MedCrave
Step into the World of Research

International Robotics & Automation Journal

Research Article

Open Access

CrossMark

Optimal solutions for the last three stages of paid version of Lightbot

Abstract

Lightbot is an interesting game app that has strong educational purpose to guide kids at young age to learn programming skills. It has a free version named Lightbot Code Hour consisting of 20 levels we studied in the past. It also has a paid version consisting of 50 levels with some more complicated rules. Although it is not too hard to solve the puzzles, finding the optimal solutions with the smallest number of blocks is non-trivial for some levels. In this paper, we provide a complete list of the block numbers for optimal solutions for the last three stages in the paid version, as well as the detailed analysis of all the non-trivial levels in these three stages. The overall optimal number for 6 stages is found to be 471.

Keywords: lightbot, optimal solution, periodicity

Volume 12 Issue 1 - 2026

Songming Hou,[1] Jianning Su,[2] Yan Huang[3]
[1]Program of Mathematics and Statistics, and Center of Applied Physics, Louisiana Tech University, USA
[2]Department of Mathematics, Computer Science, and Engineering, Georgia State University Perimeter College, Georgia
[3]Beijing AndEngine Software, China

Correspondence: Songming Hou, Program of Mathematics and Statistics, and Center of Applied Physics Louisiana Tech University, Ruston, Louisiana, 71272, USA

Received: March 15, 2026 | **Published:** March 31, 2026

Introduction

Over 30 or 40 years ago, most kids did not have computer access daily.[1,2] The situation has completely changed. Many families have at least a computer at home now.[2] Further, there are several other electronic devices that can serve almost as a computer. Kids at young age do have access to such devices. However, they might not be interested in learning programming skills. On the other hand, most of them are interested in playing games. Extensive studies have been done on serious-game designs.[3–5] In particular, Lightbot is a good tool to motivate students and kids at young age to learn programming skills through playing game to give them an advantage entering college. It introduces programming concepts like loops and subroutines. It has a free version named Lightbot Code Hour consisting of 20 levels and a paid version with 50 levels. The goal is to light up all the lights using a robot by giving it fixed instructions. It has been studied in.[6,7] However, optimal solutions are not discussed until the recent work.[8,9] Optimal solutions found add up to 173 blocks after a rigorous check by the computer for the free version. The paid version was not systemically studied in the literature until our recent work[10] that studied the first three stages.

When looking for optimal solutions for Lightbot levels, the idea of periodicity comes into play which is shared by another game named the Rubik's Snake.[11] The applications of the Rubik's Snake include the study of protein folding[12] and for the construction of reconfigurable modular robots.[13] Some mathematical problems including periodicity concerning a Rubik's Snake have been studied.[14,15] A comic book was also published to make some basic research accessible for kids.[16]

Further, the study of shortest path of Rubik's Snake knots[17–21] also utilized periodicity and share the same scheme of optimization with this work. We borrow some ideas from the study of Rubik's snake to carefully study the optimal solutions for Lightbot paid version and provide a complete list of the block numbers for optimal solutions of the first three stages, as well as the detailed analysis of all the non-trivial levels in these stages.

The difference between the paid version and the free version is that several new rules are introduced. In Stage 3 of the paid version, the elevator and the transport are introduced. In Stage 5, paints and return command are introduced. These make finding the optimal solution a lot more challenging.

The rest of the paper is organized as follows: in Section 2, we give a complete list of number of blocks needed for the paid version. In Section 3, we explain the non-trivial

levels 4-5, 4-6 and 4-8 in Stage 4 in details. In Section 4, we explain a non-trivial level 5-6 in Stage 5 in details. In Section 5, we explain all levels in the challenging Stage 6. A new stopping criteria is introduced in Section 6 to handle the more complicated rules in Stages 5 and 6 compared with earlier stages. We conclude in Section 7.

A complete list of the number of blocks needed for optimal solutions

In this section, we provide a complete list of the number of blocks for optimal solutions. All these optimal solutions are verified by our computer code that they do solve the problems and they cannot be further improved. Players can use the information to compare with their own solutions.

For Stage 1 levels, the number of blocks are 3+7+8+6+7+11+12+12+12=78.

For Stage 2 levels, the number of blocks are 8+12+16+10+8+9+9+11+13=96.

For Stage 3 levels, the number of blocks are 9+11+9+10+4+8+9+8+11=79.

For Stage 4 levels, the number of blocks are 4+5+9+9+10+10+8+12+13=80.

For Stage 5 levels, the number of blocks are 5+7+6+8+8+9+8=51.

For Stage 6 levels, the number of blocks are 13+20+17+9+9+11+8=87.

For example, if the reader would like to find the optimal number of blocks for level 6-3, just check the third number above in Stage 6 and it is 17.

Overall, the verified optimal solution is 78+96+79+80+51+87=471. See Figure 1.

Figure 1 The overall optimal solution for Lightbot paid version.

Optimal solutions for the first three stages were reported in our previous work.[10] In the following sections, we focus on explaining the non-trivial levels from the last three stages. Stage 4 is mostly concerned with loops. In Stage 5, new rules about paints and return command are introduced. Stage 6 is a collection of challenging levels and we will explain everyone in it.

Stage 4

We will explain the non-trivial levels 4-5, 4-6 and 4-8. The idea behind optimal solutions for level 4-5 is periodicity and rotation. We have seen these ideas in.[8–10] Recall that an elevator is activated by a light command when the robot is on it, and it will either go up two layers or go down four layers if it reaches the top. Figure 2 shows an optimal solution and Figure 3 gives detailed explanation of the periods. We could see the beauty of symmetry. Purple color means the light is on, off, then on again. In the end, all lights are on.

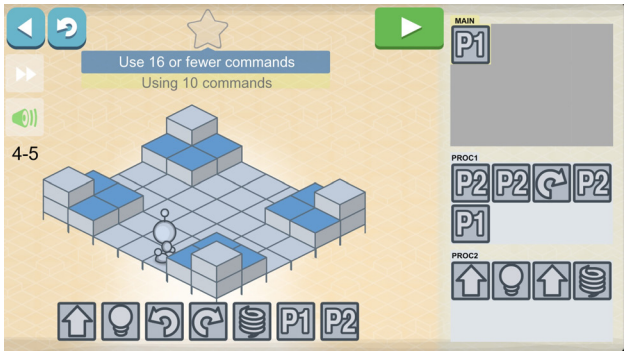

Figure 2 Level 4-5.

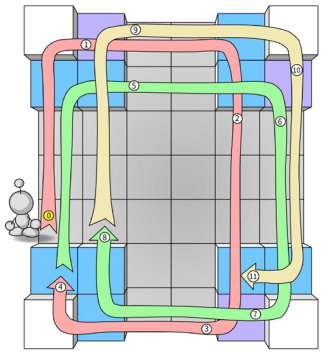

Figure 3 Level 4-5 explained.

For level 4-6, note that a transport does not change the orientation of the robot. If one makes forward, light, forward, light, right and repeats 3 times, the orientation is opposite to what it is supposed to be and two turns are needed. This way, 12 blocks are needed (using both P_1 and P_2). The optimal solution is only 10 blocks. The idea is to construct a sequence of forward, light, forward, right, light, forward, right, for-ward, right so that the robot can actually have the needed orientation after transportation. Some lights are on, off, then on. After each 3*2+1=7 periods (that is, calling P_1 7 times), the robot advances to the next unit. With 7*3=21 periods, all lights are on. Figure 4 shows this optimal solution. Our code verified that this cannot be further improved.

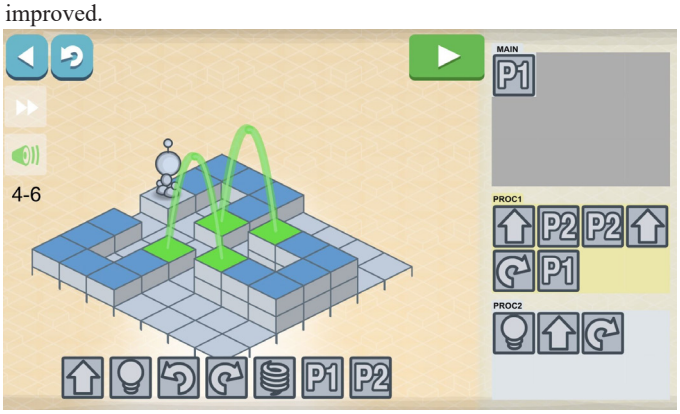

Figure 4 Level 4-6.

Although the map of level 4-8 is simple, it is worth studying cause we could generalize this optimal solution to solve $3 - by - n$ planar maps. We have set up a loop that could light up 3 lights in a same column and advance to the next column. No matter what n is the map is swept (Figure 5).

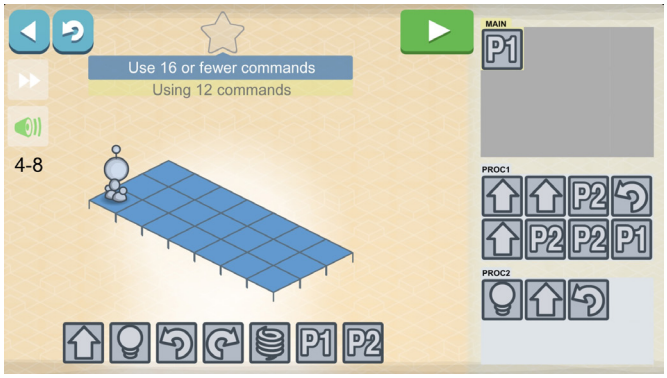

Figure 5 Level 4-8.

Stage 5

We explain a non-trivial level 5-6. Some other levels are similar. The main focus in the stage is the introduction of paints. (We will mention the return command in 6-6.) A white robot (initially) becomes red or green if a light command is executed on a red or green block, respectively. A red or green robot, instead, will become white if a light command is executed on a red or green block (not necessarily respectively). A red or green command block will only be executed if the robot has the same color. This is an interesting practice of if statement and it really helps kids to learn programming skills.

Figure 6 shows an optimal solution. The idea is to use conditional green right turn so that it only turns right when needed. Also, the red left turn and red jump are put together.

Citation: Hou S, Su J, Huang Y. Optimal solutions for the last three stages of paid version of Lightbot. *Int Rob Auto J.* 2026;12(1):48–52.
DOI: 10.15406/iratj.2026.12.00316

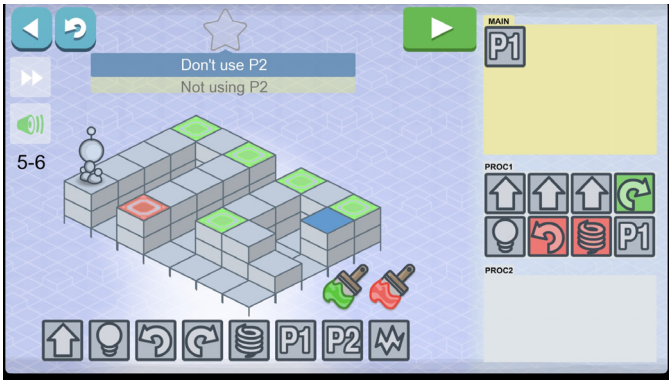

Figure 6 Level 5-6.

Stage 6

This is a list of challenging levels. Therefore we present an optimal solution for each level.

For 6-1, the idea is similar to 4-8. Figure 7 shows an optimal solution for level 6-1.

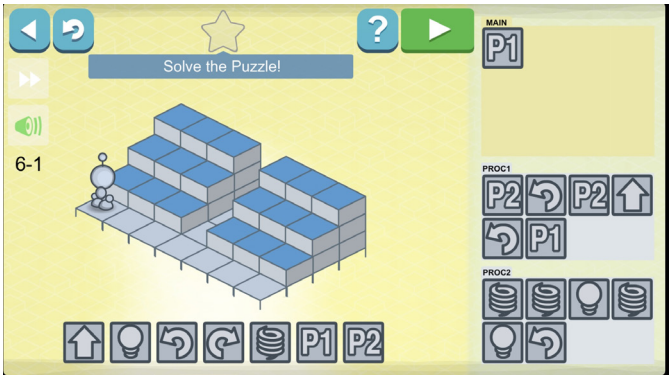

Figure 7 Level 6-1.

For 6-2, the trick is to make a couple of spiral moves down and upstairs. Figure 8 shows an optimal solution for level 6-2.

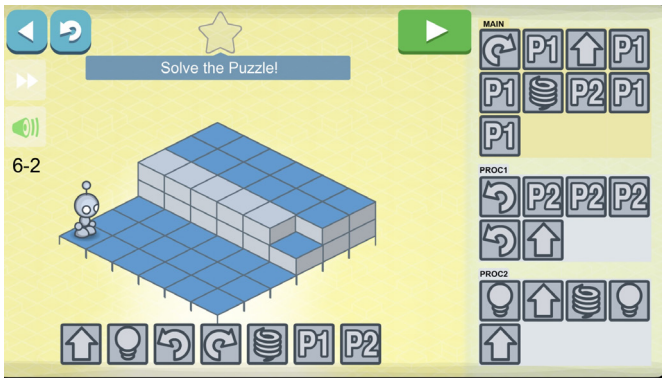

Figure 8 Level 6-2.

For 6-3, it is highly non-trivial due to the irregular shape. Figure 9 shows an optimal solution for level 6-3 found by our computer code. It is not easy for human players to find this, as it is counter intuitive. We know an infinite loop is normally efficient. However, normally that comes with a map with regular geometry: either a translation invariance or a rotation invariance. This level does not have them at all. It turns out that we could use the same sequence of moves to serve multiple purposes as some jump or forward commands are not really executed for certain height difference or at the boundary. Both lights are on after 8 periods. (that is, calling P_1 8 times)

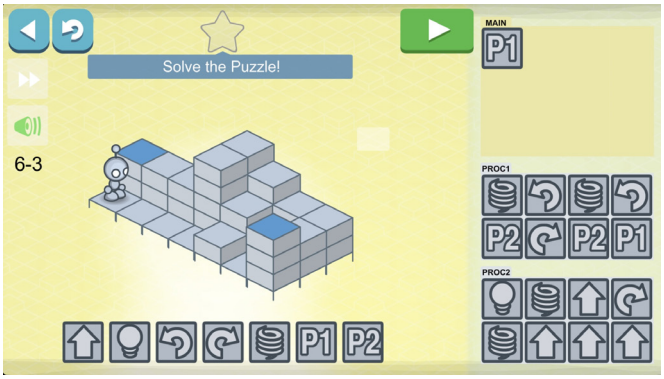

Figure 9 Level 6-3.

For 6-4, the logic is to turn right on a green block and turn right and jump on a red block. Each time the robot turns right, we must reset its color otherwise the robot would have been making unwanted turns. The trick is to use a red P_2 to include light and jump, that way the light command would recover the robot color and the jump command would allow the robot to jump up. Figure 10 shows an optimal solution.

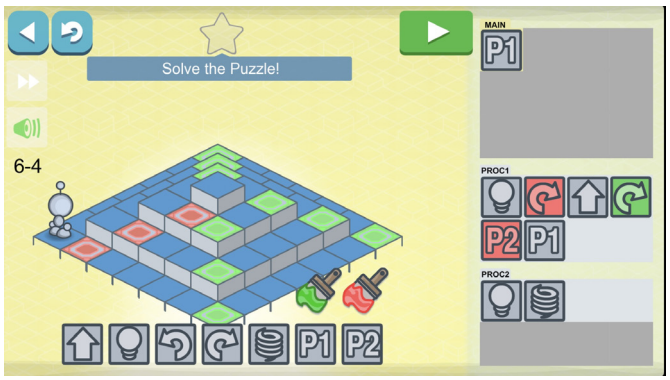

Figure 10 Level 6-4.

For 6-5, we could imagine that we could have a simplified version with just zig-zag moves. "forward, right, forward, left" would have done the job. Now use a couple of green right turns and a light to redirect the robot. Figure 11 shows an optimal solution.

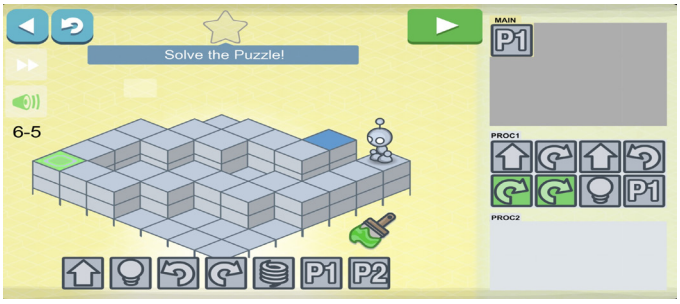

Figure 11 Level 6-5.

The most difficult level is 6-6. Figure 12 show an optimal solution for this level. Even if the optimal solution were presented, it is hard to keep track. Therefore we give detailed explanation using Figure 13. The beauty of center symmetry is revealed. Note that for the first 4 periods, the robot has yet to enter a steady state. After the transient state, we see a center symmetric pattern which is not finished. Before the 44th period is finished, all lights are on and the game is over.

Citation: Hou S, Su J, Huang Y. Optimal solutions for the last three stages of paid version of Lightbot. *Int Rob Auto J.* 2026;12(1):48–52.
DOI: 10.15406/iratj.2026.12.00316

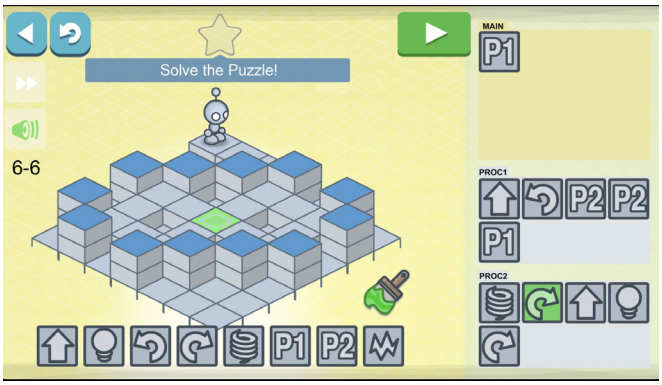

Figure 12 Level 6-6.

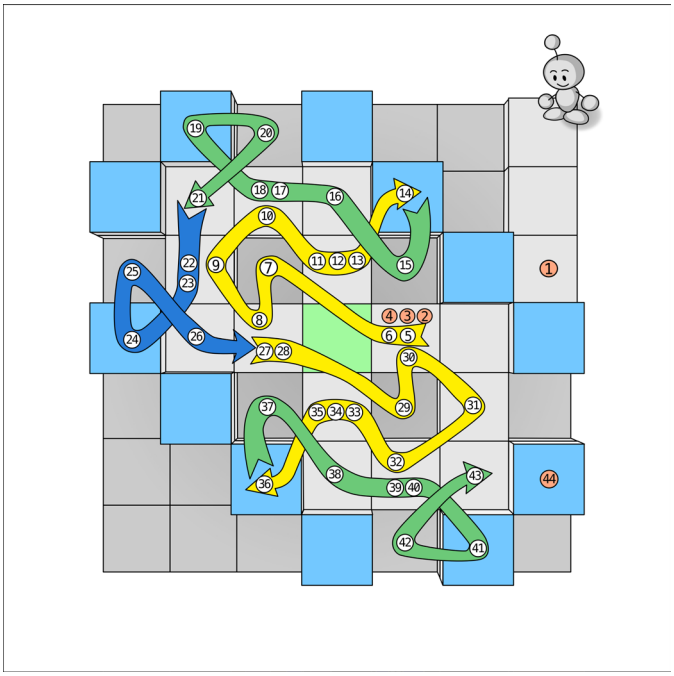

Figure 13 Level 6-6 explained.

Although there is a return command available in this level, it is not used in the optimal solution. The purpose of the return command combined with the paint is for someone to terminate a loop conditionally. However, we have exhausted the search and found that if one uses the return command for this level, then definitely the number of blocks will not be optimal. Still, players could try the return command for some non-optimal solutions for fun. The educational purpose still serves.

For 6-7, we need to pay attention that the robot would not turn red directly from green: it must recover its original color then turn green according to the rule of the game. We could set a regular light, a red light and a green light. We also need a regular forward, a red right turn and a green jump for obvious reason. Putting things in right order, we have Figure 14 to show an optimal solution for this level.

A new stopping criteria

A stopping criteria was discussed in[9] so that the code can terminate efficiently. However, when the rules about paints are added, a new stopping criteria is needed, otherwise we might miss solutions. The reason is that in the past, when we see certain state of history repeats at the beginning of a subroutine, we know we are at the same situation. Now we encounter a new issue. Who called this subroutine indeed matters, as where to return matters. Before the paint (if statement) was introduced, this issue did not exist. The new criteria is a modification that we will not terminate if where the subroutines are called are different. Then we will not miss solutions by assuming history has repeated when it has not. All levels with paints are verified using the new criteria. The optimal number of blocks of all levels of Lightbot paid version are found by our code.

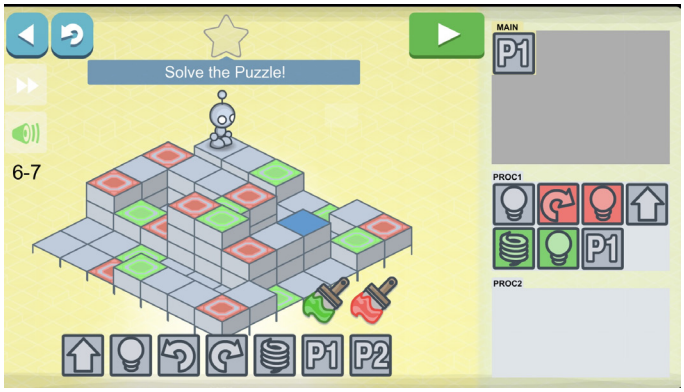

Figure 14 Level 6-7.

Conclusion

Although Lightbot is a game for kids to learn programming skills, finding optimal solutions is non-trivial especially for some tricky levels of the paid version. We gave detailed optimal solutions to the non-trivial levels of Lightbot (paid version) among the last three stages and a complete list of conclusions verified by our computer code for all levels. Overall the optimal solution is 471, which is normally at least 100 less than what players can achieve without careful study based on what we heard from volunteers.

Acknowledgment

We thank Anbo Liu, Sihan Yang, Alexander Wang, Brian Xu, Yin Sun and Ben Zhang for some inspiring discussions and testing the game. S. Hou's research is partially supported by the Walter Koss Endowed Professorship. The title of the professorship is made available through the State of Louisiana Board of Regents Support Funds.

Conflicts of interest

There are no conflicts of interest.

References

1. Households with a computer 40 years ago. 2021.
2. Households with a computer now. 2024.
3. Akkaya A, Akpinar Y. Experiential serious-game design for development of knowledge of object-oriented programming and computational thinking skills. *Computer Science Education.* 2022;32(4):476–501.
4. Arif YM, Ayunda N, Diah NM, et al. A systematic review of serious games for health education: Technology, challenges, and future directions. *Transformative Approaches to Patient Literacy and Healthcare Innovation*. 2024. p. 20–45.
5. de Carvalho CV, Coelho A. Game-based learning, gamification in education and serious games. *Computers.* 2022;11(3).

6. Gouws LA, Bradshaw K, Wentworth P. Computational thinking in educational activities: an evaluation of the educational game light-bot. *In Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education. ITiCSE* . 2013;13:10–15.
7. Gil A, Figueira T, Netto JFM. Extracting learning analytics from Lightbot gameplay sessions. *XXIII Simposio´ Brasileiro de Jogos e Entretenimento Digital (SBGames 2024) - Manaus/AM*. 2024.
8. Hou S, Huang Y. Optimal solutions for Lightbot code hour. *International Robotics and Automation Journal.* 2025;11(1):8–11.
9. Hou S, Su J, Huang Y. Lightbot code hour: optimal solutions, theories and designs. *International Robotics and Automation Journal.* 2025;11(1):19–23.
10. Hou S, Su J. Optimal solutions for the first three stages of Lightbot paid version. *International Robotics and Automation Journal.* 2025;11(3):98–102.
11. Fenyvesi C. Rubik's snake of 'infinite possibilities. *The Washington Post*. 1981.
12. Iguchi K. A toy model for understanding the conceptual framework of protein folding: Rubik's magic snake model. *Mod Phys Lett B*. 1998;**12**(13):499–506.
13. Ding X, Lu S, Yang Y. Configuration transformation theory from a chain-type reconfigurable modular mechanism-Rubik's snake. *The 13th World Congress in Mechanism and Machine Science*. 2011.
14. Hou S, Chen Y, Li Z. Some mathematical problems related to the Rubik's snake. *J Mech Rob.* 2021;13(1):014502.
15. Hou S, Su J, Chen Y. Palindromic, periodic and Möbius Rubik's snakes. *International Robotics and Automation Journal.* 2021;**7**(3):84–88.
16. Pan E, Hou S. *The Legend of the Rubik's Snake*. Lit House LLC. 2025.
17. Hou S, Su J. Shortest paths of trefoil knot designs using Rubik's snakes. *International Robotics and Automation Journal.* 2022;8(1):18–20.
18. Hou S, Su J. Shortest paths of Rubik's snake prime knots up to 5 crossings. *International Robotics and Automation Journal.* 2022;8(2):47–50.
19. Hou S, Su J, Mufutau R. Shortest paths of Rubik's snake prime knots with up to 6 crossings and application to roller coaster design". *International Robotics and Automation Journal.* 2023;9(1):30–33.
20. Hou S, Su J, Mufutau R. Shortest paths of Rubik's snake composite knots up to 8 crossings. *International Robotics and Automation Journal.* 2023;9(3):110–113.
21. Hou S, Su J, Mufutau R. Shortest paths of Rubik's snake composite knots with 9 crossings. *International Robotics and Automation Journal.* 2024;10(1):25–30.

Citation: Hou S, Su J, Huang Y. Optimal solutions for the last three stages of paid version of Lightbot. *Int Rob Auto J*. 2026;12(1):48–52.
DOI: 10.15406/iratj.2026.12.00316