International Journal of Social Robotics (2024) 16:173–183
https://doi.org/10.1007/s12369-023-01069-y

A Robotic Assistant for Disabled Chess Players in Competitive Games

Luca Pozzi[1,2] · Silvia Guerini[3] · Stefano Arrigoni[2] · Alessandra Pedrocchi[3,4] · Marta Gandolla[1,2]

Accepted: 29 September 2023 / Published online: 1 November 2023

Abstract

Access to regular sports competitions is often precluded for disabled people. Chess, which has been recognized as a sport by the International Olympic Committee in 1999, is a rare exception. Nevertheless, to compete in official tournaments, people suffering from a high level of motor impairment must rely on the assistance of a person to move their pieces on the chessboard, under their indications. This can result in a reduction of the feeling of independence and self-esteem. In this work, a service robot is employed as an assistant for competitive chess players, moving pieces on a standard chessboard for competitions, and adhering to the rules of the international chess federation (e.g. not relying on a custom sensorized chess-set). The robot is controlled through an intuitive graphical user interface. The user interface can be navigated with easy-to-use devices, such as a mouse, a touchpad, or a commodity joystick for motion-impaired people (Ottobock calibratable Mini joystick). An effective framework for the opponent's move identification from RGB-D images is proposed and used to keep track of the live game situation. The application is implemented in ROS on a PAL Robotics TIAGo robot, a service robot with a 7 degrees-of-freedom arm, an extensible torso, and a re-orientable RGB-D camera. The robustness of the application is tested by reproducing six famous chess games several times on a standard wooden competition chessboard, making TIAGo play on behalf of the player with white or black pieces, alternatively. The application is properly working without the need of operator intervention in the 91.6% of the performed moves. The proposed approach successfully opens the door to independent competitive chess playing for motor disabled people in official tournaments.

Keywords Assistive robotics · Chess · Disability · Computer vision

1 Introduction

Disability is very diverse and can manifest in different conditions of the body or of the mind. Most of these conditions generate difficulties for disabled persons to engage in activities and interact with the environment, sometimes even impacting self-esteem and giving rise to a feeling of lack of independence [18]. In this context, assistive robotics aims to support the well-being of people, carrying out tasks that would otherwise be performed by a caregiver, or, even worst, be precluded. In particular, Socially Assistive Robots (SAR) reach this goal by enabling social interaction to achieve measurable progress in convalescence, rehabilitation, learning and ultimately helping in maintaining a positive social life [7]. SAR can change how the disabled person is perceived by others and how s/he feels in a social context, and provide structure for interactions, overall enhancing the social behavior of the subject. The assistance of people with physical impairments [1] or cognitive disorders [15] are just a part of SAR's potential applications. Social robots have been effectively deployed in the rehabilitation context [12] to promote physical and mental well-being [13, 23], and with children in a *teach-through-play* approach [25].

Access to regular sports competitions is often precluded for disabled people. Chess, which has been recognized as a sport by the International Olympic Committee in 1999, is a rare exception, as everyone can play it without disparities if

✉ Luca Pozzi
luca.pozzi@polimi.it

1 WE-COBOT Lab, Department of Mechanical Engineering, Politecnico di Milano, via Gaetano Previati 1/c, 23900 Lecco, Italy

2 Department of Mechanical Engineering, Politecnico di Milano, via La Masa 1, 20133 Milano, Italy

3 WE-COBOT Lab, Department of Electronics, Information and Bioengineering, Politecnico di Milano, via Gaetano Previati 1/c, 23900 Lecco, Italy

4 Nearlab, Department of Electronics, Information and Bioengineering, Politecnico di Milano, piazza Leonardo da Vinci 32, 20133 Milano, Italy

they train for the same amount of time and at the same level. Chess players suffering from a disability are admitted to regular chess competitions, in addition to specifically dedicated tournaments, and can compete against able-bodied players thanks to ad-hoc solutions [9]. As an example, a blind person can play unassisted using a personal chess board with tactile dots on top of the pieces and saying the move out loud to the opponent. Motion-impaired players as well may not be able to move the pieces themselves. In this case, they have to rely on an assistant to play their moves on the official chessboard, an aspect that could negatively affect their sense of independence, and self-esteem.

In this context, this project aims to develop a robotic helper to support disabled chess players in competitive games. The robot would stand as an alternative to the currently involved human assistant, increasing the feeling of independence in the assisted person.

2 Related Works

The concept of a robot playing chess has been around for years. Starting from the 1950s, chess programs were able to calculate the best moves in games, but it was only 1982 when Novag Robot Adversary used a robotic arm to pick up and move the pieces automatically [20]. Chess is often used in board game research due to its high state-space complexity and complex rules-set, which make it a challenging yet achievable problem to test manipulation, Human-Robot Interaction (HRI), and perception capabilities of robotic systems. Most of the current research efforts are meant to create a robotic replica of a chess player [3, 6, 14, 24]. On the other hand, the idea of a robotic assistant for a disabled player has been seldom explored [19]. Indeed, many state-of-the-art robots are programmed for demonstrative purposes only (e.g. ChessKA, KUKA Monstr [4]). The goal of these devices is to play the moves coming from a chess engine in public exhibitions, e.g., to prove the ability of these artificial intelligence algorithms to beat human grandmasters in dedicated events [4]. These applications are not meant to be used in real tournaments, thus simplifying the game-tracking problem through additional sensors on the chessboard itself [14]. Oppositely, other works implement vision to recognize different pieces on the chessboard through machine learning algorithms. As an example, Del Toro and colleagues [6] designed a robotic arm that plays chess against a human opponent. The tracking of the pieces on the board is accomplished by training a convolutional neural network. Similarly, Matuszek and coworkers endowed a custom 6-DoF arm named *Gambit* [17] with an RGBD in-hand camera. Thanks to the visual information, Gambit can track the occupied squares, classify pieces via support vector machine, and detect the presence of hands in the field during the move execution [17]. Pieces movement recognition that does not rely on a machine learning algorithm has been implemented by Chen and Wang [3]. In the addressed work, the humanoid Rethink Robotics Baxter robot has been used to play chess against humans, exploiting computer vision to detect chess movements and using an open-source chess engine to compute the next move.

In the framework of assistive robotics, Omarsdottir and colleagues [19] created Chessmate, a robot intended as a telepresence tool for physically disabled players. This application makes use of a setup very far from the standard competition chess set, using a custom chessboard to manipulate pieces magnetically. Moreover, Chessmate relies on receiving as input the moves of both the players, lacking a system to recognize the opponents' moves.

Another solution to foster inclusion though the game of chess for people suffering from motor impairment is represented by brain-computer interfaces [5, 11, 16]. The aim of these works shares the spirit of the present one, and the effort done to advance research in brain-computer interfaces is commendable. However, these works suffer from the limitation typical of brain-computer interfaces such as long setup time, the need for calibration procedures, and mental fatigue.

3 Contribution

The contribution of this paper is to integrate and adapt existing technologies to develop an application promoting inclusion through sports competition. Ultimately, the use of the robot in official competitions would allow the user to become independent from a human assistant, positively impacting his/her self-sufficiency (both actual and perceived). This insight is supported by the outcome of a survey administered to dystrophic patients to identify the requirements for an upper limb assistive exoskeleton [10]. Amongst other activities of daily living, playing chess independently, not only through the PC screen, arose as one of the potential target tasks. In this view, the proposed robotic assistant for competitive chess players with a motion disability would act as a full-fledged SAR.

This project is intended to possibly find application in the real-world context, since, to date, there is not a similar assistive device approved by regulations in the chess world.

In this view, the project has been carried out following these phases of work. (1) The key requirements for such an application are identified; (2) a prototype solution to fulfill the requirements is implemented and tested; (3) Accessibility for disabled people and compliance with chess rules are taken into account at every step of the project. The described approach introduces these novel features with respect to current solutions:

1. The developed solution should not rely on modification of chessboard or pieces designing them as “special”, but should comply with standard chessboard and pieces;
2. The developed solution should not rely on chess rules or do not allow/correct them given that only the player should be in charge of the game;
3. The developed solution should be under the constant control of the user without the help of an external person, and without using vocal commands so as to not disturb the game.

4 Functional Requirements

The present work aims at building an effective framework to use a robot as a SAR for physically disabled chess players in official chess competitions. As far as we know, to date there are no robots used in the competitive chess world as assistants for disabled people and thus there is a lack of rules about this topic. Nevertheless, the designed robotic assistant has to adhere to International Chess Federation (FIDE) official rules, substituting the human assistant that is nowadays required for disabled chess players to take part in competitions [9]. The minimal hardware requirements of the robot to be employed in the proposed application are (1) a mechanical arm spanning the official chessboard workspace, and (2) an RGB-D camera. When coming to functional requirements, the robot should

(1) Play the moves selected by the user interacting with the standard equipment used in competitive tournaments. Therefore, the robot has to recognize and handle any type of move (i.e., captures, en-passants, castles, and promotions[1]), to place the captured pieces in a dedicated box/area, and to click the clock at the end of the move;
(2) Keep track of the game situation;
(3) Provide an intuitive graphical interface, accessible with a device designed for motion-impaired people;
(4) Be “blind to the rules of chess”, *i.e.* not checking for the validity of the received instructions. Even illegal moves must be played, if requested by the user, in order to recreate a real game situation in which these mistakes are penalized;
(5) Introduce a minimal time overhead, compared with a human assistant, to facilitate user’s acceptance.

[1] To learn more about special moves (and chess rules in general) the reader is referred to the FIDE’s Laws of Chess [8].

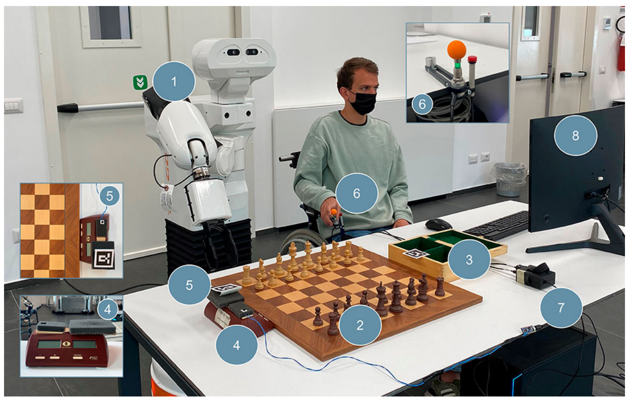


Fig. 1 Complete experimental setup. (1) TIAGo robot, mounting a parallel gripper. (2) The chessboard with the chess pieces placed in their initial configuration. (3) The box to host captured pieces, placed on the left side of the chessboard and with the ArUco marker number 300 placed on it. (4) The chess clock, positioned to the right of the chessboard and with (5) the ArUco marker number 100 placed on TIAGo’s side button. (6) The Ottobock joystick. (7) Joystick and push-button boards. (8) A computer for user interface visualization

5 Materials

The robot employed as assistant in the present application is a PAL Robotics TIAGo robot (Fig. 1, 1). It is composed by a mobile base, a lifting torso, a 7 degrees of freedom arm and a parallel gripper as end-effector. The two parallel plates of the gripper can move independently, reaching a total span of $8cm$. The head is equipped with motors for pan and tilt motions and hosts a RGB-D camera (*Orbecc Astra S camera*). Thanks to its wide range of sensors and peripherals, the TIAGo robot has been employed in several application in the context of assistive robotics [2, 21, 22].
The application has been tested on an official chess competition wooden chessboard (dimensions: 55 cm × 55 cm, 1.5 cm high) and pieces set (Fig. 1, 2). To replicate in a controlled environment the official setup, the complete experimental setup includes a wooden box (Fig. 1, 3) to place the captured pieces and a digital chess clock (Fig. 1, 4). The robot can be controlled through an *Ottobock calibratable Mini joystick*, which is usually employed to drive electric wheelchair and it is very sensible (Fig. 1, 6) linked to an electronic board connected to the control computer (Fig. 1, 8) via standard USB cable. A standard mouse or touch pad can alternatively be used to navigate through the user interface. Additionally, a push button connected to an Arduino Nano Every board (Fig. 1, 7) is mounted over the chess clock and two ArUco markers (binary fiducial markers used in computer vision for pose estimation), printed on cardboard are placed over the clock and over the box (Fig. 1 5 and 3, respectively).

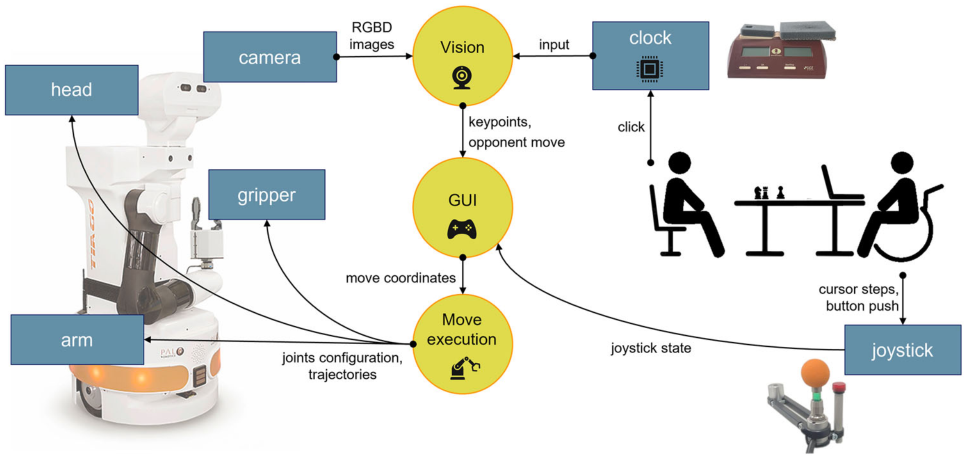


Fig. 2 Graph structure of the implemented application, divided in its functional units

6 Methods

The developed application is constructed by connecting pre-existing ROS nodes deployed on the TIAGo robot integrated with self-developed nodes written in Python language to manage different aspects of the application, namely vision, motion, and user interface. The graph in Fig. 2 schematically shows the final nodes network.

6.1 Vision

The vision system includes RGB image processing of the chessboard, ArUco markers detection, and pointcloud processing.
The image processing step is performed to localize the chessboard, the clock, and the box in the 3D space. At the beginning of the setup preparation, TIAGo's head is oriented downwards, to frame the objects on the table. At this point, the RGB images acquired by the camera are processed to localize closed contours. The biggest contour between the identified ones is identified as the chessboard contour. Then, the Hough lines transform is exploited to extract the square centers. Clock and box localization, on the other hand, relies on the identification of ArUco markers.

To keep track of the pieces positioning on the chessboard, Algorithm 1 is implemented. The tracking algorithm assumes that the game starts from the known, conventional configuration. The user moves are known, as they are given as input to the robot. Therefore, a differential approach is adopted to recognize the opponent's move by looking at the changes in the chessboard configuration compared to the previous turn.

In detail, two RGB images are acquired before and after the opponent's turn (Fig. 3a and b, respectively). A homographic transform is applied to both the images, in order to change them to a bird-eye view. The Structural Similarity Index Measure (SSIM) [26] is used to compare the two images (Fig. 3c). The similarity map given by the SSIM is binarized, to determine if each pixel has changed or not with respect to the previous state (Fig. 3d). When this procedure returns only two cells, their status at the previous turn is checked. The cell formerly occupied by an opponent piece is identified as the start square ($move_i$), and the other one (either white or empty) is identified as the destination square ($move_f$). In some cases, due to the perspective view of the camera, more than two squares are identified by the SSIM (Fig. 3f). To solve this ambiguity, the point clouds of the candidate cells are reconstructed to robustly assess whether they are free or occupied, and if so, by which army (Fig. 3g and h). In Algorithm 1, at line 22, the logic to determine occupancy and color of a cell based on its point cloud is omitted, as it is a trivial set of *if/else* statement based on point count and RGB values. It is worth to underline that an entirely point cloud-based solution has been avoided on purpose as it would significantly increase the computational cost of the algorithm, thus negatively affecting the waiting time. In the unlikely case of ambiguity after this step, the user is presented with a random candidate move.

6.2 Motion

The mechanical task that needs to be implemented is a pick-and-place task, which requires the motion of TIAGo's 7 DoF arm in combination with the parallel plates gripper's fingers. Additionally, other minor movements of TIAGo's torso and head are used throughout the complete game of chess, to ensure an optimal view of the chessboard.
The motions of the arm are planned exploiting *MoveIt!*, an *open-source robotic manipulation platform* for motion planning and collision avoidance in ROS. The arm has to lift and lower the pieces over the chessboard, place captured pieces in the box, click the chess clock and, finally, move to a rest pose. Movements over the chessboard mimic the action of a human player, *e.g.* the knight is moved along its typical L-shaped path and the bishop in a diagonal direction.

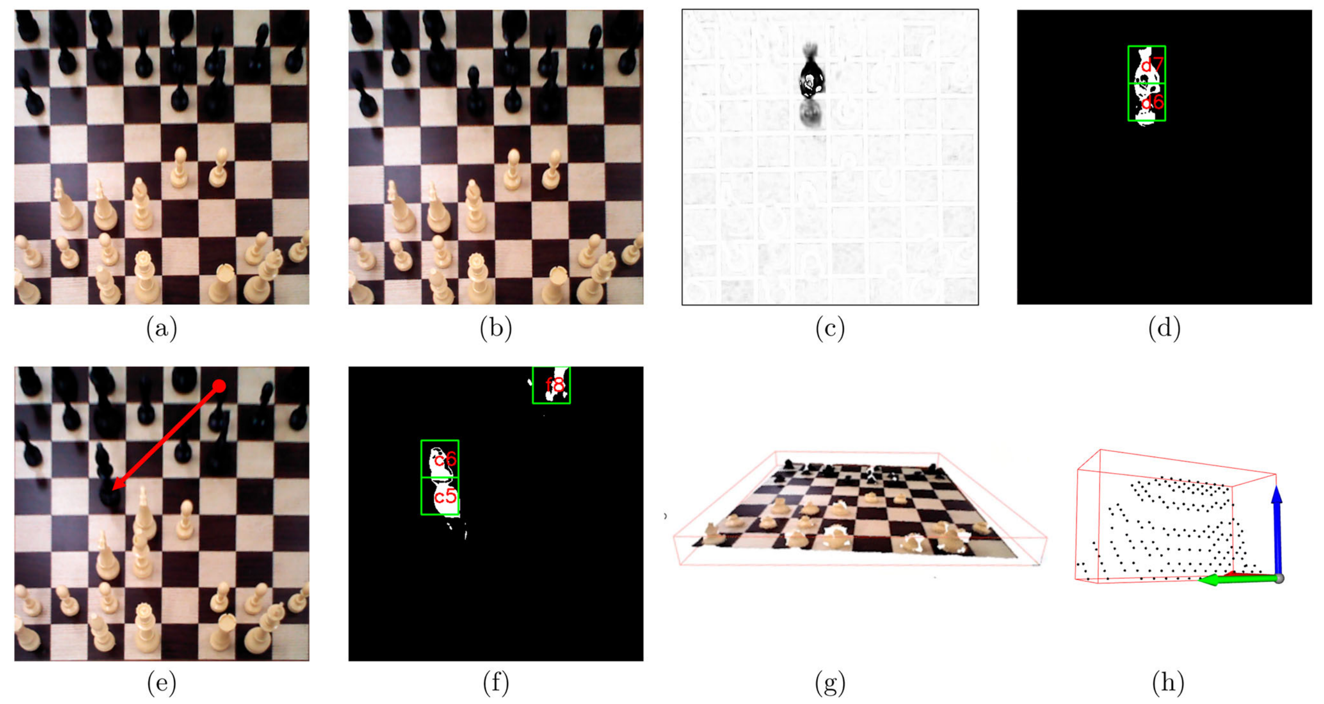


Fig. 3 The first row (a to d) shows the steps of the simplest case for the move detection process. The underlying move is black pawn from d7 to d6. **(a)** RGB image of the chessboard *before* the opponent has played the move (homographic transformation has been applied). **(b)** RGB image of the chessboard *after* the opponent has played the move (homographic transformation has been applied). **(c)** Structural Similarity Index Measure (SSIM) map, resulting from the comparison of (a) and (b). **(d)** Binarized SSIM map. Cell contours and cell names are added for the sake of readability. The second row (e to h) shows the additional steps in detecting more challenging moves when also the point cloud is taken into account. The underlying move is black bishop from f8 to c5 (highlighted by the red arrow). **(e)** RGB image of the chessboard *after* the opponent has played the move (homographic transformation has been applied). **(f)** Binarized SSIM map. Note that the binarization step returns three cells, thus the move cannot be identified with the data at hand. In this example, an ambiguity in the destination cell (either c5 or d5) is observed. **(g)** Point cloud of the scene *after* the move, reconstructed combining RGB and depth image. **(h)** Cropped point cloud of one of the candidate destination cells (i.e. c5) after removing the chessboard plane. In this example, the d5 cell is found to be empty, thus discarded from the destination candidates

The standard pick-and-place task is accomplished with the gripper oriented along the vertical axis, perpendicular to the chessboard plane. However, given some limitations of TIAGo's operational workspace (i.e., the furthest rows are close to the robot's maximum reach workspace; the closest ones require to operate close to the robot torso, thus limiting the feasible configurations), the desired gripper orientation could be unreachable. To mitigate the constraints on the robot positioning with respect to the chessboard, the gripper orientation varies depending on the pick and place locations. Moves in the central region of the chessboard are performed with the gripper in the standard vertical orientation. Two more gripper's orientations (65° and 115° with respect to the vertical axis) are defined, and used to pick and place pieces at the opposite ends of the board.

Note that the vertical orientation should be considered the standard one, as it minimizes the risk of collision with the surrounding pieces. The need for special gripper configurations arises from the geometry of the hardware at hand. Implementing the application on an arm with a wider workspace or a more favorable placement (e.g., a manipulator placed directly on the table) would remove the need for this shrewdness.

Special moves (i.e., en-passant and castle moves) are implemented as combinations of the described elementary movements. At the current stage of development, promotions require the collaboration of the opponent to place the desired piece on the chessboard.

6.3 User Interface

The graphical user interface (GUI), developed with *QtDesigner*, is meant for non-expert users affected by reduced arm mobility. In this view, the GUI is configured to respond to the inputs of the joystick, in addition to standard input devices such as a mouse or a touchpad. In the initialization phase, the intuitive instructions displayed on the screen guide the operator through the preparation of the setup, and let him/her check for the correctness of the calibration procedures. At the start of the game, an instruction wizard window appears, explain-

Algorithm 1 Move identification

```
1: $mapO_{t-1}$ = occupancy map of previous config. Values for each cell
   can be $W$ (white), $B$ (black), $E$ (empty). The user is here assumed
   to play $W$.
2: $RGB_{t-1}$ = color image of previous config.
3: $RGB_t$ = color image of current config.
4: $depth_t$ = depth image of current config.

5: Get similarity map
6: $mapS \leftarrow SSIM(RGB_t, RGB_{t-1})$

7: Evaluate which cells have changed
8: for each $cell$ in $mapS$ do do
9:     $A_{diff} \leftarrow$ number of pixel below similarity score
10:    if $A_{diff} > 0.5 * A_{cell}$ then
11:        if $mapO_{t-1}(cell) = B$ then
12:            append $cell$ to $move_i$
13:        else if $mapO_{t-1}(cell) = W$ or $E$ then
14:            append $cell$ to $move_f$
15:        end if
16:    end if
17: end for

18: Identify the move
19: if $cells$ in $move_i$ or $cells$ in $move_i$ are more than 1 then
20:    for each $cell$ in $move_i$ and in $move_f$ do
21:        Build point cloud of the cell $pcd_{t,cell}$
22:        $mapO_t(cell) \leftarrow W, B$ or $E$ depending on points number
   and RGB value
23:        if $cell$ in $move_i$ and $mapO_t(cell) = W$ then
24:            remove $cell$ from $move_i$
25:        else if $cell$ in $move_f$ and $mapO_t(cell) = B$ or E then
26:            remove $cell$ from $move_f$
27:        end if
28:    end for
29: end if
30: return $move_i$, $move_f$
```

ing how to manage the joystick to navigate the chessboard, select pieces, and control the desired move. Then, during the game, a window displaying the chessboard with the live pieces' disposition is shown (Fig. 4). The application also offers a tool to edit the game situation displayed by the user interface if needed (i.e., if the Algorithm 1 fails to recognize the opponent's move), as displayed in Fig. 4 (label 5).
The user interface is implemented as a ROS node, communicating with the vision and motion nodes to receive the results of the initialization procedure, update the chessboard live situation, and send tasks and targets to the robot.

7 Experimental Protocol

To test the developed platform, six famous chess games (reported in Table 1) have been replicated on a standard tournament chessboard, using TIAGo as an assistant for the other player. The test games include special moves (i.e., castle, en-passant, promotion), captures in the last and first rows of

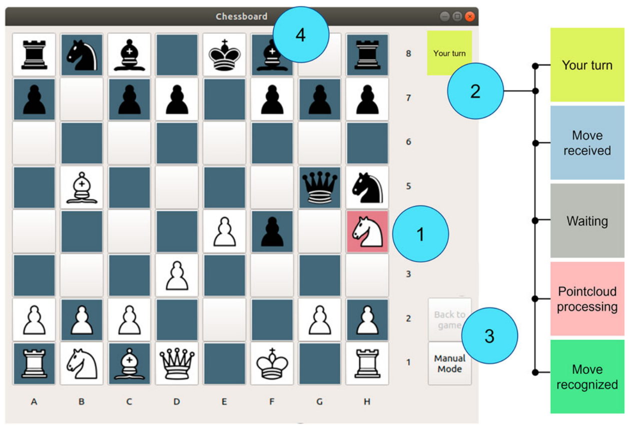


Fig. 4 User interface chessboard window. The game situation is represented on the screen. When the user moves the input device (touchpad, mouse, or joystick) the currently selected square is highlighted in light red (label 1). When selecting a move, the start and end square will be highlighted in dark red and yellow, respectively (not shown in this Figure), waiting for the user to confirm. The user is constantly informed of the application status (label 2). Errors in the move recognition can be manually corrected enabling the "Manual Mode" (label 3). Clicking the button opens a pop-up window with tools to cancel, add or replace pieces on the virtual chessboard. A similar pop-up window opens in case of promotion, *i.e.* when a pawn reaches the last rank on the opposite side (label 4)

Table 1 The six famous games that have been replicated during the experimental tests

Game	Year
M. Carlsen - L. Van Wely	2006
unknown player - G. Greco	1620
S. Polgar - J. Kontra	1982
L. de Kermur - Saint Brie	1750
A. Anderssen - L. Kieseritzky	1851
B. Spassky - R.L. Fisher	1972

the chessboard, and variable duration, to test if the robot can properly handle them.

In the numerical analysis of the outcome measures, a conceptual distinction has been done between *actions* and *moves*. An *action* has been defined as the sequence of movements to perform a pick-and-place task (going over a square, lowering the gripper, picking the piece, moving to the target location, and placing the piece), while one *move*, can be a composition of two *actions*, e.g., in captures or castle moves, and it always ends with the click of the clock. During the recreations of the games, we evaluated:

- Manipulation outcomes;
    - Success in picking;
    - Success in placing;

◦ Success in clicking the clock after the move execution;

- Recognition outcome: success in recognizing the opponent's move.

Lastly, the average time needed to execute a complete chess move and to perform an opponent move recognition, has been evaluated.

8 Results

The executed *moves* during the tests reached a total of 1029, while the total executed *actions* went up to 1303. In total, 133 out of the 1303 executed *actions* presented some errors. To have a more detailed insight of the undesired behaviors, 44 errors happened in picking, 11 in placing, and 13 in clicking the clock. In 65 cases, the gripper touched a piece on the chessboard that was not involved in the requested move. These contacts have been categorized as *hard collision* (30 occurrences) if they required an operator intervention to rearrange the chessboard situation correctly, or *soft collisions* (35 occurrences) if they did not compromise the game situation.
The opponent moves recognition system, was able to correctly track the game situation in the $96w\%$ of the cases.
It should be noticed that not all the undesired behaviors compromise the game progress. Indeed, the *soft collisions* do not require any intervention, and the tracking errors can be corrected by the user through the user interface (as described in Sect. 6.3). Considering this, the application required manual intervention during the test only in 89 of the total 1029 executed *moves*, resulting in a percentage of correct and independent functioning of 91.6%. An overview of the overall performance is reported in Fig. 5.

The percentage of failures over the totality of actions performed over each square is represented in the colormap reported in Fig. 6, where the color of each square gets darker as the frequency of failures in the square increases. To populate the map, the number of *actions* performed by TIAGo's arm over each square has been counted.

In terms of time, the move execution took on average 37.93 ± 11.50 s. The average processing time to recognize the opponent's move settled to 4.88 ± 2.78 s.

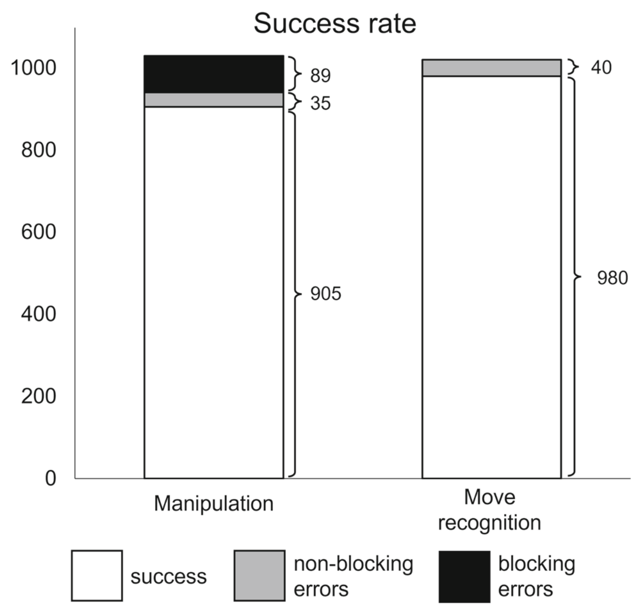
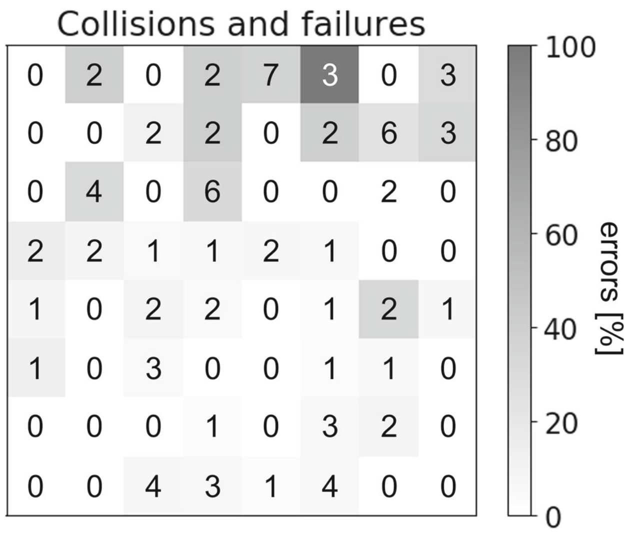


Fig. 5 Barplots reporting the application success rate. These results are relative to the executed *moves*. The sum of the white and grey bars accounts for independent functioning. The black part of the bar represents the errors that require the intervention of an operator



0	2	0	2	7	3	0	3
0	0	2	2	0	2	6	3
0	4	0	6	0	0	2	0
2	2	1	1	2	1	0	0
1	0	2	2	0	1	2	1
1	0	3	0	0	1	1	0
0	0	0	1	0	3	2	0
0	0	4	3	1	4	0	0

Fig. 6 Colormap representing the distribution of the error percentage over the performed *actions* for each square of the chessboard. In each cell, the absolute number of failures is reported

8.1 Pilot Test

In a pilot test, TIAGo assisted a person affected by muscular dystrophy in playing a chess game against a healthy subject with the aim of evaluating the feasibility of the proposed approach when used by a possible target user. The player used a Bluetooth mouse to interact with the screen interface, and the setup of the chess set was managed by the opponent. The game was played without any major inconvenience (i.e., *hard collision*). The player was able to autonomously fix the game situation from the user interface when the robot failed to properly track the opponent's moves. The game was played to the end demonstrating the validity of the proposed approach. The main limitation was identified as screen dependency, seen as a restriction to direct participation. Nevertheless, the user reported being satisfied with the operation of the application and found the experience enjoyable.

9 Discussion

The objective of the work was to implement a robotic assistant to support disabled chess players in competitive games. The robot plays the move on behalf of the player. The possible input devices (joystick, mouse, touchpad…) can be handled by a person with limited arm and hand mobility, and used to select moves from a graphical interface. The opponent's move are tracked by a vision system to update the game situation as seen by the robot and displayed in the user interface.

This work was mainly oriented to guarantee accessibility to a social activity in a competitive environment, with all the specificity of the case. This aspect strongly distinguishes the presented application from other work employing robots in a chess-related environment. Despite this, the accuracy results obtained are compared with similar robotic "chess players", to prove the validity of the implemented system. Del Toro and co-workers [6] used a four-finger gripper specifically designed for the project, but they found the placing accuracy to be strongly affected by low-cost nature of the project and by the simple open-loop mechanical system. As a result, only the 70% of the placing actions (35 out of 50) had a positive outcome. On the other hand, their machine learning algorithm for color classification and movement detection reached an impressive success rate of 99.4%. Chen and Wang [3], using a purely RGB-based (i.e., no depth information) approach to keep track of the game, registered a 100% accuracy in understanding the performed moves. Though impressive, this result may have benefited from the use of a black and white chessboard and chess set, in which the two armies are well-separated in the RGB color space. This condition is not always guaranteed, as materials, colors, and lighting conditions change from game to game. In addition, as for the arm motions, the authors report that *occasionally the arm accidentally knocks over other pieces as it actuates*, requiring human intervention to fix the chessboard situation, though no quantitative measure of this phenomenon is reported.

The main causes of failures are inaccuracies in the vision pipeline, arm replicability error (reported to be 3.5 mm in the worst-case scenario by the robot manufacturer), and failure in planning or executing the movement.
The heatmap in Fig. 6 shows how the most recurrent errors happened on the far side of the chessboard. This highlights how the localization of the squares' centers is more accurate for the squares closer to TIAGo's camera due to the perspective view of the chessboard. Moreover, most of the errors were pick errors, as expected, as this kind of error could be due to cumulative small inaccuracies committed in previous moves, i.e., when TIAGo positions a piece slightly out of the center of the square. If this happens, the robot is likely to fail when the same piece has to be picked again. To reduce the impact of this problem, the computer vision pipeline has been already re-designed, leveraging the insights gathered from the first implementation (superseded, thus not described in this paper). The solution presented significantly increased the robustness against changes in light and chessboard orientation. A further improvement in the segmentation and recognition performance could be achieved by integrating a bird-eye camera into the setup. As for the arm motions, the error introduced by the arm positioning is comparable with the size of the pieces. Nevertheless, the pieces' cylindrical symmetry and the gripper large plates (4 cm) can help mitigate the problem. To make the application more robust against planning and/or execution failures, recovery strategies should be implemented, to enable the robot to try again when an action fails to be completed. As an example, when the planner fails to find a valid arm trajectory, the arm could be randomly moved to a different (safe) configuration. This would provide a new seed to the trajectory planner, possibly leading to the identification of a valid path.

As far as the execution time is concerned, the average move execution time computed during the tests is strongly related to the speed rate chosen to test TIAGo (limited to the 20% of the maximum speed to avoid harmful collisions during the test phase). The standard deviation of the duration of double-action moves is higher than the one for single-action ones because double-action moves can be composed of different arm motion combinations that can take longer time to be completed.
Even with the current execution time, the delay introduced by the robot can be deemed to be acceptable, in the context of long-timed games, typical of competitive tournaments. Nevertheless, such a long time for the robot execution of the move is likely to negatively impact the user experience. Thus, it may represent an obstacle to the actual deployment of the device in the real world. The present limitation can be contrasted by increasing the maximum allowed velocity for the robot motors.

In terms of economic viability of the proposed application, it must be remarked that the PAL Robotics TIAGo robot has been chosen as a test platform for availability reasons. However, the robot embeds sensors and software that have not been exploited for the present application. The proposed application can be deployed on a less expensive setup, made of an RGB-D camera, and mechanical arm with sufficient workspace, connected to the PC for user interface. This would cut the costs related to many unused functionalities of TIAGo, thus making the developed application more realistically implementable in a real chess competition context.

The pilot test proved the capability of the proposed system to assist a chess player. The gathered observations provided useful feedback on the current application, and helped to prioritize the future developmental steps. In particular, the main remarks concerned the user interface. Though the GUI has been appreciated for its usability, the player underlined the importance of having direct visual contact with the chessboard. In this view, a vocal control would enable greater involvement in the game. As for the input device, the user asked to play with the mouse instead of the joystick to speed up (1) the setup phase (i.e., avoiding the time-consuming process of rigidly attaching the joystick to the armrest of the wheelchair), (2) the move selection, moving freely around the chessboard, and not square-by-square. Nevertheless, this solution is only suitable for players with sufficient residual force and mobility.

10 Conclusions and Future Sights

In this work, a PAL Robotics TIAGo robot has been used as a development platform to implement a SAR application aimed at an official chess competition context. Experimental tests resulted in a satisfactory outcome both for the performance of the pick-and-place task and the opponent's moves recognition algorithm. The percentage of moves executed without the need for an operator's intervention reached the 91.6%, with the majority of the errors happening on the side of the chessboard opposite to the robot. Though promising, these results reflect that roughly 1 move out 10 requires operator intervention. The error rate is probably too high to successfully deploy the system. Indeed, the target error rate can be considered to be around the 3–4%, corresponding to at most one error for average-length games.

The opponent's moves recognition algorithm alone reached a total success rate of 96%. Though these results could be further improved, *e.g.* by integrating a bird-eye or an in-hand camera, the vision pipeline is satisfactory in terms of accuracy. In this view, it is worth remembering that these errors can be corrected through the user interface (Fig. 4, label 3).

With the proposed application, whenever the user happens to promote a piece, the opponent is asked to substitute the promoted pawn with the user's desired piece. A possibility to remove this dependency on human intervention is to locate, *e.g.* using ArUco markers, an area outside the chessboard where to place the captured pieces in a predefined configuration. Thanks to this shrewdness, the robot would be able to go back and reach for them whenever the player promotes a pawn and asks for them to replace it.

Regardless of the above, the top priority should be given to the implementation of a vocal control strategy, and its compliance with the official rules. In this way, the need for a monitor to display the GUI would be removed, allowing a more direct involvement of the player in the game, possibly improving the user experience. In this view, also the reduction of the time taken to execute a move should be prioritized, in order to ease the acceptance by the final users.

The presented implementation of a robotic assistant for disabled chess players, helped in the identification of new required features. In particular, the development of recovery strategies in case of arm motion failures, arose as the most impacting ones. In addition, a vocal control system could be integrated as an alternative to the present graphical interface. The user could thus switch to the preferred control interface. For example, the monitor could be removed in casual, non-competitive games in favor of vocal commands. On the other hand, the applicability of vocal control in the context of official tournaments should be investigated. After the implementation of the new required features, the application will be tested with naive and relevant users to gather data about usability and user experience. In conclusion, the process tested in this work is promising, even though further work is needed to improve robustness, execution speed, and naturalness of the interaction to facilitate user acceptance. Ultimately, this work could open the door to the real-world application of robotic helpers in official chess tournaments.

Author Contributions LP, AP, MG conceived the idea and supervised the development process. LP and SA designed the control architecture. LP and SG implemented the control architecture, and conducted the validation tests. LP and MG conceived the testing protocols. LP and SG drafted the manuscript. All authors discussed the results and make significant contribution to the final manuscript. All authors approved the submitted version of the manuscript.

Funding Open access funding provided by Politecnico di Milano within the CRUI-CARE Agreement.

Data Availability The data generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest AP and MG holds shares in AGADE srl (https://agade-exoskeletons.com) and AllyArm srl. The other authors have no competing interests to declare that are relevant to the content of this article.

Consent for Publication Not applicable as the paper does not contain any individual person's data.

References

1. Abdi J, Al-Hindawi A, Ng T et al (2018) Scoping review on the use of socially assistive robot technology in elderly care. BMJ Open. https://doi.org/10.1136/bmjopen-2017-018815
2. Andriella A, Torras C, Abdelnour C et al (2022) Introducing caresser: a framework for in situ learning robot social assistance from expert knowledge and demonstrations. User Model User-Adap Inter. https://doi.org/10.1007/s11257-021-09316-5
3. Chen ATY, Wang KIK (2016) Computer vision based chess playing capabilities for the BAXTER humanoid robot. In: 2016 2nd international conference on control, automation and robotics (ICCAR), pp 11–14. https://doi.org/10.1109/ICCAR.2016.7486689
4. Chess.com (2012) Robots in a moscow park... playing chess. https://www.chess.com/news/view/robots-in-a-moscow-park-playing-chess-video. Accessed 27 Sept 2022
5. Choi J, Rho E, Huh S, et al (2018) An eog/eeg-based hybrid brain-computer interface for chess. In: 2018 IEEE international conference on systems, man, and cybernetics (SMC), pp 129–134. https://doi.org/10.1109/SMC.2018.00033
6. del Toro C, Robles-Algarín C, Rodríguez-Álvarez O (2019) Design and construction of a cost-effective didactic robotic arm for playing chess, using an artificial vision system. Electronics. https://doi.org/10.3390/electronics8101154
7. Feil-Seifer D, Matarić MJ (2005) Defining socially assistive robotics. In: 9th international conference on rehabilitation robotics, 2005 ICORR 2005, pp 465–468
8. FIDE International Chess Federation (2018) FIDE handbook - laws of chess. https://handbook.fide.com/chapter/E012018. Accessed 28 Sept 2022
9. FIDE International Chess Federation (2020) FIDE handbook - handicapped players. https://handbook.fide.com/chapter/E02. Accessed 26 Sept 2022
10. Gandolla M, Dalla Gasperina S, Longatelli V et al (2021) An assistive upper-limb exoskeleton controlled by multi-modal interfaces for severely impaired patients: development and experimental assessment. Robot Auton Syst 143:103822. https://doi.org/10.1016/j.robot.2021.103822
11. Kutlu Y, Karaca G (2018) Brain computer interface chess game platform. In: Natural and engineering sciences, pp 1–11. Supplement I of symposium articles
12. Langer A, Feingold-Polak R, Mueller O et al (2019) Trust in socially assistive robots: considerations for use in rehabilitation. Neurosci Biobehav Rev 104:231–239. https://doi.org/10.1016/j.neubiorev.2019.07.014
13. Lotfi A, Langensiepen C, Yahaya SW (2018) Socially assistive robotics: robot exercise trainer for older adults. Technologies 6(1):32
14. Lukač D (2018) Playing chess with the assistance of an industrial robot. In: 2018 3rd international conference on control and robotics engineering (ICCRE), pp 1–5. https://doi.org/10.1109/ICCRE.2018.8376423
15. Martinez-Martin E, Escalona F, Cazorla M (2020) Socially assistive robots for older adults and people with autism: an overview. Electronics 9(2):367
16. Maruthappan N, Iyengar N, Sudip P, et al (2011) Brain chess—playing chess using brain computer interface
17. Matuszek C, Mayton B, Aimi R, et al (2011) Gambit: an autonomous chess-playing robotic system. In: 2011 IEEE international conference on robotics and automation, pp 4291–4297. https://doi.org/10.1109/ICRA.2011.5980528
18. Nario-Redmond MR, Noel JG, Fern E (2013) Redefining disability, re-imagining the self: disability identification predicts self-esteem and strategic responses to stigma. Self Identity 12(5):468–488. https://doi.org/10.1080/15298868.2012.681118
19. Ómarsdóttir FY, Ólafsson RB, Foley JT (2016) The axiomatic design of chessmate: a chess-playing robot. Procedia CIRP 53:231–236. https://doi.org/10.1016/j.procir.2016.07.002
20. PC World (2013) A brief history of computer chess. https://www.pcworld.com/article/451599/a-brief-history-of-computer-chess.html. Accessed 01 Feb 2022
21. Piasek J, Wieczorowska-Tobis K (2018) Acceptance and long-term use of a social robot by elderly users in a domestic environment. In: 2018 11th international conference on human system interaction (HSI), pp 478–482. https://doi.org/10.1109/HSI.2018.8431348
22. Pozzi L, Gandolla M, Pura F et al (2022) Grasping learning, optimization, and knowledge transfer in the robotics field. Sci Rep. https://doi.org/10.1038/s41598-022-08276-z
23. Rabbitt SM, Kazdin AE, Scassellati B (2015) Integrating socially assistive robotics into mental healthcare interventions: applications and recommendations for expanded use. Clin. Psychol. Rev. 35:35–46. https://doi.org/10.1016/j.cpr.2014.07.001
24. Rath PK, Mahapatro N, Nath P, et al (2019) Autonomous chess playing robot. In: 2019 28th IEEE international conference on robot and human interactive communication (RO-MAN), pp 1–6. https://doi.org/10.1109/RO-MAN46459.2019.8956389
25. Short E, Swift-Spong K, Greczek J, et al (2014) How to train your DragonBot: socially assistive robots for teaching children about nutrition through play. In: The 23rd IEEE international symposium on robot and human interactive communication, pp 924–929. https://doi.org/10.1109/ROMAN.2014.6926371
26. Wang Z, Bovik A, Sheikh H et al (2004) Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 13(4):600–612. https://doi.org/10.1109/TIP.2003.819861

Luca Pozzi is PhD candidate in Mechanical Engineering at Politecnico di Milano. He graduated in Biomedical Engineering in 2021, with a thesis on robotic manipulation. He then joined the WE-COBOT lab (Wearable and Collaborative Robotics Laboratory, Politecnico di Milano, Polo Territoriale di Lecco) as a research fellow, working on the recognition of human pointing gestures for human-robot interaction. In November 2022 he started his PhD at the WE-COBOT lab with the objective of developing service robotics applications to support elderly people in long-term care facilities.

Silvia Guerini is currently Software Developer at DazeTechnology Srl (Almenno San Bartolomeo, Bergamo, Italy). She contributed to the work presented in the paper during her M.Sc. thesis project at WE-COBOT lab (Politecnico di Milano, Polo Territoriale di Lecco). After graduating cum laude at Politecnico di Milano in Biomedical Engineering in 2022, Silvia worked as a research fellow at IRCCS Medea on gait analysis for cognitive neuroscience studies. She is now a firmware developer in the R&D department of Daze, working on EV charging technologies.

Stefano Arrigoni has been Assistant Professor in the Department of Mechanical Engineering since 2019. He graduated cum laude in 2013 and obtained the title of PhD in Mechanical Engineering in 2017 working on MPC-based algorithms and applications of genetic algorithms for optimal control and autonomous driving. He later became a research fellow for the development of HiL techniques related to vehicle ABS systems. His main research activities are the development of control logic for autonomous driving, the development of connected ADAS, the development of the logic of traffic flow management through connected systems and infrastructure, and the development of bio-inspired robots and motion control techniques.

Alessandra Pedrocchi is professor at the Department of Electronics, Computer Science, and Bioengineering of the Politecnico di Milano, where she teaches Neuroengineering and Biomedical Instrumentation in the course of studies in Biomedical Engineering. She received her degree in electronic engineering and her doctorate in bioengineering at the Politecnico di Milano in 1997 and 2001. She is in charge of the neuroengineering section of the Nearlab laboratory, the Neuro-Engineering And Medical Robotics Lab, and of the interdipartimental laboratory WE-COBOT lab (Politecnico di Milano, Polo Territoriale di Lecco). Her research field is neuroengineering, including biomechanics in motor control, neurorobotics, new technologies for neurorehabilitation, with particular emphasis on the control systems and Human-robot interfaces of upper and lower limb exoskeletons for rehabilitation and assistive devices, neuroprostheses, and the study of the correlation between brain plasticity and functional recovery.

Marta Gandolla is currently Assistant Professor at the Department of Mechanical Engineering at Politecnico di Milano. She graduated in Biomedical Engineering in 2009 and received a European PhD cum laude in Bioengineering in 2013 from Politecnico di Milano. In 2011, she was a visiting PhD student at the Sobell Department of Motor Neuroscience of the UCL Institute of Neurology (London, UK), under the supervision of Dr. N. Ward. From 2013, she was a research fellow at the NEARLab (Politecnico di Milano), with which she is currently collaborating with interdisciplinary projects at the WE-COBOT lab (Politecnico di Milano, Polo Territoriale di Lecco). In 2016, she was a visiting fellow at Fondazione Don Carlo Gnocchi (Centro S. Maria della Provvidenza, Roma), under the supervision of Prof Luca Padua. Her scientific research investigates the field of neurological motor rehabilitation and motor assistance during daily life activities of fragile people or during exhausting activities of workers.