Computer Assisted Methods in Engineering and Science
29(4): 397–407, 2022, doi: 10.24423/cames.550

Discrete (Two-State) Modular Hyper-Redundant Planar Manipulator

Ela ZAWIDZKA*, Machi ZAWIDZKI

Institute of Fundamental Technological Research, Polish Academy of Sciences, Warsaw, Poland
* *Corresponding Author e-mail: zawidzka@ippt.pan.pl*

This paper presents a concept of an extremely simple planar manipulator composed of 24 congruent modules. Each module has only two possible discrete positions in relation to the previous module: left $(-\frac{\pi}{6})$ or right $(\frac{\pi}{6})$. However, despite its conceptual simplicity, this manipulator can perform relatively demanding tasks, for example as an inspection device. The manipulator is placed in an experimental environment, and the goal is to place its tip in close proximity to five given points without collisions. Despite the constraints of its motion, the manipulator effectively "crawls" inside the working space and visits assigned points. The control of the manipulator is executed by manual placing to desired configurations and interpolating the intermediate transitions. The preliminary results are promising and show that for certain practical types of tasks, the functionality and precision of this extremely simple manipulator could be sufficient, e.g., visual inspection, provision of survival supplies, placing of explosives, etc.

Keywords: discrete manipulator, hyper-redundant, snake robot, modular.

1. Introduction

Biological snakes are exceptionally well adapted to different environments. This is mostly the result of the high redundancy of the snake mechanisms. In many instances of irregular environments, the bio-inspired robots outperform conventional wheeled, legged or tracked robots. The snake-resembling robots have been researched for a few decades. This type of locomotion was studied already in the 1940s [1], and a half-century later, its rigorous mathematical model was developed. In the late 1990s, a trunk-like locomotors and manipulators were introduced in [2].

A number of various snake-like robots have been built [3]; most of the designs were intended for crawling on the ground [4–8], some of them for swim-

ming [9, 10], and even fewer for both swimming and crawling on the ground [11, 12]. Figure 1 shows an amphibious snake robot designed for planar locomotion: serpentine crawling or swimming.

Fig. 1. A photograph of the modular AmphiBot robot constructed in the School of Computer and Communication Sciences, EPFL Switzerland (https://www.epfl.ch/labs/biorob/).

For both biological snakes and bionic trunks, in various environments, the characteristic type of motion gives this type of manipulators a certain advantage over conventional robotic manipulators. They can operate in geometrically complicated environments not accessible by other approaches. Depending on the required task, various working heads can be installed on such manipulators, e.g., for welding, cleaning, monitoring, etc., as shown in Fig. 2.

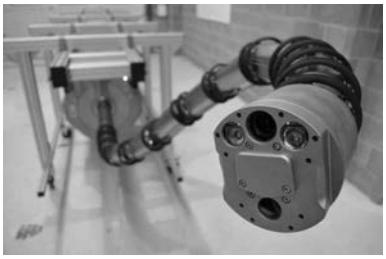

Fig. 2. Oliver Crispin Robotics Ltd.: a modular bionic trunk-like manipulator: an unsleeved, integrated X125 system with an inspection camera and light tool (http://www.ocrobotics.com).

In principle, six degrees of freedom (DOFs) are enough to complete any motion in three-dimensional space: displacement along three Cartesian axes X, Y, Z; and three rotations: yaw, pitch and roll. A conventional industrial manipulator has a low number of DOFs – usually just six. On the other hand, the human arm is the biological archetype of a kinematically redundant manipulator with seven DOFs: three at the shoulder, one at the elbow and three at the wrist. Many robots use this kinematic arrangement. Such robots are called human-arm-like manipulators, e.g., PA-10 robot by Mitsubishi, Lightweight Robot DLR (Deutsches Zentrum für Luft- und Raumfahrt), etc. DEXTER by Scienzia Machinale is an example of an eight-DOF manipulator. Systems with a larger number of joints are called redundant robots, while the term hyper-redundant refers to redundant manipulators with a very large, possibly infinite, number of DOFs [13]. They can be further classified into two groups: vertebrate-like rigid link manipulators, such as snakes, and invertebrate-like continuum manipulators, such as octopus arms or elephant trunks.

In principle, the presented here manipulator has as many degrees of freedom as the number of units less one. This redundancy allows it to perform complicated spatial movements but also may improve the robustness and fault tolerance of the system. The question, however, arises: how far can we simplify this system maintaining a certain level of functionality? Although the relative rotation between each pair of modules is reduced to only two positions: left/right, this system can still be considered as a hyper-redundant-manipulator (HRM) [14]. The inverse kinematics problem for a serial-chain manipulator is to determine the positions of joints given the position and orientation of the end-effector. So-called closed-form solutions are practical because they readily identify all possible solutions faster than numerical methods [15]. The inverse kinematic problem of a typical industrial manipulator can be solved easily [16]. As a result, its control is straightforward. However, HRMs are highly non-linear systems; therefore, their control is by no means straightforward, and requires the application of artificial intelligence methods [17–19]. A method of solving the closed-form solution to the inverse kinematics of a planar redundant manipulator has been proposed in [20]. It was based on employing the Frechet differential of a certain criterion function introduced to resolve the redundancy. However, that model did not include any constraints on the range of motion of the joints – which in fact makes the formulation much more complicated. Moreover, already in the forward kinematics, the analytical description "explodes" with the number of links (DOFs), not to mention the inverse kinematics, which by nature is more difficult. Therefore the proposed approach is not practical for HRMs. The pioneering work of mathematical modeling of a discrete redundant planary manipulator in the Cartesian space was presented in 1989 [21]. In [22], the same author presented a model of the kinematics of a rotary, redundant manipulator, in the form of a finite state machine. An implementation of a heuristic graph searching algorithm for finding collision-free trajectory for a (five-link) planar redundant manipulator was presented in [23]. For the obstacle avoidance problem of planar hyper-redundant manipulators, the so-called "tunneling" approach was presented in [13]. In [24] the same authors presented hyper-redundant robot mechanisms and their applications, including a 30 DOFs hyper-redundant robot. In [25] the dynamics of hyper-redundant manipulators were formulated as a continuum mechanics problem. The advantage of the presented method was that it could be easily parallelized. For more information on this type of manipulators see [25].

The tip of the proposed manipulator crawls like a snake (see Fig. 1); however, it can not be considered as a robotic snake *per se* since its base module is fixed. It can be perceived as an entangled snake by its tail. It also resembles a bionic elephant trunk (see Fig. 2), but unlike a proper bionic trunk, it is constrained to a 2D surface. Thus it can be viewed, alternatively, as a planar robotic trunk.

2. The concept

The modules for the mock-up prototype have been 3D printed with PLA filament, as shown in Fig. 3.

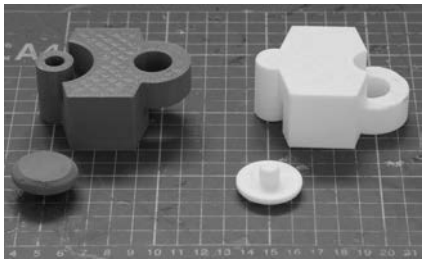

Fig. 3. A photograph showing the top (on the left) and bottom (on the right) of the 3D-printed mock-up prototype. The units are centimeters.

The presented manipulator is a chain of 24 congruent modules linked together by pivot joints. Figure 4 shows the individual module and illustrates the way of connecting the modules together.

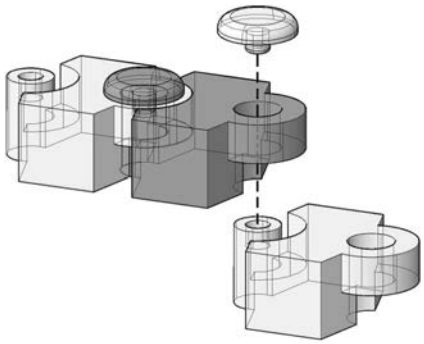

Fig. 4. Isometric view of the modules: two modules linked together (on the left) and exploded view showing the connection (on the right).

In general, each module can have a continuous relative twist with respect to the previous module. The practical angle of twist seems to be in the range $\left[-\frac{\pi}{2}, \frac{\pi}{6}\right]$. In the presented case the range is more narrow, from $-\frac{\pi}{6}$ to $\frac{\pi}{6}$. However, to make the manipulator as simple conceptually as possible, the relative states have been discretized to only two possible positions, namely: $-\frac{\pi}{6}$ (called here "0") and $\frac{\pi}{6}$ ("1").

Since the base module is fixed, there are 2^{24-1}, which is over 8 million (8 388 608) possible configurations. Since the manipulator should not: self-intersect and violate the environment, the number of allowable configurations is smaller and depends on the geometry of the given environment.

3. The experimental setup

The experimental environment is defined by the geometry of the given space and the location of the points to be inspected as well as the placement of the base module of the manipulator. At first, the manipulator must crawl inside the inspection space. Next, the tip of the manipulator must visit five given points (A–E). Finally, it must crawl out of the inspection space.

Each configuration of the manipulator can be simply encoded in the binary list of 0s and 1s starting from the base and ending at the tip. Figure 5 shows the experimental setup with the mock-up manipulator in the initial configuration (before entering the test area A–E) with its binary and hexadecimal representations.

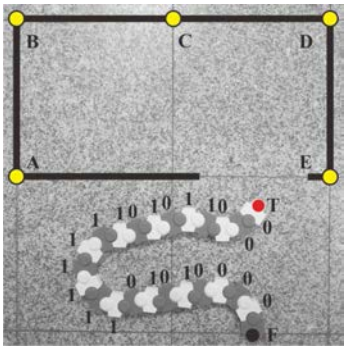

Fig. 5. The experimental setup: the given environment with the inspection space with the five inspection points (shown in yellow); the fixing point (F) of the base of the manipulator. The tip of the manipulator (T) is shown in red. Manipulator is in the initial configuration {0 0 0 0 1 0 1 0 1 1 1 1 1 1 1 0 1 0 1 1 0 0 0}.

4. The experiment

The experiment has been performed by setting manually the mock-up prototype with the fixed base module in seven main configurations:

1) the initial configuration (manipulator clear of the test area) – Fig. 6.1,
2) the tip of the manipulator enters the test area – Fig. 6.3,
3) the tip reaches close proximity of the first test point A – Fig. 6.10,
4) the tip reaches close proximity of the second test point B – Fig. 6.12,
5) the tip reaches close proximity of the third test point C – Fig. 6.15,
6) the tip reaches close proximity of the fourth test point D – Fig. 6.18,
7) the tip reaches close proximity of the last test point E – Fig. 6.20,
8) manipulator reverts to the initial configuration (clear of the test area) – Fig. 6.24.

Figure 6 collects all 24 key configurations.

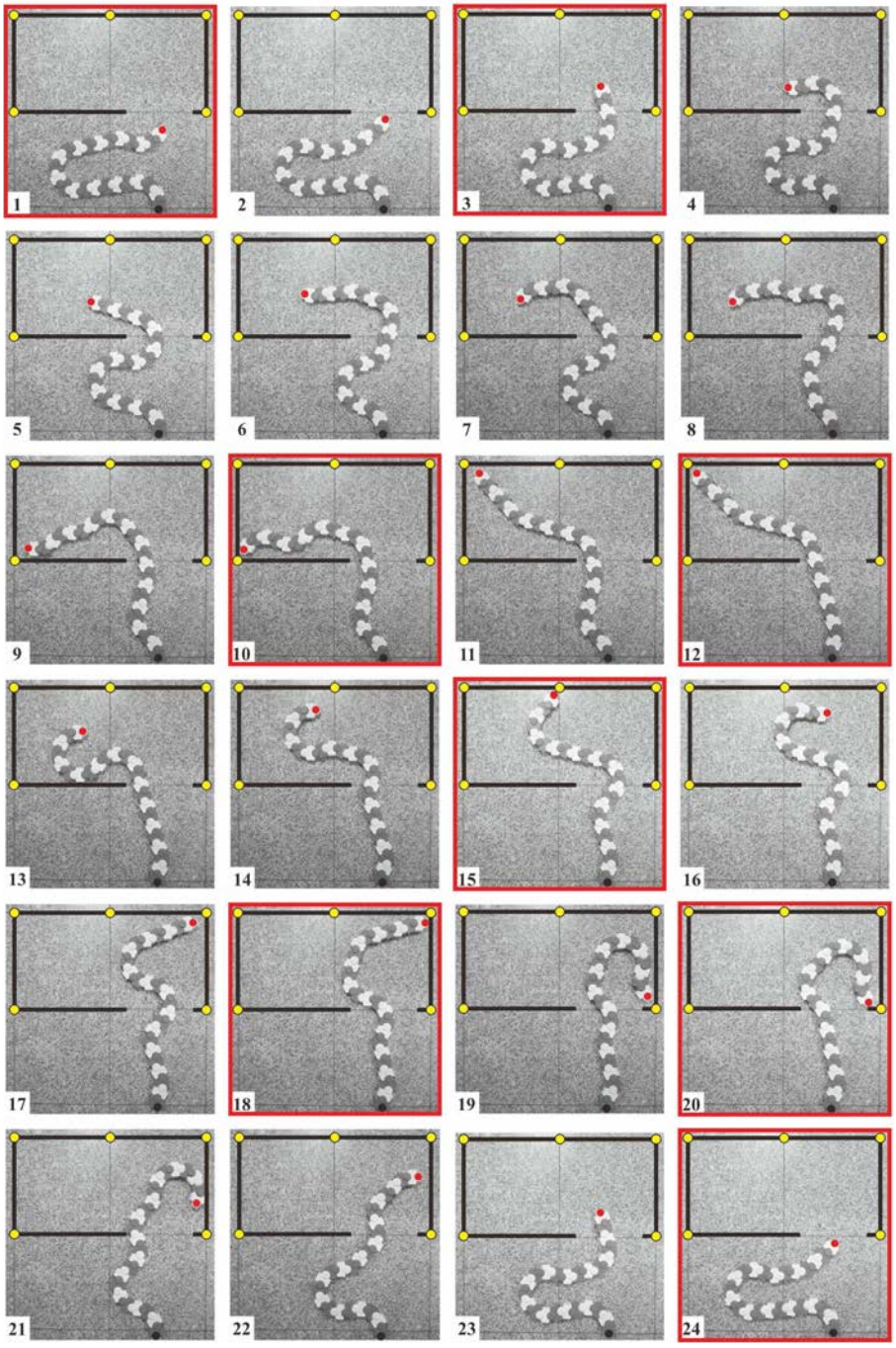

Fig. 6. Seven main configurations (highlighted in red) and the intermediate key configurations of the experiment have been set manually.

As mentioned above, the units have discrete positions; however, it is necessary to check what happens during the transitions. Therefore, each key configuration has been replicated in the *Mathematica* environment using a built-in function **AnglePath**, as shown in Fig. 7.

```
With[{s = 1.08, zero = {-.35, .6}, init = {-.3, 1.4}, img = img12},
 With[
  {anglePath =
    s AnglePath[{init, 91.5 °}, {1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1} /. {0 → 30 °, 1 → -30 °}]},
  Graphics[{{EdgeForm[None], Opacity[0], Red, Rectangle[{-14.22, 0.6}, {4.2, 19.15}]},
    {EdgeForm[Thick], Opacity[0], White, Rectangle[{-15, 0}, {5, 20}]}, Black, Disk[#, .2] & /@ anglePath,
    {Thickness[0.005], Line[{zero, s init}], Line[#] & /@ Partition[anglePath, 2, 1], White, Disk[zero, .2]},
   Prolog → Inset[img, {-15, 0}, {0, 0}, 20], ImageSize → {500, 500}, Axes → False]]]
```

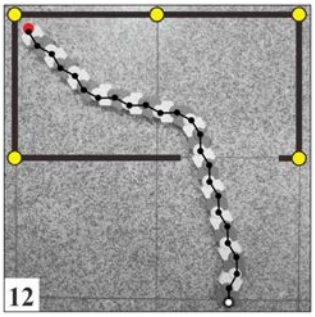

Fig. 7. Each configuration has been replicated in *Mathematica*. The code highlighted in blue shows the part responsible for generating the desired configuration, the rest of the code deals with styling and alignment of the graphics superimposed on the photograph. The camera was not positioned perfectly, which resulted in a certain inaccuracy in the image.

This allows to generate the straightforward transition from one configuration to another at arbitrary number of steps. For example, let us consider the first two configurations. Simple subtraction of the respective genotypes identifies the differences between them:

$$\{0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 0\ 0\ 1\ 0\} \text{ (second configuration)}$$
$$-\{0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ 0\} \text{ (previous/first configuration)}$$
$$= \{0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ -1\ 0\ 0\ -1\ 0\ 1\ 0\}.$$

This means that for this pair the only units whose positions differ are: the second last, fourth last, seventh and eighth last. This method is applied to all 24 key configurations and gives quasi-continuous operation of the manipulator. Note that here the rotation rate is the same for each rotating module. Figure 8 shows the corresponding envelope. For an animation see [26].

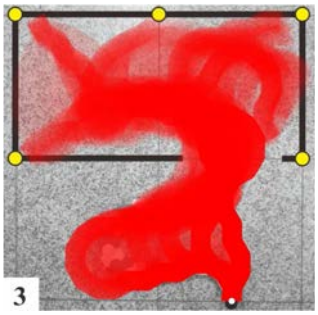

Fig. 8. Red indicates the envelope of all configurations during the experiment including the generated intermediate transitions. The fixed (base) unit is indicated by the white dot.

5. Discussion

The presented physical model and the experiment are quite realistic but not very precise: the modules have been 3D printed and are not as perfect as the CAD model, which results in certain errors in their angular positions. The base module was not actually fixed to the working space which resulted in certain inaccuracies during the manual setting of the configurations.

As Fig. 8 indicates, the manipulator slightly violated the constraints. This can be avoided in a number of ways, e.g.:

- by manually setting intermediate "safe" configurations between the problematic configurations,
- by implementing a more sophisticated method for computing the transitions between given configurations.

A manipulator composed of 13 such modules would already self-collide in two possible configurations, as this problem obviously grows with the number of modules. In this experiment the manipulator has as many as 24 modules. The self-collisions are avoided simply by setting the configurations manually making sure that they are "safe". The transitions between these configurations were also visually inspected.

Wobbling can be a major issue in the case of this discrete-state manipulator. However, it can be minimized in the process of motion optimization. Moreover, it is possible that such a manipulator could be practical only for certain classes of tasks where the smoothness of action is not so important, such as: visual inspection (where the end-effector can be equipped with a gimbal), provision of survival supplies, placing of materials, etc.

More analytic work on the kinematics of this system needs to be done in order to better understand how to effectively and smoothly control this system. Nevertheless, it seems like the use of meta-heuristics will give the best results for this highly non-linear system.

6. Conclusions and future work

Several interesting locomotors and manipulators inspired by snakes and elephant trunks have been constructed in the past. Some of them were also modular, which is particularly relevant in the context of this paper. Although, those systems are very complicated and technically advanced, the goal of our project, however, is to show that robotic systems can be conceptually extremely simple and still perform relatively demanding tasks.

The approach presented here is straightforward and uses physical model and available computational tools. Nevertheless, the results are not idealized but relatively realistic.

Simple interpolation between configurations quickly computes the intermediate configurations; however, the results are not fully satisfactory.

Future work will focus on the optimization of the real-time control of this manipulator. As previous related work indicated [27], the application of heuristic methods seems as a natural direction in this highly non-linear optimization problem. At the moment we are working on the application of combination of rapidly-exploring random tree (RRT) with probabilistic roadmap (PRM) for this type of constrained manipulation.

On the other hand, in the case of highly constrained environments, generating possible configurations of the manipulator by graph-theoretic approach can be effective for finding not only very good solutions but even the ideal ones, as demonstrated in [28]. Another challenge is the structural optimization of the congruent modules, which must withstand the worst scenario loads [29, 30]. It would also be interesting to construct a 3D analog of our 2D manipulator with modules that have a minimal number of discrete positions.

Finally, it is important to construct an operational prototype as the proof-of-the-concept.

Acknowledgements

This research is a part of the project titled *Arm-Z: an extremely modular hyper-redundant low-cost manipulator – development of control methods and efficiency analysis* and funded by *OPUS 17* research grant No. 2019/33/B/ST8/02791 supported by the National Science Centre, Poland.

References

1. J. Gray, The mechanism of locomotion in snakes, *Journal of Experimental Biology*, **23**(2): 101–120, 1946, doi: 10.1242/jeb.23.2.101.
2. S. Hirose, *Biologically Inspired Robots: Snake-Like Locomotors and Manipulators*, Oxford University Press, 1993.

3. A.J. Ijspeert, A. Crespi, Online trajectory generation in an amphibious snake robot using a lamprey-like central pattern generator model, [in:] *Proceedings of the 2007 IEEE International Conference on Robotics and Automation* (ICRA 2007), pp. 262–268, IEEE, 2007, doi: 10.1109/ROBOT.2007.363797.

4. G.S. Chirikjian, J. W. Burdick, Design and experiments with a 30 DOF robot, [in:] *Proceedings IEEE International Conference on Robotics and Automation*, Vol. 3, pp. 113–119, 1993, doi: 10.1109/ROBOT.1993.291862.

5. B. Klaassen, K.L. Paap, GMD-SNAKE2: a snake-like robot driven by wheels and a method for motion control, [in:] *Proceedings 1999 IEEE International Conference on Robotics and Automation* (Cat. No. 99CH36288C), Vol. 4, pp. 3014–3019, IEEE, 1999, doi: 10.1109/ROBOT.1999.774055.

6. G. Miller, Snake robots for search and rescue, [in:] *Neurotechnology for Biomimetic Robots*, J. Ayers, J.L. Davis, A. Rudolph [Eds], MIT Press, pp. 16–21, 2002.

7. H.R. Choi, S.M. Ryew, Robotic system with active steering capability for internal inspection of urban gas pipelines, *Mechatronics*, **12**(5): 713–736, 2002, doi: 10.1016/S0957-4158(01)00022-8.

8. D.P. Tsakiris, M. Sfakiotakis, A. Menciassi, G. la Spina, P. Dario, Polychaete-like undulatory robotic locomotion, [in:] *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 3018–3023, IEEE, 2005, doi: 10.1109/ROBOT.2005.1570573.

9. K.A. Melsaac, J.P. Ostrowski, A geometric approach to anguilliform locomotion: modelling of an underwater eel robot, [in:] *Proceedings 1999 IEEE International Conference on Robotics and Automation* (Cat. No. 99CH36288C), Vol. 4, pp. 2843–2848, IEEE, 1999, doi: 10.1109/ROBOT.1999.774028.

10. C. Wilbur, W. Vorus, Y. Cao, S.N. Currie, A lamprey-based undulatory vehicle, *Neurotechnology for Biomimetic Robots*, MIT Press: Cambridge, MA, USA, pp. 285–296, 2002.

11. H. Yamada, S. Chigisaki, M. Mori, K. Takita, K. Ogami, S. Hirose, Development of amphibious snake-like robot ACM-R5, [in:] *The 36th International Symposium on Robotics* (ISR 2005), Tokyo, 2005.

12. A. Crespi, A. Badertscher, A. Guignard, A.J. Ijspeert, AmphiBot I: an amphibious snake-like robot, *Robotics and Autonomous Systems*, **50**(4): 163–175, 2005, doi: 10.1016/j.robot.2004.09.015.

13. G.S. Chirikjian, J.W. Burdick, An obstacle avoidance algorithm for hyper-redundant manipulators, [in:] *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 625–631. IEEE, 1990, doi: 10.1109/ROBOT.1990.126052.

14. K. Ning, F. Wörgötter, A novel concept for building a hyper-redundant chain robot, *IEEE Transactions on Robotics*, **25**(6): 1237–1248, 2009, doi: 10.1109/TRO.2009.2032968.

15. B. Siciliano, O. Khatib, T. Kröger, *Springer Handbook of Robotics*, Vol. 200, Springer, 2008.

16. R.M. Murray, Z. Li, S.S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, CRC Press, 1994.

17. M. Rolf, J.J. Steil, Efficient exploratory learning of inverse kinematics on a bionic elephant trunk, *IEEE Transactions on Neural Networks and Learning Systems*, **25**(6): 1147–1160, 2014, doi: 10.1109/TNNLS.2013.2287890.

18. A. Melingui, C. Escande, N. Benoudjit, R. Merzouki, J.B. Mbede, Qualitative approach for forward kinematic modeling of a compact bionic handling assistant trunk, *IFAC Proceedings Volumes*, **47**(3): 9353–9358, 2014, doi: 10.3182/20140824-6-ZA-1003.01758.

19. V. Falkenhahn, A. Hildebrandt, R. Neumann, O. Sawodny, Dynamic control of the bionic handling assistant, *IEEE/ASME Transactions on Mechatronics*, **22**(1): 6–17, 2017, doi: 10.1109/TMECH.2016.2605820.

20. M. Galicki, A closed solution to the inverse kinematics of redundant manipulators, *Mechanism and Machine Theory*, **26**(2): 221–226, 1991, doi: 10.1016/0094-114X(91)90085-I.

21. W. Jacak, A discrete kinematic model of robots in the Cartesian space, *IEEE Transactions on Robotics and Automation*, **5**(4): 435–443, 1989, doi: 10.1109/70.88058.

22. W. Jacak, Strategies of searching for collision-free manipulator motions: automata theory approach, *Robotica*, **7**(2):129–138, 1989, doi: 10.1017/S0263574700005439.

23. M. Galicki, A. Morecki, Finding collision-free trajectory for redundant manipulator by local information available, [in:] *RoManSy 9*, Lecture Notes in Control and Information Sciences, Vol. 187, pp. 61–71, Springer, Berlin, Heidelberg, 1993, doi: 10.1007/BFb0031432.

24. G.S. Chirikjian, J.W. Burdick, Hyper-redundant robot mechanisms and their applications, [in:] *Proceedings IROS '91: IEEE/RSJ International Workshop on Intelligent Robots and Systems '91*, pp. 185-190, Vol. 1, 1991, doi: 10.1109/IROS.1991.174447.

25. G.S. Chirikjian, J.W. Burdick, A hyper-redundant manipulator, *IEEE Robotics & Automation Magazine*, **1**(4): 22–29, 1994, doi: 10.1109/100.388263.

26. M. Zawidzki, *Animation of the Snake 2D*, 2022, https://www.youtube.com/watch?v=P_70iVP5DCQ.

27. M. Zawidzki, J. Szklarski, Transformations of Arm-Z modular manipulator with particle swarm optimization, *Advances in Engineering Software*, **126**(C): 147–160, 2018, doi: 10.1016/j.advengsoft.2018.05.003.

28. M. Zawidzki, Retrofitting of pedestrian overpass by Truss-Z modular systems using graph-theory approach, *Advances in Engineering Software*, **81**: 41–49, 2015, doi: 10.1016/j.advengsoft.2014.11.004.

29. M. Zawidzki, L. Jankowski, Optimization of modular Truss-Z by minimum-mass design under equivalent stress constraint, *Smart Structures and Systems*, **21**(6): 715–725, 2018, doi: 10.12989/sss.2018.21.6.715.

30. M. Zawidzki, Ł. Jankowski, Multiobjective optimization of modular structures: Weight versus geometric versatility in a Truss-Z system, *Computer-Aided Civil and Infrastructure Engineering*, **34**(11): 1026–1040, 2019, doi: 10.1111/mice.12478.

Received March 21, 2022; revised version October 30, 2022; accepted December 1, 2022.