Computer Vision Interface for Symbolic Programming of Cartesian Motion to introduce Visually Impaired Children into Robotic Sciences

Juan P. Cavazos-Carrizales
Robotica y Manufactura Avanzada
CINVESTAV-Saltillo
A.P. 663
25000 Saltillo, Coah.
Mexico
email: juan.cavazos@cinvestav.mx

Francisco J. Ruiz-Sanchez
Robotica y Manufactura Avanzada
CINVESTAV-Saltillo
A.P. 663
25000 Saltillo, Coah.
Mexico
email: fruiz@cinvestav.mx

***Abstract*—Didactic material, aimed at sightless children interested in Robotics, should provide a suitable working environment for their specific sensory capacities and instill a spatial perception of motion and an ability to describe it in a logical and structured way. In this paper, to improve the user experience learning the basics of coding robotic motion in a non-visual environment, we introduce a symbolic programming interface of assembling tokens based on artificial vision recognition that preserves the working space free of obstacles and simplifies the programming process by sliding tokens together on a flat board instead of requiring complex electromechanical connections between them. The computer vision system detects symbols and numbers based on a Boundary Object approach that extracts the main features of the captured images and assigns them to the closest class contained in a reference library. Detected symbols and numbers are identified as part of an ordered array and interpreted as commands with an argument, providing auditory assistance in the case of errors, before being executed in a real mobile robot for instant learning reinforcement. Tokens were ergonomically redesigned for children allowing a simple manipulation for easy assembling and providing tactile and visual information to allow easy recognition by users, their instructors, and the artificial vision system. Throughout the paper, we discuss the design criteria before presenting the final implementation and its first results.**

I. Introduction

Robotic Science is a multidisciplinary research area that considers Robotics and Artificial Intelligence, the study of autonomous mechanisms, and their interaction with humans [1]. Understanding it is considered a strategic advantage in today's world and represents an attractive opportunity for professional fulfillment. However, involvement in Robotics and its related areas requires a clear spatial perception of mechanical motion and the ability to describe it in a logical and structured way; skills that must be instilled from a young age to attain a competitive level.

Children are encouraged to participate in Robotic activities by means of ludic platforms that introduce them to the experience of describing the movements of basic mechanisms and coding algorithms to accomplish ordinary motion tasks. These platforms are, in general, composed of a robotic device, either real or virtual, and a programming interface with intuitive graphic symbols. Early interaction with robotic devices fosters both, the mental representation of moving elements and the capacity to describe them in an abstract language of simple commands. Regrettably, didactic material aimed at children interested in Robotics is mainly visual, excluding visually impaired people from the possibility of early training in Robotics and its related areas[2] [3].

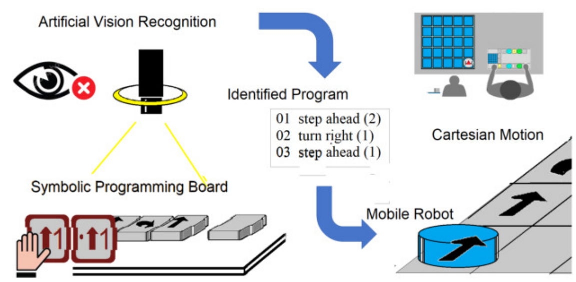

Fig. 1. Computer vision interface for symbolic programming board with assembling tokens describing basic Cartesian movements.

The perception of motion is strongly related to a mental representation of tridimensional objects, including their allowed time-evolving positions inside the space. In general, this mental representation is created from visual information received from the surrounding environment but, in sightless children, this image is constructed by the alternative information received from an improved sensibility to their available senses and it is characterized by being centered in their own body. These representations differ in the spatial description of distant objects making difficult the recognition of moving mechanisms in a sightless condition. It is then required to stimulate the ability to properly perceive and represent the motion in a general referential frame that results from the extrapolation of consecutive small changes in their egocentric frame of reference, and simultaneously describe their displacement in space as a series of single actions, or commands, that can be executed in a logical and structure manner. In this way, didactic strategies for visually impaired children should also consider integrating into the didactic tools the induction of spatial perception in a tridimensional representation [5].

Programming interfaces for sightless people are usually provided with Braile displays, word processors with auditory reading assistance, and haptic interfaces [2], [3], [4] requiring some writing skills and experience using computers which limits the age of possible users. The available robotic platforms for children are provided with a symbolic programming interface and assembling tokens of graphic commands that offer an intuitive use and are accessible even before they learn to read. Thus, the natural extension to improve the design of programming interfaces for visually impaired children involves the use of assembling tokens. However, these tokens must be designed specifically to be manipulated by kids providing easy handling in a controlled and safe vision-less environment, compatible with a simple robotic application [5].

In this paper, to improve the experience of sightless users coding Cartesian movements with a symbolic programming interface with assembling tokens, we introduce an artificial vision interface that assists the user to recognize basic commands of Cartesian motion by interpreting them as a program (Fig.1). The computer vision system is based on a Boundary Object approach that detects numbers and commands of basic egocentric Cartesian movements, integrated into sequences of instructions describing the program, interpreting them, and providing auditory assistance, before being executed in a real mobile robot for instant learning reinforcement. The tokens were ergonomically redesigned for children allowing simple manipulation for easy assembling and providing tactile and visual information for easy recognition by users, their instructors, and by the artificial vision system. Throughout the paper, we discuss the design criteria before presenting the final implementation, and its first results. The use of visual recognition preserves the working space free of obstacles and simplifies the programming process by sliding together the command tokens on a flat board, instead of requiring complex electromechanical connections that forced the user to manipulate the tokens out of the programming board.

II. Computer Vision Interface for Symbolic Programming of Cartesian Motion

The Computer Vision Interface presented in this work is a visual-less programming interface designed to teach visually impaired children the basic concepts of coding in a friendly and safe environment. This interface replaces the electronic interface of the first version of the system (Fig. 2) [5] with a computer vision one that visually recognizes tokens and their linear arrangement to improve the user experience coding in the symbolic programming platform. The introduction of digital processing of images removes the electromechanical connections between tokens (Fig. 3) simplifying the process of coding by sliding the tokens together without requiring to manipulate the tokens out of the surface of the programming

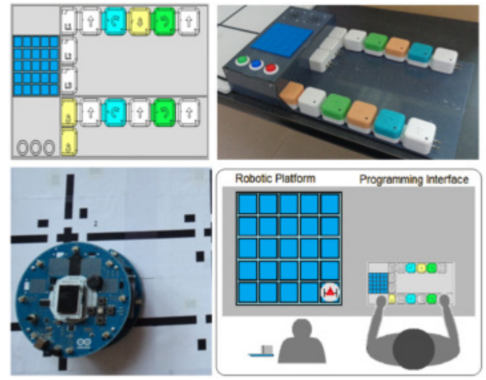

Fig. 2. Electronic interface for symbolic programming platform of Cartesian motion.

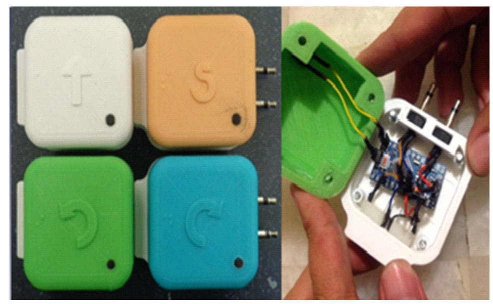

Fig. 3. Command Tokens designed with an electronic identification system requiring an electromechanical connection.

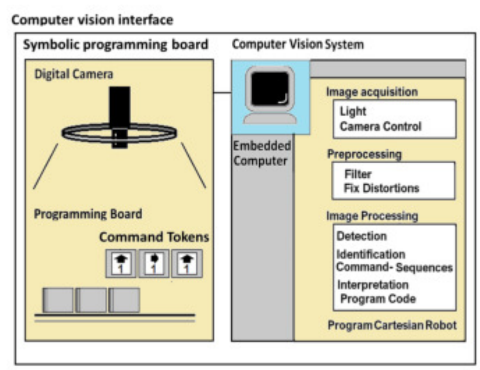

Fig. 4. Computer vision interface for symbolic programming board with assembling tokens composed of two main elements: the symbolic programming board and the computer vision interface.

board. The proposed computer vision interface can be described by its two main elements: A symbolic programming board of assembling tokens in a controlled environment for safety and comfort of the user, and the artificial vision recognition system described in four functional blocks of the image process as shown in Figure 4.

III. Symbolic Programming Board

The symbolic programming board is the work environment for coding using command tokens. It is composed of the programming board and the command tokens.

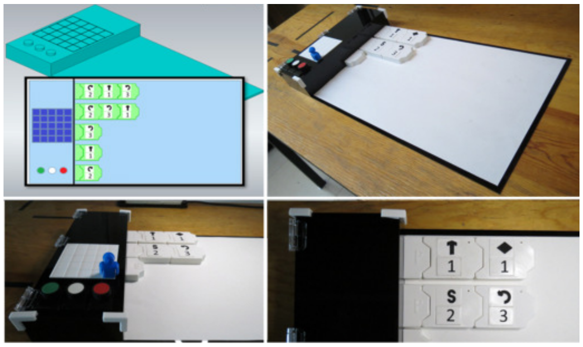

Fig. 5. Programming Board: Control panel and working area for programming.

A. Programming Board

The programming board is a rectangular flat horizontal plane of opaque surface located in a semi-controlled-light environment free of intense luminous sources. Its dimensions are determined by the approximate length of the children's upper extremities, allowing them to freely pick and place the tokens on the whole surface of the board with a comfortable arm position, i.e. allowing the children to reach distant tokens without needing to use their elbows over the table for support. In a similar way as in the previous version (Fig. 2), the programming board is provided with a control panel, a small Cartesian plane for reference, and fixed ports to start connecting the strings of tokens while programming (Fig. 5).

B. Command Tokens

Tokens are symbolic commands or instructions, corresponding to basic Cartesian motions on an egocentric reference frame (step forward, turn left, turn right), and additional programming functions (conditional, subroutines). Commands are represented graphically by intuitive symbols based on arrows that can be easily interpreted by kids even before they acquire reading skills (respectively straight arrow, curved to the left arrow, curved to the right arrow, and additionally letter S for subroutine and a diamond) graved on high relief and painted in the top surface. Additionally, it is indicated the number of times each command should be repeated by the removable plate located below the fixed command plate, this replaces the function of the small button used in the previous version to indicate the number of repetitions. In the case of the subroutine and conditional commands, this plate is used to indicate which subroutine to call. They were designed with visual and tactile information considering both, their main users (visually impaired children and their sighted instructors), and the computer vision system for efficient detection. Different sizes and weights were tested before settling on the current design to maximize ergonomy while handling without affecting the characteristics that make them visually recognizable.

Every token has a generic base, size adjusted to a children's hand, and provided with rounded edges for safe handling, offering an easy-to-fit shape for quick attachment to the ports on the programming board, simple connection between tokens, and support to the removable plates indicating the command and its argument, i.e. the basic Cartesian movements and the number of times this movement is executed (Fig. 6). The commands and their argument are graved and colored for easy tactile and visual identification. Also, the tactile recognition of the tokens is reinforced by including a letter in Braille (A for forward, D for right, I for left, and S for subroutine), and a small mark on the top right corner to determine the right orientation of the token. The command symbol, once selected, is fixed to the base and can not be changed, but the plate with the number is always removable depending on the requirements of the program. Observe that in general, the size, weight, and layout of the token were inherited from the previous version of the interface, which in turn, was designed with feedback from visually impaired people.

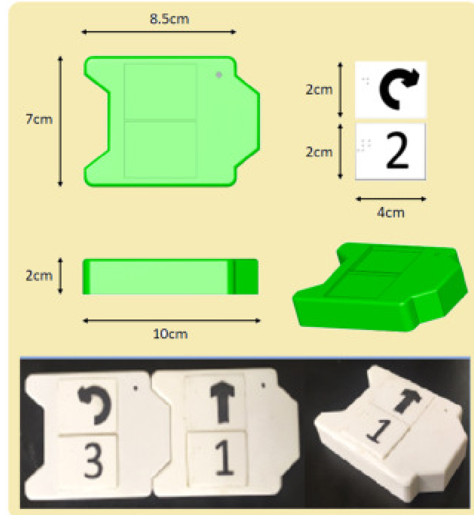

Fig. 6. Assembling Tokens with tactile and visual symbols of commands and their arguments with Braille identifiers.

Regarding the computer vision system, symbols and numbers in the token are black over a white background to add contrast to the images. The font of the letters and their size were selected not only to allow the instructor to recognize the tokens while assisting the children but also to improve the performance of the visual detection, in particular when using a low-resolution camera for reducing cost.

IV. Computer Vision System

The Computer Vision System considers four functional blocks for the image processing applied to the symbolic programming board: image acquisition, preprocessing to improve the visual characteristics needed for detection of saliency objects, digital processing to detect and identify the commands contained in the tokens, and the interpretation of their array as a program to be sent to the robotic platform.

1) Image Acquisition: In order to generate images of the programming board for visual detection, the interface is conditioned to a semi-controlled-light environment emulating

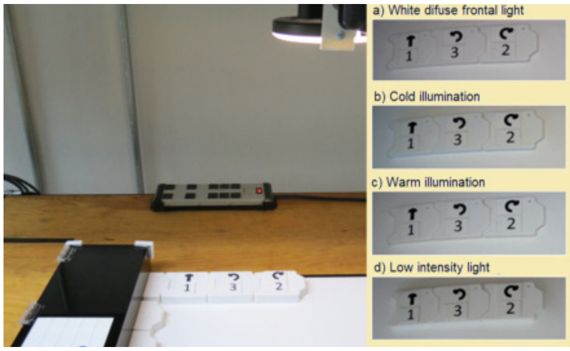

Fig. 7. Semi-controlled-light environment of the programming board and examples of captured images with different light conditions.

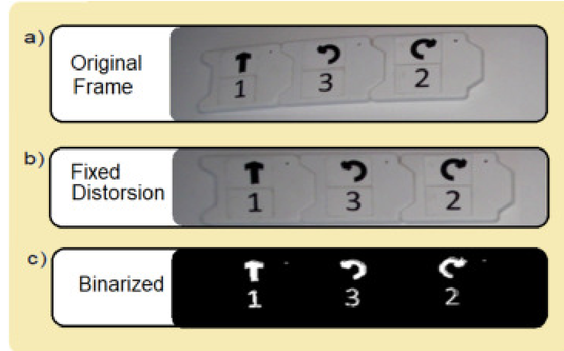

Fig. 8. Example illustrating the result of the pre-processing filter: a) Original frame, b) Fixed distortion, c) Binarization.

the possible real conditions of use in educational centers. Its surface is illuminated by a ring of white light surrounding the camera, providing a uniform, diffused, frontal illumination to avoid undesired lights and shadows that may affect the computer's visual detection (Fig. 7). Light conditions were determined after testing several sources of light and configurations considering the simplest architecture and the best optical conditions. Thus, captured images improve the result of the visual detection by rejecting possible noises that could interfere with it (Fig. 7). Images are acquired by a conventional webcam of manually adjustable focus, whose depth of field allows the capture of sharp images considering its distance to the programming surface and to the upper side of the tokens. In particular, the camera is programmed to stay idle state until the green button on the control panel of the programming board is pressed, triggering the image capture to acquire a picture of 640x480. Observe that the control panel is provided with three buttons, green, white, and red, each identified by a small plate with braille on it, to start the visual identification, pause the program, and stop the program, respectively.

V. Image Pre-processing: Filter and Fix Distorsion

The pre-processing function of the artificial vision system is a simple digital process applied to the images, previous to the main detection process of the system. It is aimed to prepare the acquired images improving their visual properties before being analyzed by the computer vision algorithms. Images are filtered to remove visual noise, such as undesired brights and shadows, that could affect the result of the visual detection, and modified in their optical properties, reducing their size when possible, to improve the effectiveness and the performance of the digital analysis.

The image acquired by the camera is filtered to remove distortion caused by the fisheye effect of the camera. The filter fixes the distortion by estimating the intrinsic parameters of the camera by determining the corresponding point of 3D objects and their corresponding 2D projection. This is achieved by using a calibration pattern obtained from a chessboard pattern of known dimension allowing easy detection of reference points [6], [7]. Then, the image is reduced by extracting an ROI of 573x390, to later be converted from BGR, the original format, to a grayscale of 0 to 255 per pixel. Finally, the image is binarized to a high contrast image using a threshold determined for our specific environment, before being sent to the procession of detection and identification (Fig. 8).

VI. Visual Processing: Detection and Identification

The image processing to detect and identify the tokens is described by a vision algorithm developed to identify the visual commands displayed on the top face of the tokens. This algorithm is based on finding the Boundary Object Functions [8] in the image where the preliminary results are filtered of noise and fixed for distorsion to extract the contours of symbols and numbers before comparing them with a database of preprocessed contours. This is a library of relevant contour information extracted from 619 images corresponding equally to all the possible symbols and numbers, saved in a configuration file.

Contour detection is applied to the binarized images using the Satoshi Suzuki algorithm [9] which results in a collection of coordinates arranged in a vector directing to the points distributed along the edges of any shape found in the image. These contours are filtered by size and color to remove noise caused by shadows and reflections in the original image. Next, the center mass of every contour is determined to calculate the Boundary Object Function, which is the distance from the center of mass to every point in the contour, these distances are saved in a vector and normalized. This is the same process that prepared the contour database used for comparison. Afterward, the BOF vector of the tested image is compared with the BOF vector of every image in the database to determine the closest similarity. The recognition program ends when every symbol in the image was detected and associated with another symbol in the database (Fig. 9).

VII. Commands Interpretation

After every symbol and number in the original image has been recognized and identified as a linear array of commands, the program orders the symbols from left to right and top to bottom and translates them into code recognizable by the

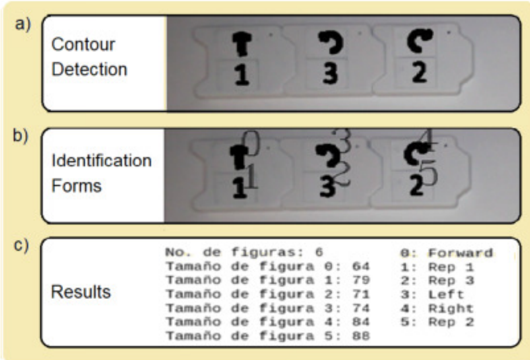

Fig. 9. Example illustrating the result of the processed image detecting the commands and their arguments: a) Contour detection, b) Identified forms, c) Results.

mobile robot. Before being sent to the robot, the translated sequence of commands is checked to ensure that the instructions described are syntactically correct and if they are possible to be executed, depending on the robot's position. If the code cannot be executed, the interface will prompt the user with an auditory signal to check and retry, otherwise, the code is sent and executed by the mobile robot while the central processor receives back, in real-time, the information of the robot to confirm its position in its area.

VIII. Results

In order to test the computer vision interface, in particular, the visual characteristics designed to generate the images (illumination conditions, symbols, colors) and the efficiency of the detection algorithm, we performed a test-bench implementation constraining the Cartesian motion to a single line while granting free access to the embedded system. This allowed us to focus on the problems of the induced optical conditions in the programming board, the performance of the computer vision algorithm while detecting commands and their arrangement in a string of instructions, the translation of the identified program to the mobile robot, and the real behavior of the different components of the system (Fig.10).

A. *Test-bench Implementation*

Tokens, removable plates, and connection ports of the programming board are manufactured in a 3D printer using white PLA material for low weight and high durability, where the symbols on the removable plates are colored in black for increased contrast. The Programming Board was implemented as a flat acrylic surface with an opaque surface in white, where, for the tests, was used without the control box or the connection ports. This was decided to allow free access to the embedded systems and their components, to check their performance and their real working conditions, also, to test the visual detection along the surface of the programming board without the spatial constraints imposed by the connecting ports to start constructing the string of tokens.

The illumination system is a ring of white led light, and the digital camera is a common USB webcam with a maximum

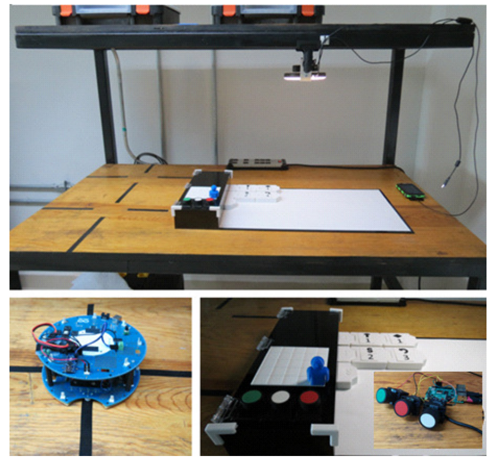

Fig. 10. Implementation of the Computer Vision Interface testbench showing both the mobile robot of the programming platform and the embedded system of the control panel before being installed in the programming board.

resolution of 3840x2160 pixels, frame rate between 16-25 fps, 24bit real color, USB connector, and power input of 5 VDC. A balance on performance price was prioritized when choosing a camera. The computer vision systems (image processing) and the communication control of the system (communication and signals from input and output to interact with the user) are implemented in an affordable and flexible embedded system. The Raspberry PI4 was chosen because of its integrated Bluetooth, IO sockets, and reprogramming capabilities. The Raspberry has USB-C connectors for the camera and illumination setup, ARM Cortex-A72 1.5GHz quad-core processor, Ethernet Gigabit connector, wireless LAN dual-band adapter, and microSD port, in addition to the Bluetooth port used for communication.

The mobile robot is a two-wheel differential drive mobile robot that moves in a step-by-step motion either to arrive at its next forward position or to turn around its axis to the next direction. The robot is controlled with an embedded computer in a move-and-wait mode using infra-red sensors to guide its motion and detect obstacles. It is also provided with sound signals indicating its position and allowing the user to trace its motion. It is wirelessly communicated with the programming interface to receive the program coded by the user.

B. *Results*

The first test is focused on the camera recognition and algorithm. Every symbol and number combination possible was put in the field of view of the camera, changing the location of the tokens on every image to assure reliance when programming on every part of the board. The second test focused on the program readability and translation to instructions understandable by an external device. The processor had to evaluate the syntax of the program, automatically detecting the lines used in the program and if a special command was used, to ensure that it could be executed to then translate it into a sequence of commands executable by the mobile robot used.

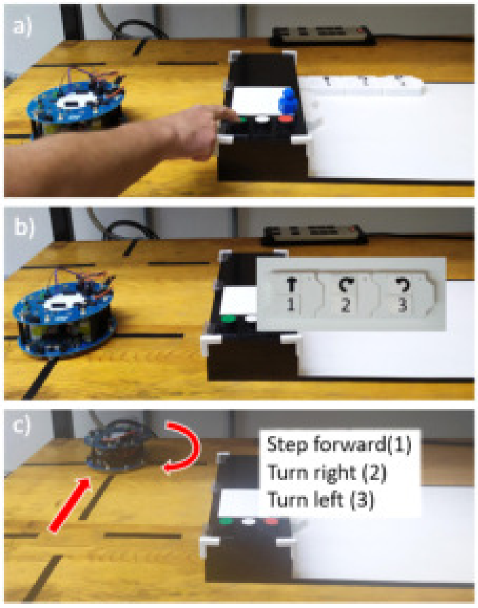

Fig. 11. Example illustrating the result after executing the command Forward step once and Turn right twice: a) Start image capture by pushing the green button, b) Visual identification of the commands and their argument, c) Interpretation as programming instructions, Step forward (1) and Turn right (2) executed by the robot.

The results produced in the test indicated a very high rate of success both in the algorithm recognition of symbols and numbers and in the evaluation and translation of the programmed code as long as the illumination conditions were maintained. The test allowed us to determine some design criteria for the final version of the interface, showing that the color of the tokens does not affect the result of detection as long as the plates of the commands and arguments are black over a white background to preserve contrast. The test also allowed us to determine the dimensions of the working space and the programming board, including the size and font of the symbols for a better performance of the system and ergonomy for the user. We observed that once the illumination changes, or in every new place the interface is set to work, the camera has to be calibrated to ensure reliable working conditions. At this point, we cannot provide a quantitative parameter regarding the accuracy of the visual recognition because in case of failure the result is used to adjust the sensitivity of the system, however, the tests indicate a recognition rate superior to 98% following corrections.

IX. Conclusion

In this paper, we presented a computer vision interface for the symbolic programming platform of Cartesian motion for visually impaired children. This new interface replaces the electronic interface of the previous version of the programming platform, providing a friendlier and easier to use system in a safer working environment. Visual detection simplifies the use of the platform programming by just sliding together the command tokens to form the string of instructions instead of manipulating them to force the assembling of electromechanical connectors. The touchless identification of commands on the programming board preserves the working space free of obstacles that could hinder the mobility of the children when assembling the strings of tokens.

The proposed system required the redesign of tokens for visual detection of commands and their arguments resulting in an improved ergonomic design for small children. On the other hand, the computer vision system was developed and optimized to be implemented with commonly used components, i.e., the 3D printed tokens, the embedded computer, the light system, and the webcam are low specification components, however, the preliminary results show that the system is robust enough to support disturbances encountered in the real conditions of work as in a children classroom.

From the point of view of engineering and manufacturing design, the use of the artificial visual interface instead of the electronic one, not only results in simple and lower manufacturing costs for token production, but the suppression of the electronic devices inserted in tokens eliminates the programming hardware required for the communication protocols to identify the ordered commands in a string of tokens which proved to be inefficient for a constantly changing configuration. This simplification takes into account the comments of interested users signaling the complexity and low performance of the electronic interface as the major drawback of the previous version of the system. Nevertheless, further research is required, in particular, working with volunteers of the potential users of the system, before the final design can be certified as a regular didactic platform to be used in educational centers.

References

[1] K Rajan and A. Saffiotti, *Editorial: Towards a science of integrated AI and Robotics*, Artificial Intelligence no. 247, Elsevier, 2017, pp 1-9.

[2] A. M. Howard, Ch H Park and S. Remy, *Using Haptic and Auditory Interaction Tools to Engage Students with Visual Impairments in Robot Programming Activities*, IEEE Transaction on Learning Technologies, vol. 5, No. 1, January-March 2012, pp 87-95.

[3] N. M. Al-Ratta and H. S. Al-Khalifa, *Teaching Programming for Blinds: A Review*, Forth International Converence on Information and Communication Technology and Accessibility (ICTA), Hammamet, Tunisia, 2013.

[4] V. Jaijongrak, I. Kumazawa, S. Thiemjarus, *A Haptic and Auditory Assistive User Interface: Helping the Blinds ontheir Computer Operations*, 2011 IEEE International conference on Rehabilitation Robotics, Zurich, Switzerland, 2011.

[5] F. J. Ruiz-Sanchez, E. Mireles-Rodriguez, G. Guzman-Solis, *Non-visual Platform for Symbolic Programming of Cartesian Motion to include Children with Visual Impairment into Robotic Sciences*, Inspired Education, Italia, 2018.

[6] Z. Zhang, *A flexible new technique for camera calibration*, Pattern Analysis and Machine Intelligence, IEEE Transactions on, 22(11):1330–1334.

[7] J. Bouguet, *Camera calibration tool box for matlab* Last referenced July 14, 2022, http://robots.stanford.edu/cs223b04/JeanYvesCalib.

[8] E. R. Davies, *Machine Vision: Theory, Algorithms, Practicalities*, Academic Press, 1990.

[9] S. Suzuki and others, *Topological structural analysis of digitized binary images by border following*, Computer Vision, Graphics, and Image Processing, 30(1):32–46.