Contributions: Conceptualization, H.M., B.T. and T.M.; methodology, H.M., B.T. and T.M.; software, H.M.; validation, H.M.; formal analysis, H.M. and T.M.; investigation, H.M. and T.M.; resources, H.M. and T.M.; data curation, H.M. and T.M.; writing—original draft preparation, H.M.; writing—review and editing, B.T., D.T. and T.M.; visualization, H.M.; supervision, B.T. and T.M.; project administration, H.M. and T.M.; funding acquisition, H.M., D.T., B.T. and T.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partly supported by the National Research, Development, and Innovation Fund, grant numbers: OTKA K-131459 (Magura, Tibor) and OTKA-PD-138806 (Tőzsér, Dávid).

Institutional Review Board Statement: Ethical review and approval were waived for this study because its protocol did not involve invasive measurements in animals.

Informed Consent Statement: Not applicable.

Data Availability Statement: Codes, usage guide, and example videos are available on GitHub: https://github.com/Roffagalaxis/Bugtracker (accessed on 29 March 2023).

Acknowledgments: We thank Maria Toth and Panna Boros for supporting the development of *BugTracker*.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kerr, J.T.; Currie, D.J. Effects of human activity on global extinction risk. *Conserv. Biol.* **1995**, *9*, 1528–1538. [CrossRef]
2. Wong, B.B.M.; Candolin, U. Behavioral responses to changing environments. *Behav. Ecol.* **2015**, *26*, 665–673. [CrossRef]
3. Cote, J.; Clobert, J.; Fogarty, S.; Sih, A. Personality-dependent dispersal: Characterization, ontogeny and consequences for spatially structured populations. *Philos. Trans. R. Soc. B* **2010**, *365*, 4065–4076. [CrossRef] [PubMed]
4. Sih, A. Understanding variation in behavioural responses to human-induced rapid environmental change: A conceptual overview. *Anim. Behav.* **2013**, *85*, 1077–1088. [CrossRef]
5. Seehausen, O.; Alphen, J.J.M.; Witte, F. Cichlid fish diversity threatened by eutrophication that curbs sexual selection. *Science* **1997**, *277*, 1808–1811. [CrossRef]
6. Dell, A.I.; Bender, J.A.; Branson, K.; Couzin, I.D.; de Polavieja, G.G.; Noldus, L.P.J.J.; Pérez-Escudero, A.; Perona, P.; Straw, A.D.; Wikelski, M.; et al. Automated image-based tracking and its application in ecology. *Trends Ecol. Evol.* **2014**, *29*, 417–428. [CrossRef] [PubMed]
7. Zaller, J.G.; Kerschbaumer, G.; Rizzoli, R.; Tiefenbacher, A.; Gruber, E.; Schedl, H. Monitoring arthropods in protected grasslands: Comparing pitfall trapping, quadrat sampling and video monitoring. *Web Ecol.* **2015**, *15*, 15–23. [CrossRef]
8. Richard, F.O.; Margaret, C.T. Monitoring animals' movements using digitized video images. *Behav. Res. Meth. Instr.* **1988**, *20*, 485–490. [CrossRef]
9. Martin, B.R.; Prescott, W.R.; Zhu, M. Quantitation of rodent catalepsy by a computer-imaging technique. *Pharmacol. Biochem. Behav.* **1992**, *43*, 381–386. [CrossRef]
10. Sridhar, V.H.; Roche, D.G.; Gingins, S. Tracktor: Image-based automated tracking of animal movement and behaviour. *Methods Ecol. Evol.* **2019**, *10*, 810–820. [CrossRef]
11. Rodriguez, A.; Zhang, H.; Klaminder, J.; Brodin, T.; Andersson, P.L.; Andersson, M. ToxTrac: A fast and robust software for tracking organisms. *Methods Ecol. Evol.* **2017**, *9*, 460–464. [CrossRef]
12. Yamanaka, O.; Takeuchi, R. UMATracker: An intuitive image-based tracking platform. *J. Exp. Biol.* **2018**, *221*, jeb182469. [CrossRef] [PubMed]
13. Harmer, A.M.T.; Thomas, D.B. pathtrackr: An r package for video tracking and analysing animal movement. *Methods Ecol. Evol.* **2019**, *10*, 1196–1202. [CrossRef]
14. Vineesh, C.; Nehemiah, S. A review on using Python as a preferred programming language for beginners. *Int. Res. J. Eng. Technol.* **2021**, *8*, 4258–4263. Available online: https://www.irjet.net/archives/V8/i8/IRJET-V8I8505.pdf (accessed on 29 March 2023).
15. Málik-Roffa, H. Ízeltlábúak Viselkedés Kutatását Támogató Szoftver Fejlesztése Python Nyelven (Development of Software Supporting the Research of Arthropod Behavior in Python). Master's Thesis, GAMF Faculty of Engineering and Computer Science John von Neumann University,, Kecskemét, Hungary, 2021.
16. Pulli, K.; Baksheev, A.; Kornyakov, K.; Eruhimov, V. Real-time computer vision with OpenCV. *Commun. ACM* **2012**, *55*, 61–69. [CrossRef]
17. Piwowar, H.A.; Day, R.S.; Fridsma, D.B. Sharing detailed research data is associated with increased citation rate. *PLoS ONE* **2007**, *2*, e308. [CrossRef]

18. Jones, Z.M. Git/GitHub, transparency, and legitimacy in quantitative research. *Political Methodol.* **2013**, *21*, 6–7.
19. Suzuki, S.; Be, K. Topological structural analysis of digitized binary images by border following. *Comput. Vision Graph.* **1985**, *30*, 32–46. [CrossRef]
20. Rosenfeld, A.; Kak, A.C. *Digital Picture Processing*, 2nd ed.; Academic Press: New York, NY, USA, 1982; Volume 2.
21. Rosenfeld, A. Connectivity in digital pictures. *J. Assoc. Cornput. Mach.* **1970**, *17*, 146–160. [CrossRef]
22. Yokoi, S.; Toriwaki, J.; Fukumura, T. An analysis of topological properties of digitized binary pictures using local features. *Comput. Graph. Image Process* **1975**, *4*, 63–73. [CrossRef]
23. Li, Z.; Yokoi, S.; Toriwaki, J.; Fukumura, T. Border following and reconstruction of binary pictures using grid point representation. *Trans. Inst. Electron. Commun. Eng. Jpn.* **1982**, *J65*, 1203–1210.
24. Pavlidis, T. *Algorithms for Graphics and Image Processing*, 1st ed.; Computer Science Press, Inc.: Rockville, MY, USA, 1982.
25. Morrin II, T.H. Chain-link compression of arbitrary black-white image. *Comput. Graph. Image Process* **1976**, *5*, 172–189. [CrossRef]
26. Agui, T.; Iwata, I. Topological structure analysis of pictures by digital computer. *Syst. Comput. Controls* **1979**, *10*, 10–19.
27. Pavlidis, T. *Structural Pattern Recognition*; Springer: Berlin/Heidelberg, Germany, 1977.
28. Wu, Y.; Lim, J.; Yang, M.H. Online object tracking: A benchmark. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 2411–2418. [CrossRef]
29. Kristan, M.; Pflugfelder, R.; Leonardis, A.; Matas, J.; Porikli, F.; Cehovin, L.; Nebehay, G.; Fernandez, G.; Vojir, T.; Gatt, A.; et al. The Visual Object Tracking VOT2013 Challenge Results. In Proceedings of the 2013 IEEE International Conference on Computer Vision Workshops, Washington, DC, USA, 2–8 December 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 98–111. [CrossRef]
30. Kristan, M.; Pflugfelder, R.; Leonardis, A.; Matas, J.; Cehovin, L.; Nebehay, G.; Vojir, T.; Fernandez, G.; Lukežič, A.; Dimitriev, A.; et al. The Visual Object Tracking VOT2014 Challenge Results. In Proceedings of the Computer Vision—ECCV 2014 Workshops, Zurich, Switzerland, 6–7 September 2014; ECCV 2014, Lecture Notes in Computer Science. Agapito, L., Bronstein, M., Rother, C., Eds.; Springer: Cham, Switzerland, 2015. [CrossRef]
31. Kristan, M.; Matas, J.; Leonardis, A.; Felsberg, M.; Cehovin, L.; Fernandez, G.; Vojir, T.; Hager, G.; Nebehay, G.; Pflugfelder, R. The Visual Object Tracking VOT2015 Challenge Results. In Proceedings of the 2015 IEEE International Conference on Computer Vision Workshop (ICCVW), Santiago, Chile, 7–13 December 2015; IEEE: Piscataway, NJ, USA, 2015. [CrossRef]
32. Liang, P.; Blasch, E.; Ling, H. Encoding color information for visual tracking: Algorithms and benchmark. *IEEE Trans. Image Proc.* **2015**, *24*, 5630–5644. [CrossRef] [PubMed]
33. Smeulders, A.; Chu, D.; Cucchiara, R.; Calderara, S.; Dehghan, A.; Shah, M. Visual tracking: An experimental survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 1442–1468. [CrossRef] [PubMed]
34. Mueller, M.; Smith, N.; Ghanem, B. A Benchmark and Simulator for UAV Tracking. In *Computer Vision—ECCV 2016. ECCV 2016. Lecture Notes in Computer Science*; Leibe, B., Matas, J., Sebe, N., Welling, M., Eds.; Springer: Cham, Switzerland, 2016. [CrossRef]
35. Panadeiro, V.; Rodriguez, A.; Henry, J.; Wlodkowic, D.; Andersson, M. A review of 28 free animal-tracking software applications: Current features and limitations. *Lab. Anim.* **2021**, *50*, 246–254. [CrossRef]
36. Lukezic, A.; Vojir, T.; Cehovin, L.; Matas, J.; Kristan, M. Discriminative Correlation Filter with Channel and Spatial Reliability. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6309–6318. [CrossRef]
37. Farkhodov, K.; Lee, S.H.; Kwon, K.R. Object Tracking using CSRT Tracker and RCNN. In Proceedings of the BIOIMAGING, Valletta, Malta, 24–26 February 2020; pp. 209–212.
38. Magura, T.; Horváth, R.; Mizser, S.; Tóth, M.; Nagy, D.D.; Csicsek, R.; Balla, E. Urban individuals of three rove beetle species are not more exploratory or risk-taking than rural conspecifics. *Insects* **2022**, *13*, 757. [CrossRef]
39. Réale, D.; Reader, S.M.; Sol, D.; McDougall, P.T.; Dingemanse, N.J. Integrating animal temperament within ecology and evolution. *Biol. Rev.* **2007**, *82*, 291–318. [CrossRef]
40. Kortet, R.; Hedrick, A.N.N. A behavioural syndrome in the field cricket Gryllus integer: Intrasexual aggression is correlated with activity in a novel environment. *Biol. J. Linn. Soc.* **2007**, *91*, 475–482. [CrossRef]
41. Schuett, W.; Delfs, B.; Haller, R.; Kruber, S.; Roolfs, S.; Timm, D.; Willmann, M.; Drees, C. Ground beetles in city forests: Does urbanization predict a personality trait? *PeerJ* **2018**, *6*, e4360. [CrossRef]
42. Jones, K.A.; Godin, J.G.J. Are fast explorers slow reactors? Linking personality type and anti-predator behaviour. *Proc. R. Soc. B Biol. Sci.* **2010**, *277*, 625–632. [CrossRef] [PubMed]
43. Labaude, S.; O'Donnell, N.; Griffin, C.T. Description of a personality syndrome in a common and invasive ground beetle (Coleoptera: Carabidae). *Sci. Rep.* **2018**, *8*, 17479. [CrossRef] [PubMed]