Delft University of Technology

An open-source reproducible chess robot for human-robot interaction research

Zhang, R.; de Winter, J.C.F.; Dodou, D.; Seyffert, Harleigh C. ; Eisma, Y.B.

DOI
10.3389/frobt.2025.1436674

Publication date
2025

Document Version
Final published version

Published in
Frontiers In Robotics and AI

Citation (APA)
Zhang, R., de Winter, J. C. F., Dodou, D., Seyffert, H. C., & Eisma, Y. B. (2025). An open-source reproducible chess robot for human-robot interaction research. *Frontiers In Robotics and AI*, *12*, Article 1436674. https://doi.org/10.3389/frobt.2025.1436674

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Frontiers in Robotics and AI

TYPE Original Research
PUBLISHED 16 April 2025
DOI 10.3389/frobt.2025.1436674

Check for updates

OPEN ACCESS

EDITED BY
Alan R. Wagner,
The Pennsylvania State University (PSU),
United States

REVIEWED BY
Pitoyo Hartono,
Chukyo University, Japan
Shu Li,
Liaoning University of Technology, China

*CORRESPONDENCE
Joost de Winter,
✉ j.c.f.dewinter@tudelft.nl

RECEIVED 22 May 2024
ACCEPTED 28 February 2025
PUBLISHED 16 April 2025

CITATION
Zhang R, de Winter J, Dodou D, Seyffert H and Eisma YB (2025) An open-source reproducible chess robot for human-robot interaction research.
Front. Robot. AI 12:1436674.
doi: 10.3389/frobt.2025.1436674

COPYRIGHT
© 2025 Zhang, de Winter, Dodou, Seyffert and Eisma. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.

An open-source reproducible chess robot for human-robot interaction research

Renchi Zhang, Joost de Winter*, Dimitra Dodou,
Harleigh Seyffert and Yke Bauke Eisma

Faculty of Mechanical Engineering, Delft University of Technology, Delft, Netherlands

Recent advancements in AI have accelerated the evolution of versatile robot designs. Chess provides a standardized environment for evaluating the impact of robot behavior on human behavior. This article presents an open-source chess robot for human-robot interaction research, specifically focusing on verbal and non-verbal interactions. The OpenChessRobot recognizes chess pieces using computer vision, executes moves, and interacts with the human player through voice and robotic gestures. We detail the software design, provide quantitative evaluations of the efficacy of the robot, and offer a guide for its reproducibility. An online survey examining people's views of the robot in three possible scenarios was conducted with 597 participants. The robot received the highest ratings in the robotics education and the chess coach scenarios, while the home entertainment scenario received the lowest scores. The code is accessible on GitHub: https://github.com/renchizhhhh/OpenChessRobot.

KEYWORDS

artificial intelligence, chess, human-robot interaction, open-source, user studies

1 Introduction

Robots are becoming increasingly common across a variety of traditionally human-controlled domains. Examples range from automated mowers that maintain community lawns to robots on assembly lines and in agricultural settings. Recent advancements in AI have created new opportunities for intelligent sensing, reasoning, and acting by robots. In particular, the rapid development of large language models, such as ChatGPT and vision-language models, has lowered the barrier to human-robot communication by transforming text and images into interpretable actions or *vice versa*.

As technology advances, robots will likely attain greater capabilities and be able to tackle tasks previously within the exclusive realm of human expertise. This ongoing evolution may also lead to closer and more productive interactions between humans and robots. At the same time, integrating different AI-based robotic components remains a challenge, and the human-robot interaction (HRI) field lags in terms of endorsing reproducibility principles (Gunes et al., 2022). Encouraging transparent and reproducible research, therefore, remains an ongoing task.

The use of chess as a testbed for evaluating the effect of technology on human perception and behavior dates back to the 18th-century chess automaton Mechanical Turk (Standage, 2002). Furthermore, chess has played an important role in advancing the field of AI, starting with Claude Shannon's chess-playing algorithm (Shannon, 1950) to the success of IBM's Deep Blue (Campbell et al., 2002) and DeepMind's self-play learning algorithm (Silver et al., 2018). In this paper, we incorporate AI algorithms into the design of a chess-playing robot to be used for studying HRI. HRI research may benefit from a chess-based setup because the game of chess provides a controlled, rule-based environment in which the impact of robots on human players can be precisely measured.

HRI-oriented studies with chess robots have typically used them to imitate human behaviors and assess their resulting impact on the human opponent. Pereira et al. (2008) and Leite et al. (2013) used the social robot iCat to play chess with children. This robot relies on an electronic chessboard as input and emits emotional responses and verbal utterances, guided by an emotion model. Sajó et al. (2011) developed Turk-2, a multimodal chess robot with human-like communication skills, while LC et al. (2021) explored human-robot and robot-robot-human interaction using artistic intervention, where expressive robot arms played chess and embodied distinct personalities.

When an electronic chessboard is not used, a camera is needed to determine if a move has been made and, if so, which move it was. A common solution is a monocular top-view camera, which has the advantage of not experiencing perspective-induced occlusion, making it relatively easy to identify any changes in the chess position (Golz and Biesenbach, 2015; Kołosowski et al., 2020; Larregay et al., 2018; Luqman and Zaffar, 2016; Mac et al., 2023; Srivatsan et al., 2020; Thanh Tra et al., 2018). An exception to this concept is the Gambit chess robot by Matuszek et al. (2011), which does not require a top-down view but uses a stereo camera instead. This robot is unique because it classifies the pieces by type and color, whereas other robots identify and track move-by-move changes from the starting position (e.g., Golz and Biesenbach, 2015; Kołosowski et al., 2020; Larregay et al., 2018; Siraj, 2017).

Other research focuses on the development of computer-vision methods for the detection of the chessboard, and the subsequent classification of the pieces on it, without always linking these computer-vision methods to a chess robot (Christie et al., 2017; Czyżewski et al., 2020; Ding, 2016; Koray and Sümer, 2016; Ranasinghe et al., 2023; Schwenk and Yuan, 2015; Xie et al., 2018). The detection of the chessboard typically involves line- or edge-detection techniques (Chen and Wang, 2019; Czyżewski et al., 2020; Srivatsan et al., 2020; Wölflein and Arandjelović, 2021; Xie et al., 2018), while the classification of chess pieces commonly uses convolutional neural networks (CNN) (Mallasén Quintana et al., 2020; Shin et al., 2023; Wölflein and Arandjelović, 2021). Challenges in both cases lie in testing for robustness under various conditions. A common technique to improve classification performance is to verify if the move/position is legal or plausible according to a chess engine. With such methods, it is possible to let the algorithm home in on the most probable classification outcome (e.g., Czyżewski et al., 2020; Mallasén Quintana et al., 2020; Ranasinghe et al., 2023). Currently, a state-of-the-art method is that of Wölflein and Arandjelović (2021), which used a diverse set of neural network models and achieved a per-square classification accuracy of 99.8%.

Our contribution centers on a software platform that others can reproduce using open-source robotics middleware. The chosen robot arm is widely available in research, while other hardware components can be substituted if required. The OpenChessRobot can detect pieces using computer vision and execute moves. For verbal interaction, we rely on large language models. Specifically, by connecting ChatGPT to the chess engine output, our robot interprets strategies and explains them to humans in natural language. Non-verbally, the robot is able to express its evaluation of the game by either nodding or shaking its end-effector. Supplementary Material Part A provides a comparison of the OpenChessRobot with other chess robots designed for HRI studies.

This paper first presents the hardware and software design of the chess robot. It then provides an outline of its core modules, which include Perception, Analysis and Evaluation, Motion Planning and Execution, and Interaction. Module-specific experiments for performance evaluation are presented as well. Next, the paper reports findings from two human evaluations: an interview with an expert chess player who played two games with the robot, and an online study of 597 participants, which investigated perceptions of a talking robot equipped with AI in three different scenarios. Finally, the paper concludes with a discussion of the robot's limitations and outlines directions for future research in HRI.

2 Hardware and software architecture

Our chess-playing robot (Figure 1A) consists of the following hardware components (Figure 1B):

- A Franka Emika Panda robot arm (Franka Emika, 2020) equipped with a Franka Hand and a customized 3D-printed robot gripper. The robot arm has 7 degrees of freedom and is a widely used collaborative robot, suited for safe human interaction. It operates on firmware version 5.4.0
- A ZED2 StereoLabs camera (StereoLabs, 2020)
- A keyboard
- An external microphone and a speaker, connected to the PC
- An NVIDIA Jetson Nano (NVIDIA, 2019)
- A Linux PC with Ubuntu 20.04, running a real-time kernel. It is equipped with an Intel I7-8700K processor and an NVIDIA RTX 2080 graphics card.

Our setup includes a number of accessories. These comprise a chessboard (48 × 48 cm), a set of chess pieces (model: Staunton No. 4; ranging in height from 4 to 7.8 cm and in width from 2 to 4 cm), and four printed ArUco markers. Considering the robot arm's range, the chessboard should be positioned so that the farthest rank remains within 65 cm of the robot. In Figure 1B, the distance between the closest edge of the board and the robot is 16 cm.

The robot arm, the Jetson Nano, and the Linux PC connect to a local network via a router. Communication between the arm and the control PC is realized through the Franka Control Interface (Franka Robotics GmbH, 2023) by integrating the Libfranka (0.9.2) library for low-level control and the franka_ros package for the Robot Operating System (ROS) (Quigley et al., 2009). The ZED2 camera is attached to the Franka Hand via a custom mount, and it interfaces with the NVIDIA Jetson Nano through a USB connection. The Jetson Nano streams the camera view to the PC through ZED SDK v3.5. Considering that our setup uses images from a single camera, the ZED2 camera and Jetson Nano can be substituted with more economical monocular alternatives.

The software architecture of the OpenChessRobot is built upon ROS Noetic (Open Robotics, 2020), which offers

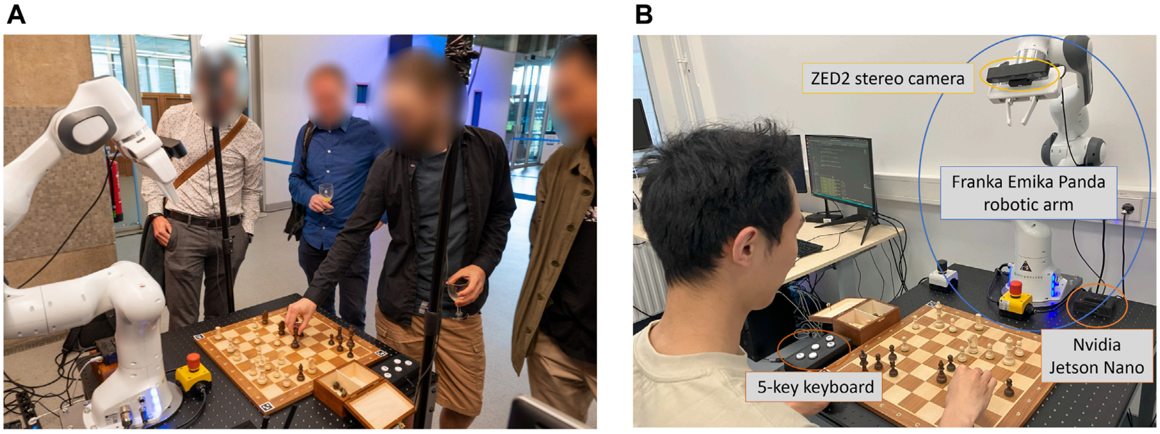

FIGURE 1
The system at a glance. **(A)** The robot at a demonstration event during which attendees could interact with it. **(B)** The complete system, comprising a Franka Emika Panda robotic arm, a ZED2 stereo camera, and an NVIDIA Jetson Nano computing board.

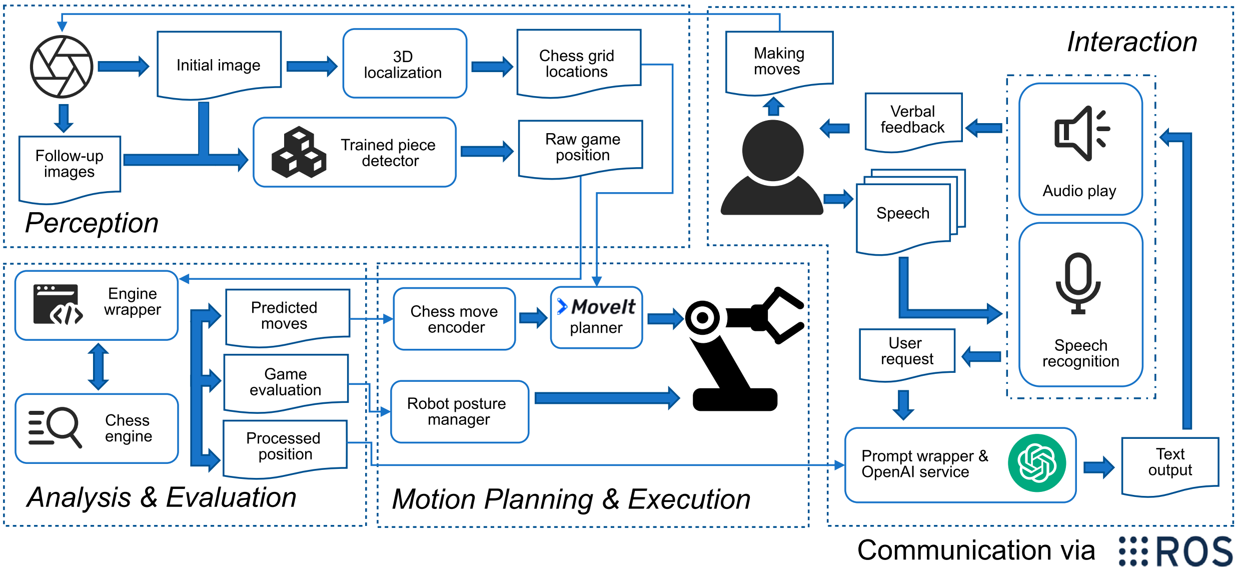

FIGURE 2
The software architecture of four modules and inter-module communication. Within the Perception module, the camera images are used to determine the 3D positions of the squares on the chessboard and identify individual pieces. The identified game position is analyzed through the chess engine wrapper, yielding the best next move and an evaluation of the current board position. The OpenChessRobot then enacts the chess move. Additionally, the robot is capable of responding through posture and speech.

common robotics data formats and message-passing among software modules. Figure 2 provides an overview of the software architecture, divided into four modules: Perception, Analysis and Evaluation, Motion Planning and Execution, and Interaction.

The Perception module uses the ZED2 camera SDK to capture the chess game images and relies on a neural network-based chess identifier to translate the game images into text descriptions. The latter identifier requires PyTorch and CUDA runtime. The Analysis and Evaluation module feeds the chess game annotation to a chess engine in order to get predicted moves and their corresponding scores. In the Motion Planning and Execution module, the predicted move, accompanied by the 3D chessboard localization results from the Perception module, is used by the MoveIt! motion planner (Coleman et al., 2014) to plan and execute chess moves.

The Interaction module takes the human inputs and manages the OpenChessRobot's verbal and non-verbal feedback. For the verbal feedback, a prompt wrapper, which combines the user request and outputs from the Analysis and Evaluation module, serves as a client of OpenAI's ChatGPT API service and generates the responses to talk with the human.

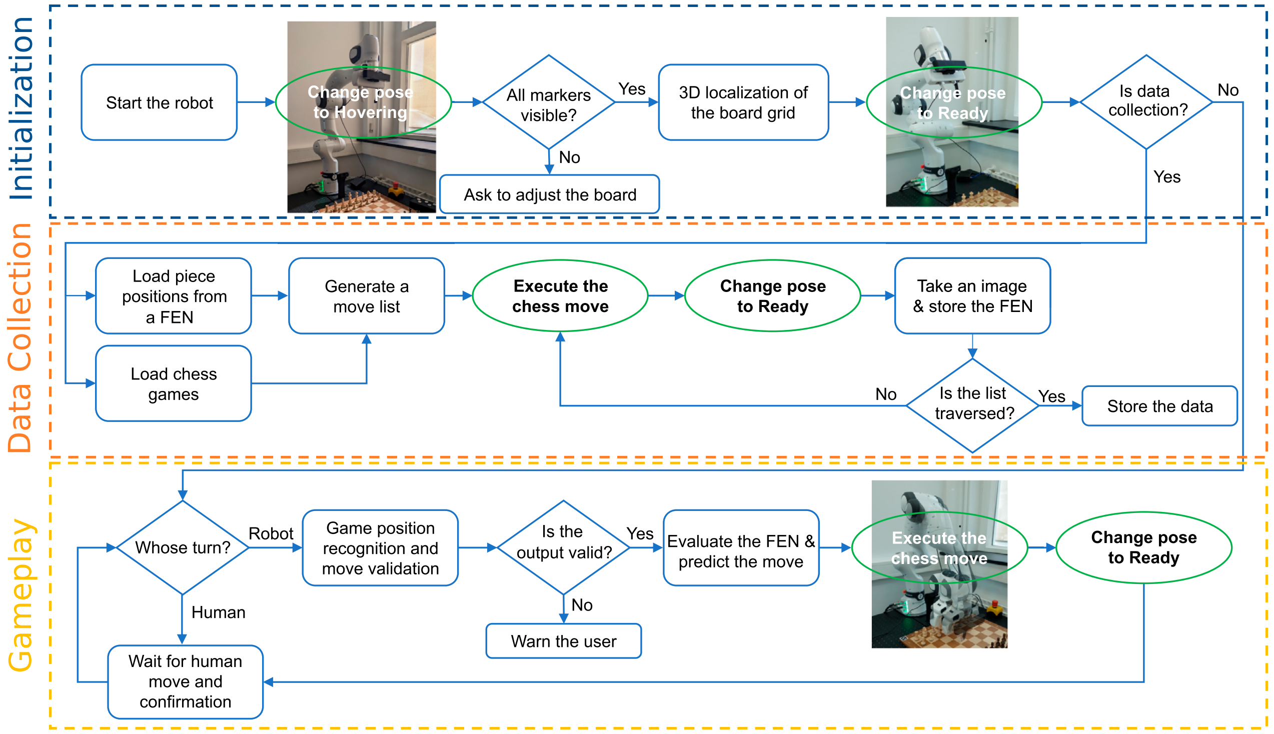

FIGURE 3
Flowchart of the OpenChessRobot execution using the core modules. The robot uses computer vision to recognize the chessboard and pieces, engages with a chess engine to determine moves, and executes chess moves for gameplay or data gathering. The chart splits into two key workflows: one for collecting data to improve the robot's perception of different chessboards and chess pieces, and another for playing chess with a human.

3 Modules

The Perception module uses computer vision to identify the chessboard and the pieces; it can distinguish between occupied and empty squares and recognize pieces and their color. The perceived game position is sent to the Analysis and Evaluation module, which interfaces with a chess engine to determine the best move. The Perception module also calculates the chess squares' real-world locations, which are sent to the motion planner for planning the robot's end-effector trajectory to execute chess moves provided by the Analysis and Evaluation module.

The OpenChessRobot runs a data collection pipeline (orange box in Figure 3) and a human-robot gameplay pipeline (yellow box in Figure 3). The data collection pipeline is used to collect real-world data of chess pieces for retraining the Perception module and adapting to a new chessboard. The basic gameplay pipeline allows the robot to play the chess game with a human from any game position.

3.1 Perception

The ability to perceive the chess pieces is a key function of the chess robot. While the commonly used top-down view resolves the issue of piece occlusion, its practicality is limited by its narrow perspective on the pieces. Adopting lower camera angles and dynamic camera positions to observe the game supports the robot's natural interaction with the human player (see Figure 1). Additionally, our approach uses RGB images captured by a single camera instead of point clouds (Matuszek et al., 2011), improving the simplicity and reproducibility of our setup.

Figure 4 shows the Perception module of the OpenChessRobot. The module consists of two distinct classifiers, one for occupancy and the other for piece classification.

3.1.1 3D chessboard localization

To manipulate the chess pieces, their 3D coordinates in the real world need to be acquired. Four ArUco markers are used to indicate the 3D position of the chessboard. An image capturing these markers, along with the marker length and the camera's intrinsic and extrinsic parameters and distortion coefficients, allows for the computation of the marker-to-camera translation and rotation using OpenCV.

The 3D board localization is activated when the OpenChessRobot is initialized. The robot moves to its Hovering pose to observe the chessboard from above (blue box in Figure 3). Although the robot typically operates with low camera positions during the game, a vertical camera angle is used for 3D chessboard localization to reduce errors caused by image distortion. Then, a grid corresponding to the chess squares is formed based on the actual square size, and a non-linear least squares algorithm is used to optimize the grid corner positions to closely match the marker positions.

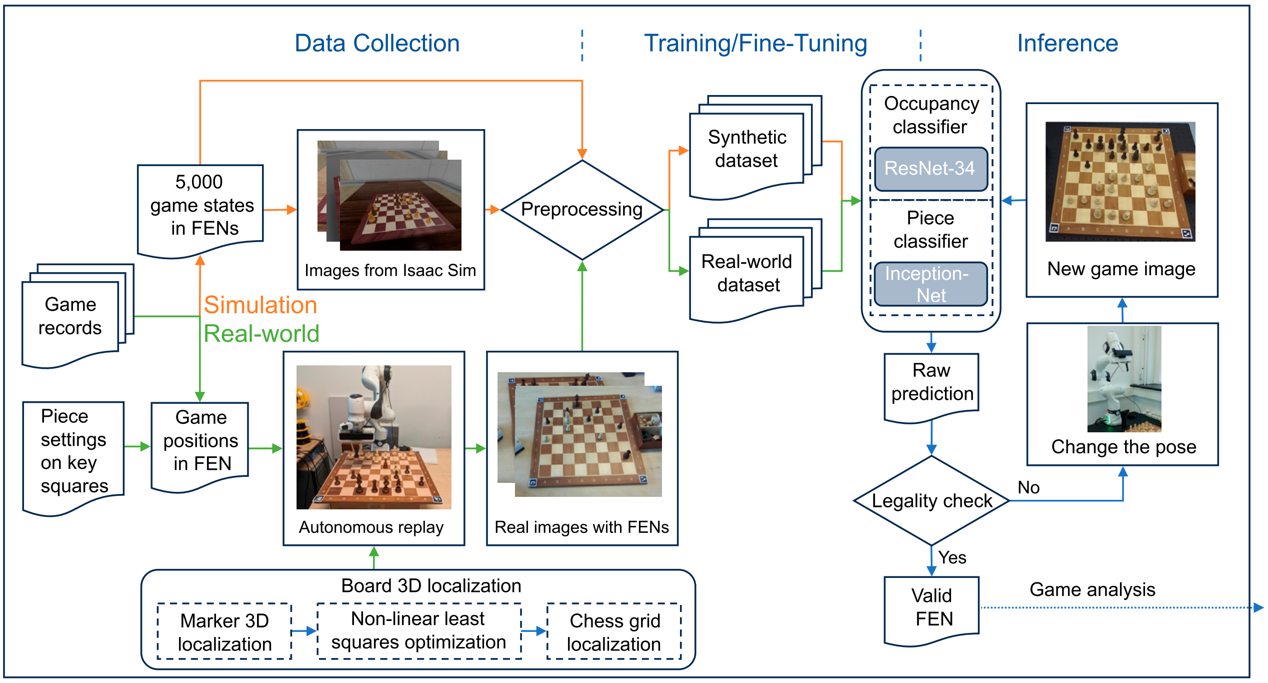

FIGURE 4
The Perception module of the OpenChessRobot relies on two distinct classifiers, one for occupancy and the other for piece classification. To train these classifiers, we synthesized chess positions in the NVIDIA Isaac Simulator (Makoviychuk et al., 2021) based on previous chess games and assigned ground truth labels, including the game positions represented in Forsyth–Edwards Notation (FEN) and pixel coordinates of board corners. In the real-world setup, the robot collects game images from the predefined camera pose (Figure 3) after autonomously placing pieces on key squares and replaying given games. These real-world images are then used to fine-tune the classifiers that were initially trained on synthetic datasets.

3.1.2 Chess piece detection

In the preceding section, we explained how the 3D coordinates of the squares are estimated. The next task is to identify the chess pieces on each square.

The piece detection model is an extension of a CNN-based model from Wölflein and Arandjelović (2021). We complemented their method by using a different synthetic dataset as well as a real-world dataset to fine-tune the pretrained models in order to effectively handle our real-world chess set.

To synthesize images of chess pieces on a board, we varied backgrounds, lighting conditions, camera poses, and piece locations. A total of 5,000 game positions, randomly selected from games played by grandmaster Bobby Fischer, were used. We synthesized images of selected chess games using the NVIDIA Isaac Simulator (Makoviychuk et al., 2021). Ground truth labels were created, consisting of the game positions represented in Forsyth–Edwards Notation (FEN) and the positions of the board corners. Following a similar approach to previous works (Mallasén Quintana et al., 2020; Matuszek et al., 2011; Wölflein and Arandjelović, 2021), the images were cropped into sub-images, each corresponding to a square on the chessboard. Two categories, "empty" or "occupied" were assigned for occupancy classification, while 12 categories (representing the six types of chess pieces in both black and white) were designated for piece classification.

To train the occupancy and piece classifiers with the synthetic data, 80% of the cropped images were used as the training set, while the remaining 20% was evenly split between a test set and a validation set. A ResNet-34 (He et al., 2016) and an InceptionV3 (Szegedy et al., 2016) were trained independently on their respective training datasets. This is illustrated in the left portion of Figure 4.

To adapt to real-world chess, a piece-square dataset was created by manually iterating over all the chess pieces on the board. During the iteration, the game positions were recorded both in FEN notation and in images taken at 1080p resolution from the perspective of the robot in the Ready pose (Figure 3). A number of training datasets to fine-tune the piece classifier were extracted from the piece-square dataset using sub-images of chess pieces on key squares. The trained models were evaluated on the remaining portion of the piece-square dataset. Additionally, to evaluate the classifiers' performance in games, a game dataset was collected by the robot autonomously replaying the provided chess games. Two games played by the grandmaster Michael Adams were used, with the game positions recorded both in FEN and in images, following the same procedure used when collecting the piece-square set. Dataset creation, training settings, and evaluation are explained in more detail in Section 5.1 and Supplementary Material Part B.

The preceding chess detection model considers only the position of the chess game. It does not verify the legality of the observed game position and does not possess knowledge of the previous game position. Such information can potentially improve the accuracy of the Perception module. Our legality check consists of two steps: first, it verifies if the game position conflicts with the chess rules, such as detecting no king or too many pawns. If the game position is invalid, it requests a new image (using the same camera pose but with potentially better camera exposure) and reanalyzes it. The robot can also change the camera angle to improve perception. More specifically, in case the predicted game states are detected as illegal, the robot slightly moves to its side in order to capture an image from a different perspective to improve the occupancy and piece classification.

In the second step of the legality check, the robot examines if the recognized chess move is among the legal moves from the previous game position. When the inferred game position and chess move prove to be legitimate, the robot adopts the inference and proceeds with gameplay. If not, the robot will halt its operation and wait for manual correction by the operator.

3.2 Analysis and evaluation

The processed chess FEN is forwarded to a chess engine wrapper, which uses the Universal Chess Interface (UCI) protocol for interaction with standard engines. Our chess engine wrapper builds upon an existing open-source Python wrapper for Stockfish (Zhelyabuzhsky, 2022). Our system integrates Stockfish 15 (Stockfish, 2022) as the default engine, using a total of 10 CPU threads. The chess engine assigns scores to candidate moves, which are monotonically related to the player's win rate, should the engine play against an equally strong opponent.

3.3 Motion planning and execution

For the planning of the OpenChessRobot's motions, we rely on the MoveIt! motion planner (Coleman et al., 2014). To determine the joint configurations, an inverse kinematics solver named IKFast is used (MoveIt, 2021). This solver calculates suitable robot joint angles based on the specified end-effector coordinates relative to the robot's base.

The robot arm is equipped with a customized gripper (Figure 5), designed to secure the underside of the piece, which is cylindrical in shape. The gripper is designed to be tolerant to minor deviations between the anticipated pickup positions (square centers) and the actual positions. Consequently, the human (or robotic) chess player is not bound to place a piece exactly at the center of a square.

There are two basic robot motions: a sliding motion (Figure 6A) and a jumping motion (Figure 6B). If the path between the start and end positions of a movement is unobstructed, the robot grasps the piece and pushes it to the destination square. When the path is obstructed, the robot raises the piece to leap over other pieces. Special movements like capture, castling, and en passant are manually programmed as combinations of the two basic motions.

A chess move for the robot to execute is provided by the Analysis and Evaluation module in an encoded text format, e.g., g1f310000. The first four characters indicate the chess move. Additional information for a move is stored in the remaining characters. Specifically, the fifth to ninth characters indicate whether the move is a jump, capture, castling, en passant, or promotion move.

With the encoded string, we create waypoints, using the 3D locations of the start square, the destination square, as well as intermediate states for special motions. The intermediate states are generated when the robot needs to lift pieces first rather than slide them on the board. The motion planner receives these waypoints and plans the trajectory of the end-effector using Rapidly-exploring Random Trees (RRTConnect; Kuffner and LaValle, 2000), with a 3D box surrounding the chessboard as a workspace constraint. After trajectory planning, velocity and acceleration limits are enforced on the planned trajectory (Kunz and Stilman, 2013).

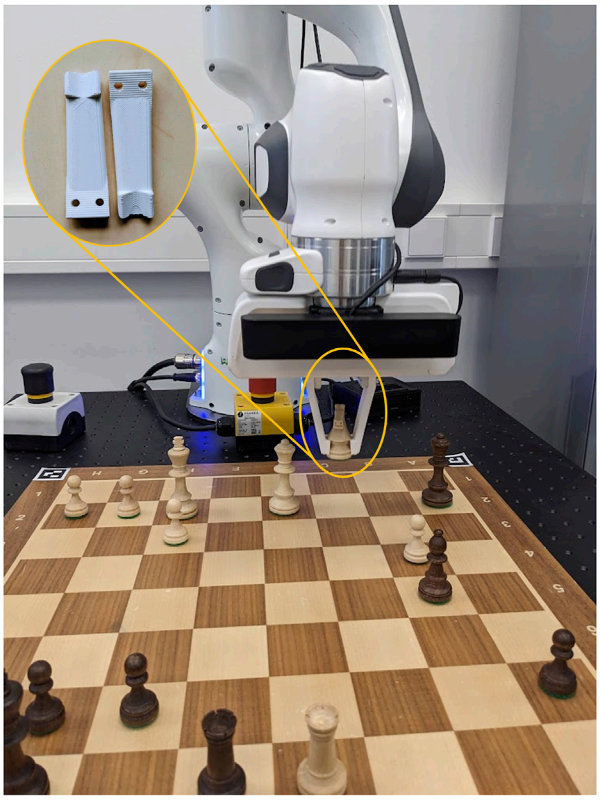

FIGURE 5
The customized 3D-printed gripper. This gripper has two fingers that are tolerant to deviations of chess pieces if they are not in the center of a square.

4 Interactive modules

As the chess robot is designed for HRI research, providing various modalities for human-robot communication is important. We implemented an interactive gameplay pipeline based on the core modules (Figure 7). Specifically, depending on the human's chess performance, the OpenChessRobot can move its end-effector to express its evaluation, and it can provide verbal information regarding move qualities.

4.1 Verbal interaction

In typical chess games, verbal communication does not hold a central role, although some professional players have explained the rationale and inner thoughts of chess moves in “Banter Blitz”. Additionally, chess trainers commonly use verbal feedback to train

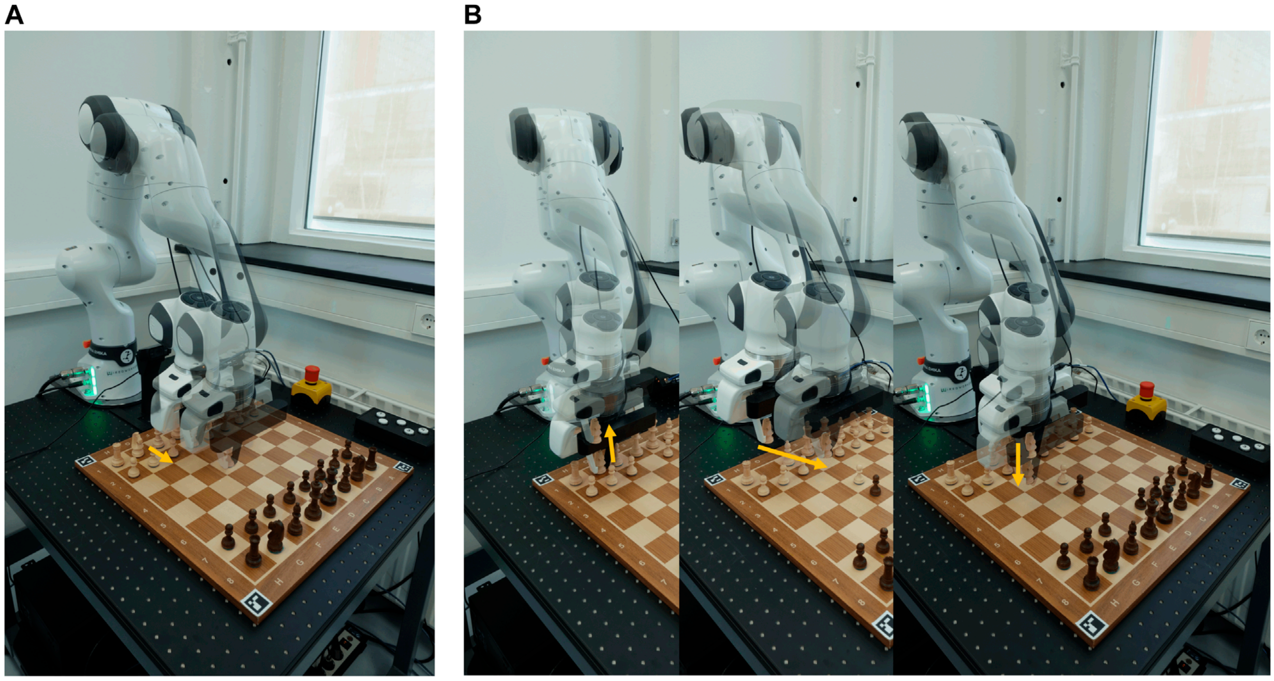

FIGURE 6
The robot is executing planned trajectories to move chess pieces. **(A)** The robot slides a pawn from the E2 square to the E4 square. **(B)** The robot lifts the knight on G1 to jump over other pieces on its way.

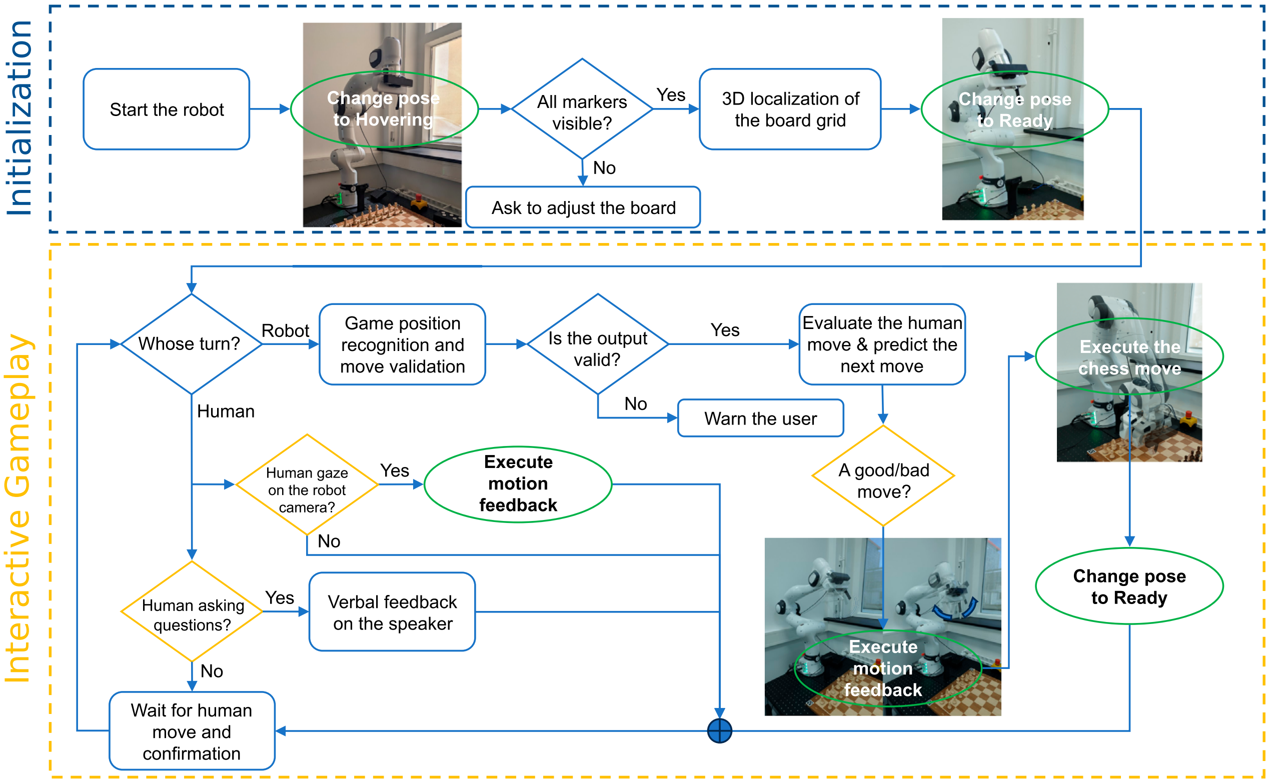

FIGURE 7
Flowchart of the OpenChessRobot execution using the core modules and Interactive module. As an extension of the core modules, the robot has an interactive gameplay pipeline. It evaluates human moves using predefined criteria and provides feedback by adjusting its posture. The system is also programmed to interpret human behaviors, such as gazing or asking questions, to provide a responsive and engaging experience.

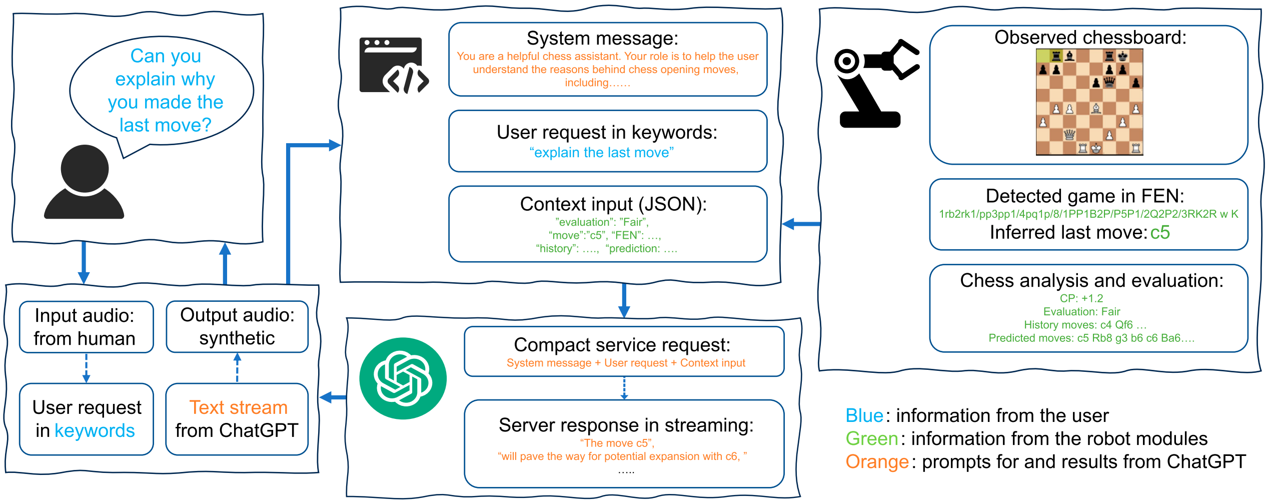
Can you explain why you made the last move?
Input audio: from human
Output audio: synthetic
User request in keywords
Text stream from ChatGPT
System message:
You are a helpful chess assistant. Your role is to help the user understand the reasons behind chess opening moves, including……
User request in keywords:
"explain the last move"
Context input (JSON):
"evaluation": "Fair",
"move":"c5", "FEN": …,
"history": …., "prediction: ….
Compact service request:
System message + User request + Context input
Server response in streaming:
"The move c5",
"will pave the way for potential expansion with c6, "
…..
Observed chessboard:
Detected game in FEN:
1rb2rk1/pp3pp1/4pq1p/8/1PP1B2P/P5P1/2Q2P2/3RK2R w K
Inferred last move: c5
Chess analysis and evaluation:
CP: +1.2
Evaluation: Fair
History moves: c4 Qf6 …
Predicted moves: c5 Rb8 g3 b6 c6 Ba6….
Blue: information from the user
Green: information from the robot modules
Orange: prompts for and results from ChatGPT

FIGURE 8
Verbal interaction. In this diagram, the user asks for an explanation of the last move. Next, a prompt consisting of a system message, the user request, and the processed game position is provided to the ChatGPT API to generate verbal feedback.

their students. OpenAI's ChatGPT has proven capable of generating human-like dialogues. However, ChatGPT is unable to properly play chess on its own (Kuo et al., 2023). Our proposed solution relies on combining the analysis from the chess engine with ChatGPT without model fine-tuning.

Depending on the human player's request, the OpenChessRobot can explain the last move or the next move. The process is depicted in Figure 8. Capturing the user's request is achieved through voice recognition. To trigger the voice interaction, the robot should be in the Ready pose, waiting for the user's move. The player must vocalize specific keywords of the phrases, such as "explain the (last) move" or "analyze/predict the (next) move".

When explaining the last move, the move history and the current game FEN are sent to the Analysis and Evaluation module. Alternatively, when the user seeks advice on subsequent moves, the system uses the chess engine to predict several top-ranked moves, along with their game continuations. A wrapper integrates global instructions and the chess game information into a prompt ready for submission to the ChatGPT API (using model GPT-4-0613). The global instructions, which remain consistent across requests, are embedded in a system message (Table 1).

In response to the prompt, the ChatGPT API is requested to provide an analysis within the specified chess context in a tutor-like tone. To reduce the latency in vocalizing text responses from the server while maintaining a natural flow in synthetic voice, the feedback from the server, streaming in words/characters, is organized into short sentences, which are stored in a queue and played sequentially.

4.2 Non-verbal interaction

In human-human communication, non-verbal cues play an important role. In professional chess games, verbal exchanges are often sparse. Apart from the moves played, participants may rely on behavioral cues to understand their opponents.

The posture feedback consists of basic nodding and shaking motions executed by the robot's hand (Figure 9). These gestures are triggered based on the chess engine's evaluation. Specifically, the robot enacts nodding or shaking gestures when the reduction in win probability due to the latest move exceeds a predefined threshold.

5 Technical evaluation

We performed the following assessments of the chess robot system: (1) Perception module: accuracies of the piece classifiers; (2) Motion Planning and Execution module: success rates of grasping a piece from different sections of each square.

5.1 Chess perception

For the piece classifier of the chess perception module, several training settings were tested, using two sets of key squares, different components of the training dataset, and varying training lengths. The challenge was to identify a training setting that minimizes the number of training samples while achieving good generalization performance.

The two sets of key squares were:

- 3×3 squares, which include intersections between the key files A, E, and H and the key ranks 1, 4, and 8.
- 4×4 squares, which contain intersections between the key files A, E, and D, H and the key ranks 1, 4, 5, and 8.

The piece-square dataset contains three subsets: default, rotated, and visually shifted:.

TABLE 1 Prompt and ChatGPT output to explain the last move (bishop c4).

Prompt	
System message	You are a helpful chess assistant. Your role is to help the user understand the reasons behind chess opening moves, including their benefits, weaknesses, and overall strategic concepts. There are some instructions on how to respond to the user. Instruction 1: You should always start your response with a short sentence or few words to acknowledge that you've received the user's message and are considering it. Instruction 2: The user prefers concise explanations, so please keep your responses brief. Instruction 3: Your response should be informative and fluid, avoiding bullet points or index numbers. Instruction 4: When analyzing a board, mention one key reason for a move. Try to use the given history moves and predicted future moves. The input will include a user question, a chess FEN for the current game, the move to make, predicted future moves and a move history in a dictionary format under the keys 'question', 'fen', 'move', 'evaluation', 'future' and 'history'. Input:
User message	{'question':'can you explain?', 'fen':'r1bqkbnr/pppp1ppp/2n5/4p3/4P3/5N2/PPPP1PPP/RNBQKB1R w KQkq - 2 3', 'move':'Bc4', 'evaluation':'Excellent',\ 'future':'Bc4 Bc5 d3 Nf6 O-O d6 c3 O-O h3 h6 ',\ 'history': '1. e4 e5 2. Nf3 Nc6'}
GPT-4-0613	
Sure, I can assist with that. The move Bc4 in this position is known as the Italian Game. The aim of this move is to control the center quickly with your pawn and knight and then put your bishop on its most dangerous square. After this, both white and black usually castle and then start a pawn storm against the opponent's king. In this case, future moves predict a traditional development of pieces with both sides castling kingside. The main goal is to have a solid position and try to exploit tiny advantages.	

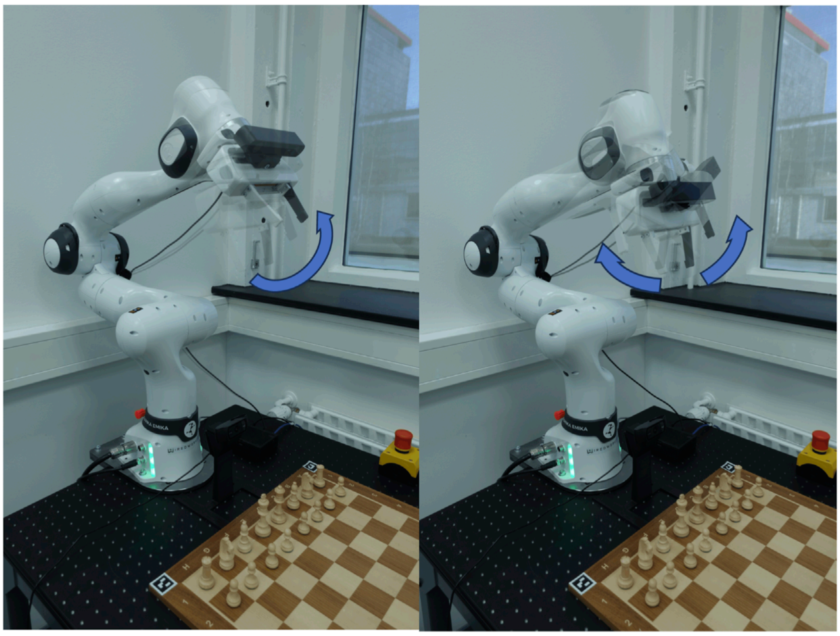

FIGURE 9
Posture feedback of the robot nodding and shaking its end-effector after a chess move was evaluated as good or bad, respectively.

- The default dataset **(D)** contains sub-images of 768 unique piece-square pairs (64 squares × 12 pieces), with each pair occurring at least once.
- The rotated dataset **(R)** includes sub-images of four types of pieces: white king, white knight, black king, and black knight. Kings were rotated twice by 45° (64 squares × 2 rotations), while knights were rotated four times (64 squares × 4 rotations) to create different shapes due to their rotational asymmetry (see Figure 10 right). Sixty-four samples of each rotated piece were randomly selected, forming the final rotated dataset.
- The visually shifted dataset **(S)** contains sub-images with cropping windows shifted in four directions (up, down, left, right). The displacement in each direction was ¼ square in length. For each shift direction, all 768 unique piece-square pairs were extracted. The **S** dataset serves as a test set with increased noise.

Additionally, the data recorded by the OpenChessRobot while replaying the games was used as another test set **(G)** to evaluate the piece classifiers trained with piece images on the key squares. Only unique piece-square pairs in each game are extracted, making up the **G** dataset.

Different compositions of training and test datasets were extracted from these three datasets:

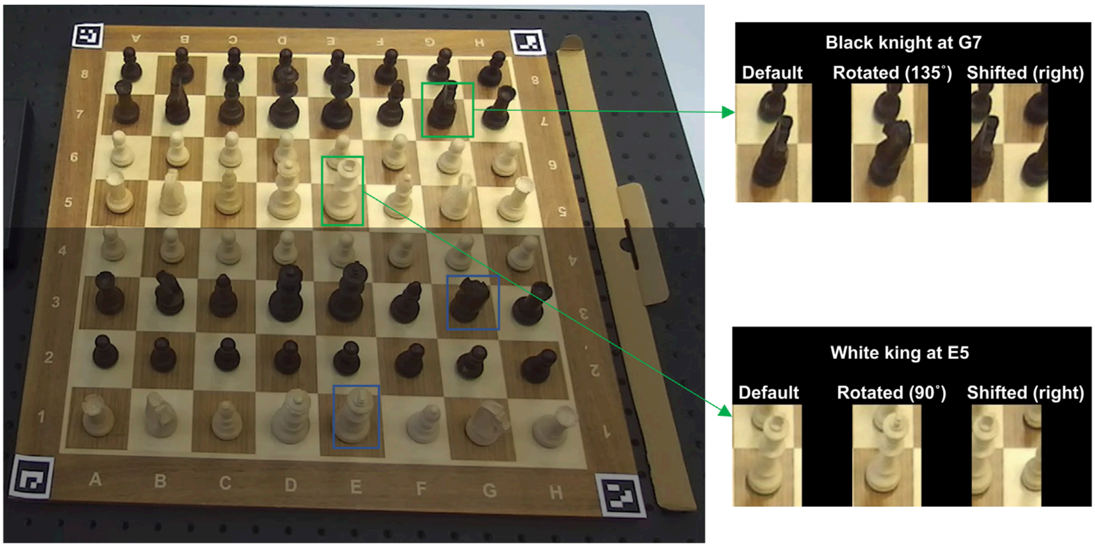

FIGURE 10
The raw images to create the piece-square dataset. The upper half image is taken without any rotations, and the lower half image is taken with 135-degree piece rotations.

- The training set **(d)** contained sub-images of each piece on the key squares. The remaining samples in the **D** dataset were divided a validation set and a test set in a 20:80 ratio.
- The training set **(r)** was generated from the **R** dataset using pieces on the key squares rotated by 90°.
- Similarly, the training set **(rr)** contains all pieces on the key squares with all the rotations from the **R** dataset. The corresponding test sets and validation sets of **r** and **rr** were created in the same manner as the **D** dataset.

The fine-tuning was conducted in two phases: in the first phase, the weights of most layers of the model were frozen, and only the last layer (head) was trained over epochs with a learning rate of 0.001. In the second phase, all the model's weights were trained with a learning rate of 0.0001. The Adam optimizer was used in both phases.

Following the aforementioned training conditions, the piece classifier models were evaluated on their corresponding test sets, as well as on the **S** and **G** datasets. The evaluation results are detailed in Table 2. When comparing the two selections of key squares based on the model accuracies on the **G** dataset, it is evident that using 4 × 4 squares provides more samples for fine-tuning and results in good accuracies with better robustness. Additionally, including the rotated samples in the training set also helps improve performance, but it requires a longer training duration to be effective. Results on the **S** dataset show that using 4 × 4 squares provides better robustness against the noise caused by piece displacements compared to using 3 × 3 squares. While longer training durations and adding rotations to the training set yield some improvements, these improvements are not substantial.

Regarding the occupancy classifier, it was found to achieve 100% accuracy on the **G** dataset after retraining for 50 epochs using piece sub-images at the 4 × 4 squares and all the adjacent empty squares.

5.2 Grasping of pieces

An experiment was performed in which the OpenChessRobot was tasked with grasping various pieces positioned on different squares of the chessboard, repeated multiple times. To reduce positional discrepancies, a compensation of 1 cm along the y-axis and 0.55 cm along the x-axis (Figure 11) was applied as the system error.

Specifically, the evaluation focused on testing the grasp performance using the largest piece, the king (4 cm in width, 7.8 cm in height), and the smallest piece, the pawn (2 cm in width, 4 cm in height). These pieces were positioned at each of the four corner squares (A1, A8, H1, H8) and physically shifted from the edges of the squares to their centers (depicted in Figure 11). Two shifting distances were used: (1) placing the piece tangent to the square's edge or (2) 0.625 cm away from the edge, corresponding to 1/8 the length of a square. Each piece in each square position was tested 10 times.

Figure 12 shows the success rates for the different independent variables of the grasping experiment. A distinction is made between three categories: (1) *accurate grasping:* a correct grasp in 10 out of 10 trials, (2) *remedied grasping:* a correct grasp in 8 or 9 out of 10 trials (usually correct grasping occurred here in such a way that the piece slid into the grasper's teeth), and (3) *missed grasping*: a correct grasp in 0–7 out of 10 trials. Table 3 shows the corresponding results in numerical form. A trend can be noticed, whereby the grasping was least successful when the piece was positioned toward the edge of the square or toward the 'up' direction, i.e., in the positive direction of the y-axis (see Figure 11), positioned away from the base of the robot arm. It is hypothesized that this effect stems from image projection and coordinate transformation errors in the 3D chessboard localization.

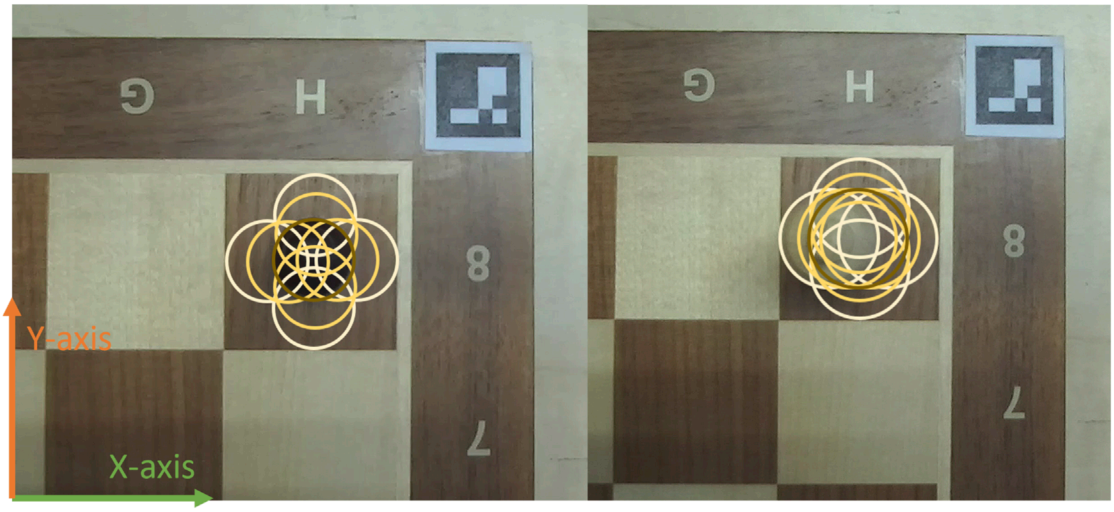

FIGURE 11
The positions by physically shifting the piece at square H8. In the left image, the black pawn is in the center without shifting. In the right image, the white king shifts to the left edge of the square. The brown circle shows the center position. The white circles indicate the first shifting distance (to the edge), and the yellow circles indicate the second shifting distance (intermediate).

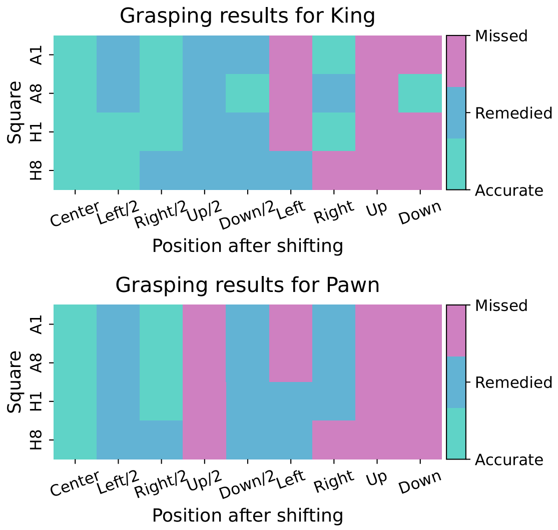

FIGURE 12
Results from the grasping experiment.

5.3 Performance in human-robot chess play

To assess the core components of the chess robot, a chess game between a human and the OpenChessRobot was conducted. The robot had control of the white pieces while the human played with the black pieces. To determine its next move, Stockfish 15, with a search depth of 20, was used.

The OpenChessRobot initiated the game from its Ready pose, executed a move, and subsequently returned to its Ready pose, awaiting the human player's move. The human player was prompted to press the middle keyboard button (see Figure 1B) upon completing their move. The manual button press was maintained to mirror traditional chess tournament practices and ensure clear separation of moves. This design also maintains safety and user control safeguards by providing an explicit signal before the robot begins its movement. The robot then captured an image, identified the human's move, and made its subsequent move. Two demo videos of human-robot chess play can be found in Supplementary Material Part C.

To ensure a fair match, the human player also had access to Stockfish 15 to determine their next best move. The game ended when a checkmate was achieved or a draw due to repetition. Throughout the experiment, we recorded the time taken by each robot module as it carried out its respective tasks.

The game ended in a draw after 97 moves, including 49 robot moves and 48 human moves. Out of these 49 robot moves, 7 involved capturing pieces, wherein the robot first removed the captured piece before executing its move, 6 moves required jumping pieces over others, resulting in longer execution times compared to moves made on unobstructed paths, and one was the castling. The other 35 moves followed the pick-and-slide approach.

The time taken for the OpenChessRobot to execute its moves is depicted in Figure 13A. On average, it took 7.33 s for the robot to complete a chess move, from its initial position to the resumption of that position after the move. Capture moves, on average, consumed 6.24 s. Figure 13B shows the time allocation for move detection, evaluation, and prediction. Move detection was typically completed within 1 s, with one outlier attributed to camera failure. Evaluating a single move with Stockfish took

TABLE 2 Piece classifier accuracies in different training conditions. The best accuracies on the S and G datasets are underlined.

	Key squares	3 × 3									4 × 4								
	Epoch	500 + 500			1000 + 500			1000 + 1000			500 + 500			1000 + 500			1000 + 1000		
Test set	Training set	d	d + r	d + rr	d	d + r	d + rr	d	d + r	d + rr	d	d + r	d + rr	d	d + r	d + rr	d	d + r	d + rr
D-d		0.9924			0.9943			0.9943			0.9978			1.0000			1.0000		
D + R-d-r			0.9794			0.9835			0.9863			0.9939			0.9939			0.9954	
D + R-d-rr				0.9915			0.9887			0.9930			0.9984			0.9922			1.0000
S		0.9762	0.9704	0.9684	0.9704	0.9691	0.9759	0.9652	0.9688	0.9740	0.9837	0.9831	0.9844	0.9834	0.9883	0.9886	0.9814	0.9886	0.9844
G		0.9932	1.0000	0.9932	0.9865	0.9662	1.0000	0.9932	0.9865	1.0000	0.9865	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

approximately 1 s. Finding the next move with Stockfish generally took approximately 5 s, unless Stockfish had already encountered the current human move during its previous search. In such cases, the search time was virtually zero, allowing the robot to execute the next move immediately. The relatively large computation time can be explained by the fact that Stockfish was allowed to search at a high depth, which is typically unnecessary for amateur-level play.

6 Evaluation of people's views on the chess robot

To collect an expert chess player's opinion on the OpenChessRobot, we involved a FIDE Master to play against the robot in two games (Supplementary Material Part D). The interview conducted after playing two games revealed that the robot, while potentially useful for beginner-level practice, lacks human-like feedback or commentary from which experts might benefit. The player also indicated that the robot might be useful for entertainment purposes.

In addition to the evaluation by an expert player, we conducted an online study to investigate how the broader population perceives the OpenChessRobot and to gather opinions on the future uses of such embodied AI in general. Participants watched a demo video of the robot and completed a corresponding questionnaire.

Specifically, demo videos were shown featuring the robot in three scenarios:

- The scenario *Robotics Education* featured the robot explaining how it works. The 28-s demo video showed the robot describing its perception function and how it localizes the chessboard.
- In the *Chess Coach* scenario, the robot teaches a chess opening. The 36-s video demonstrates specific moves and highlights key squares on the chessboard.
- The scenario *Home Entertainment* represents a future household robot that intends to entertain people through conversation and chess play. The 42-s demo video showed the robot discussing a chess game from a movie (Campbell, 1997). The videos and corresponding transcripts can be found in Supplementary Material Part C.

The choice of these scenarios is partly based on the aforementioned experiences with the expert chess player and also represents how (chess) robots could potentially be used in the future. That is, chess robots do not necessarily have to provide chess lessons (*Chess Coach* scenario) but can also support education by explaining their own functionality and serving as a knowledge resource (*Robotics Education* scenario). Additionally, robots can be seen as a companion or interaction partner (*Home Entertainment* scenario).

6.1 Participants

We aimed to recruit 600 participants through the online research platform Prolific (Douglas et al., 2023; Peer et al., 2021) from six countries where English is an official or *de facto* primary language:

TABLE 3 Piece grasping results. The percentages represent the number of trials that were marked as "accurate grasping" or "remedied grasping".

Piece	Overall		Shift distance		Shift direction			
	Without shifting (n = 4)	With shifting (n = 32)	Halfway (n = 16)	Edge (n = 16)	Left (n = 8)	Right (n = 8)	Up (n = 8)	Down (n = 8)
Pawn	100%	65%	75%	31%	75%	88%	0%	50%
King	100%	78%	100%	31%	63%	88%	50%	63%

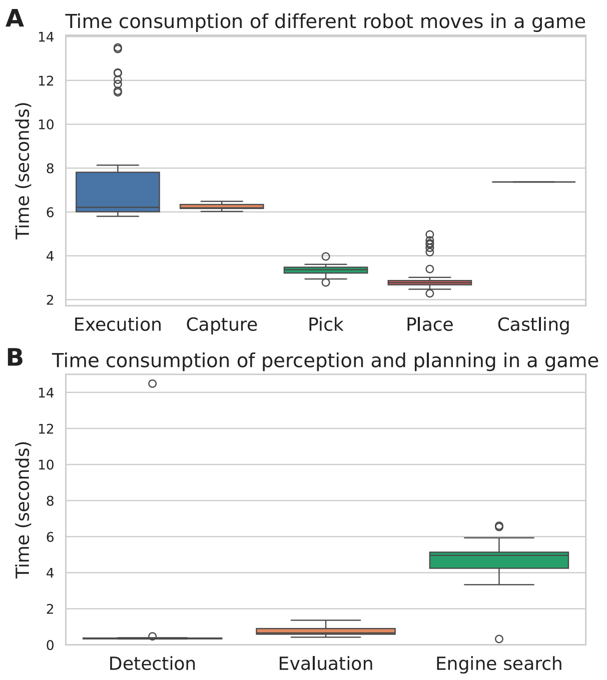

FIGURE 13
(A) Time consumption of different robot moves in the game. **(B)** Time consumption of move detection, Stockfish evaluating a move, and Stockfish searching for the next move.

Australia, Canada, Ireland, New Zealand, the United Kingdom, and the United States.

A questionnaire was launched on Prolific on Wednesday, 12 February 2025, at 22:00 Central European Time. In Prolific, prospective participants were provided with a hyperlink to the survey platform Qualtrics to complete our questionnaire titled "User perception of a talking chess robot". The purpose of the study was described as "*to examine how people evaluate a talking chess robot, which is powered by artificial intelligence including ChatGPT*". Once the target of 600 active participants was reached, the survey was automatically closed. Ultimately, 597 responses were collected. Respondents gave informed consent via a dedicated questionnaire item. Each respondent received a reward of £1.50 for completing the estimated 10-min questionnaire. The research was approved by the TU Delft Human Research Ethics Committee (Application Number 5109).

6.2 Experiment design

The online study followed a between-subjects design, with each participant presented with one of the three demo videos. All participants first answered questions about their familiarity with ChatGPT. After this, a short text was presented introducing one of the three demo videos, such as (*Chess coach* scenario): "*The following video shows a talking chess robot that uses artificial intelligence (including ChatGPT) to give a chess lesson. The robot explains a chess opening. Please turn on your sound.*"

Next, the demo video was played, followed by questions about people's acceptance (Q1), impression of the technical soundness of the robot (Q2), possible improvements (Q3), and opinions on the potential for such talking robots equipped with AI to replace existing technology (Q4) or human workers (Q5). The first three questions collected people's perceptions of the current OpenChessRobot, and the last two questions aimed to acquire their opinions on the future use of such talking robots equipped with AI. The five questions were as follows:

Q1. "*Do you think this robot could be used <in schools to explain to students how a robot works?>/<in schools to teach students chess?>/<at home to provide entertainment?>*" (Definitely not, Probably not, Neutral, Probably Yes, Definitely yes, I prefer not to respond)
Q2. "*Comment on the technical quality of the robot as shown in the video. Mention positive or negative aspects.*" (A response of at least 20 characters was required)
Q3. "*Do you have any recommendations about how to improve the robot?*" (A response of at least 20 characters was required)
Q4. "*Do you think that within 10 years, robots like this could serve as a replacement of existing screen-based < computers to help understand technical topics such as computer vision?>/<computers to improve chess skills?>/<forms of entertainment such as television?> Please elaborate on why or why not.*" (A response of at least 20 characters was required)
Q5. "*Would you welcome a future where robots that can talk, equipped with artificial intelligence (such as improved versions of ChatGPT), replace human workers?*" (Definitely not, Probably not, Neutral, Probably Yes, Definitely yes, I prefer not to respond)

The remaining questions gathered demographic information (age, gender, and education) and assessed participants' attitudes toward and familiarity with chess, technology, and the movie used in the *Home Entertainment* scenario. The full questionnaire can be found in Supplementary Material Part E.

6.3 Data analysis

For the open-ended questions (Q2–Q4), we used ChatGPT o1 (OpenAI, 2024) to analyze the responses (see Supplementary Material Part F for the full prompts and raw outputs). Previous research shows the possibility of using large language models to analyze textual data from user studies (Tabone and De Winter 2023). In addition, the reasoning model ChatGPT o1 has demonstrated its efficiency in textual content classification for social media analysis, sentiment analysis (Zhong et al., 2024) and categorical reasoning (Latif et al., 2024). We requested ChatGPT o1 to extract the three most mentioned positive and negative aspects from responses to Q2, the five most mentioned recommendations from responses to Q3, and the three most mentioned concerns from responses to Q4.

Additionally, for the open-ended question Q4, we used two reasoning models, ChatGPT o3-mini-high and Gemini 2.0 Flash Thinking, as text classifiers to classify the responses regarding people's opinions into *positive*, *neutral*, or *negative*. If the two classifiers produced different predictions, a human annotator manually determined the final label.

6.4 Results

A total of 195, 211, and 191 participants were randomly allocated to the *Robotics Education*, *Chess Coach*, and *Home Entertainment* scenarios, respectively. The median time to complete the survey was 6.48 min. The mean age of the participants was 36.5 years (SD = 12.33, n = 597). The gender distribution was: 302 females (50.6%), 286 males (47.9%), 8 *"other"* (1.3%), and 1 *"I prefer not to respond"*.

Figure 14 presents the distribution of responses for Q1, Q4, and Q5. For Q4, the two reasoning models classified the opinions identically for 528 participants, while a human annotator manually labeled the remaining 69 responses.

6.4.1 Analysis of acceptance per scenario

Acceptance was the highest for the *Robotics Education* scenario (Figure 14A), with a mean of 3.96 on a scale of 1 ("*Definitely not*") to 5 ("*Definitely yes*") (SD = 0.98, n = 195) (Q1). According to the ChatGPT o1 analysis, the most mentioned positive aspects for this scenario were the "*Advanced technical ability*" in terms of computer vision and the "*Clear voice/explanation*" demonstrated by the robot (Q2). However, participants often found the technical language in this scenario rather hard for beginners or students to comprehend.

The acceptance of the *Chess Coach* scenario was the second highest, with a mean of 3.82 (SD = 1.05, n = 210, excluding one participant who chose "*I prefer not to respond*") (Q1). The most mentioned positive aspect specifically for this scenario according to ChatGPT o1's analysis of Q2 was that it was seen as an "*effective teaching approach*". However, participants also often mentioned the "*limited interactivity*" as a downside.

The acceptance of the Home Entertainment scenario was the lowest (mean = 3.59, SD = 1.07, n = 191), with only 15% of the participants answering "*Definitely yes*" to Q1. A similar pattern can be observed in the classified opinions of Q4, where a majority of participants (57%) expressed a negative expectation regarding the use of such a robot to replace screen-based entertainment like television (Figure 14B).

6.4.2 Analysis of open-ended questions across the three scenarios

The summary results of the open-ended questions are shown in Table 4. Participants frequently praised the smooth and precise movements of the robot and its clear teaching style, but they criticized its robotic voice, mechanical noise, and slow operation. Common recommendations included adopting a more natural, human-like voice, simplifying the explanations, reducing the mechanical noise, speeding up performance, making the design more friendly, while keeping the robot cost-effective.

7 Discussion and conclusion

In this paper, we introduced an open-source cognitive robot designed for engaging in chess matches with humans. While multiple chess robots have been developed over the years, none have been made available as a reproducible platform, and many of them have not been specifically designed for conducting HRI studies. Our robot takes a unique approach by integrating robust robotic perception, evaluation of gameplay, and move execution with verbal and non-verbal interactions. Our development prioritized adaptability and reproducibility, to create an accessible platform for researchers and enthusiasts alike.

We focused on the game of chess because of its value as a controlled experimental environment where both human behavior and machine performance can be accurately measured. Chess also serves as a 'battleground' for Moravec's paradox, which states that while machines excel at computational tasks (such as playing chess), they traditionally struggle with tasks requiring human-like perception, motor control, and language processing (Moravec, 1988). Our research platform aims to transcend Moravec's paradox by integrating chess engine capabilities with innovations in human-like perception and the verbalization of moves using synthetic speech.

Furthermore, the integration of verbal and non-verbal interactions is included to increase the depth of engagement between humans and the robot. By using a large language model for verbal communication, we tried to create a solution that lowers the barrier of understanding the reasons behind robot chess moves. Our platform, in essence, revisits the conceptual foundation laid by the Mechanical Turk in the 18th century (Standage, 2002).

We invited an expert chess player to play with the platform and found that although it can be useful for beginners, it failed to provide human-like in-depth feedback from which expert-level players might benefit. Additionally, we conducted an online survey with 597 participants recruited from six countries to gather their views about the OpenChessRobot as well as talking robots equipped with AI in general. The online survey results revealed that participants favored the robot's use in educational contexts (where its advanced technical skills were appreciated) while expressing reservations about its role in home entertainment, primarily due to concerns over

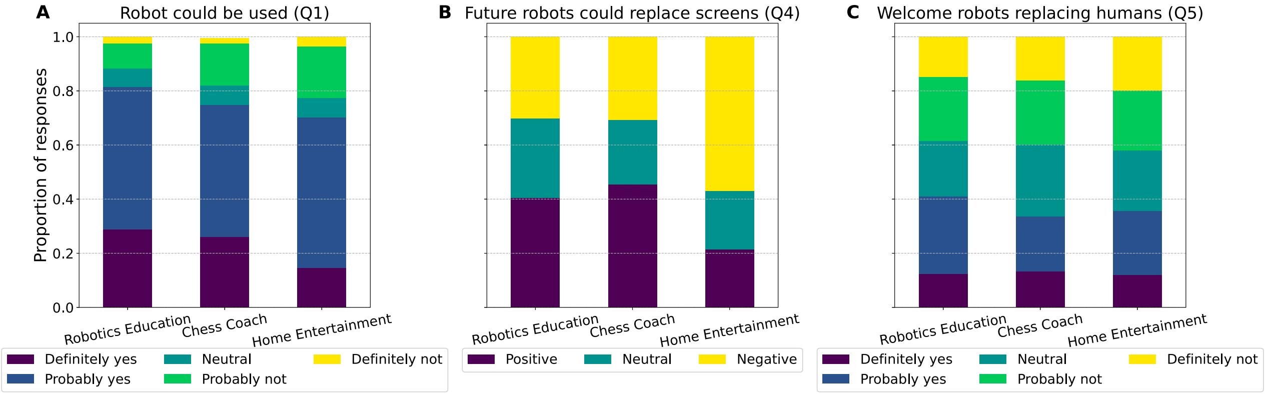

FIGURE 14
(A) The proportion of responses to Q1 "Do you think this robot could be used ...", **(B)** Q4 "Do you think that within 10 years, robots like this could serve as a replacement of existing screen-based ...Please elaborate on why or why not." (answers classified into three categories), and **(C)** Q5 "Would you welcome a future where robots that can talk, equipped with artificial intelligence (such as improved versions of ChatGPT), replace human workers?".

TABLE 4 Summary of results from the open-ended questions analyzed by ChatGPT o1. Underlined items indicate aspects that were commonly mentioned across all three scenarios.

	Theme	Robotics education	Chess coach	Home entertainment
Q2	The three most mentioned positive and negative aspects	Positive • Smooth and precise movement • Clear voice/explanation • Advanced technical ability (vision, analysis) Negative • Overly technical/Complex language • Slow or lengthy process • Robotic or distracting Voice/Noise	Positive • Smooth and precise movement • Clear explanations • Effective teaching approach Negative • Robotic or unnatural voice • Noisy or distracting movements • Limited interactivity	Positive • Smooth and precise movement • Intelligent or knowledgeable • Entertaining or interactive Negative • Loud or noisy • Bulky or large • Slow or boring
Q3	The five most mentioned recommendations	• More natural, varied, or human-like voice • Simpler, age-appropriate explanations • Friendlier or more appealing design • Reduced mechanical noise • Faster or smoother operation	• More natural, simpler speech • Reduce mechanical noise • Friendlier design and smaller size • Greater interactivity and Q&A • Clearer movement cues	• Reduce mechanical noise • Use a more natural, expressive voice • Make the design more compact • Improve overall aesthetics/approachability • Enhance movement fluidity
Q4	The three most mentioned concerns	• High cost and limited cost-effectiveness • Acceptance and human interaction • Incomplete or restricted functionality	• High cost • Limited portability • Reduced versatility	• Lack of variety and visual/storytelling depth • Preference for passive entertainment • Insufficient human/emotional element

limited interactivity and its inability to replace traditional screen-based media. These findings motivate future research on HRI in educational settings using this robot.

One of the limitations of the OpenChessRobot, as also shown in Table 4, is that its movement is rather slow and noisy. Another limitation is that the ChatGPT-based explanations are still narrow in scope and quality. To improve the robot, we will focus on faster arm movements and the incorporation of advanced large language models that are specifically trained or fine-tuned on chess-related text databases (see also Feng et al., 2023). Here, there is a need for explainable AI, where the chess position, represented by a FEN notation, is translated into a verbal explanation that is not only correct in terms of the chess engine's evaluation (as our current robot already does) but also contains meaningful content about why a particular move is good or bad. Moreover, although the results from the online study demonstrate promising aspects of the platform, any application beyond the research setting, such as education, coaching or entertainment, should be researched further. Finally, a limitation of the current paper is the lack of physical human-subject experiments among a large number of chess players. In the future, we intend to use this setup to study how AI-embodied robots influence people during interactions. This will involve the robot communicating with humans through emotional expressions and more natural/meaningful verbal interactions.

In conclusion, this paper presents the OpenChessRobot, an open-source, reproducible cognitive chess robot that integrates robust computer vision, chess engine evaluation, and both verbal and non-verbal interactions for HRI studies. A large language model is used to translate the evaluation from the chess engines into human-like speech. Supported by an international survey and expert gameplay, our findings show the robot's potential in educational and research contexts, while highlighting limitations in human-like interaction.

Data availability statement

Models for chess recognition used in the OpenChessRobot software, material from the online study, images used to finetune and evaluate the models, raw data, and scripts for reproducing the analyses in the paper are available in a public data repository: https://doi.org/10.4121/1cb5bf64-468e-462a-a82ec847d88a7a86.

Author contributions

RZ: Conceptualization, Data curation, Formal Analysis, Investigation, Methodology, Software, Validation, Visualization, Writing–original draft, Writing–review and editing. JW: Conceptualization, Formal Analysis, Methodology, Supervision, Validation, Writing–review and editing. DD: Resources, Supervision, Writing–review and editing. HS: Funding acquisition, Supervision, Writing–review and editing. YE: Conceptualization, Funding acquisition, Methodology, Project administration, Resources, Supervision, Writing–review and editing.

Funding

The authors declare that financial support was received for the research, authorship, and/or publication of this article. The authors declare that financial support was received for the research, authorship, and/or publication of this article. Specifically, this project has received funding from the Cohesion Project of the Faculty of Mechanical Engineering, Delft University of Technology.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/frobt.2025.1436674/full#supplementary-material

References

Campbell, M. (1997). "An enjoyable game: how HAL plays chess," in *HAL's legacy: 2001's computer as dream and reality*. Editor D. G. Stork (MIT Press). doi:10.7551/mitpress/3404.003.0007

Campbell, M., Hoane, A. J., and Hsu, F.-h. (2002). Deep Blue. *Artif. Intell.* 134, 57–83. doi:10.1016/S0004-3702(01)00129-1

Chen, A. T.-Y., and Wang, K. I.-K. (2019). Robust computer vision chess analysis and interaction with a humanoid robot. *Computers* 8, 14. doi:10.3390/computers8010014

Christie, D. A., Kusuma, T. M., and Musa, P. (2017). "Chess piece movement detection and tracking, a vision system framework for autonomous chess playing robot," in *Proceedings of the 2017 Second International Conference on Informatics and Computing*. Jayapura, Indonesia. doi:10.1109/IAC.2017.8280621

Coleman, D., Sucan, I., Chitta, S., and Correll, N. (2014). Reducing the barrier to entry of complex robotic software: a MoveIt! case study. *arXiv*. doi:10.48550/arXiv.1404.3785

Czyżewski, M. A., Laskowski, A., and Wasik, S. (2020). Chessboard and chess piece recognition with the support of neural networks. *Found. Comput. Decis. Sci.* 45, 257–280. doi:10.2478/fcds-2020-0014

Ding, J. (2016). *ChessVision: chess board and piece recognition*. Stanford University. Available online at: https://web.stanford.edu/class/cs231a/prev_projects_2016/CS_231A_Final_Report.pdf.

Douglas, B. D., Ewell, P. J., and Brauer, M. (2023). Data quality in online human-subjects research: comparisons between MTurk, Prolific, CloudResearch, Qualtrics, and SONA. *PLOS ONE* 18, e0279720. doi:10.1371/journal.pone.0279720

Feng, X., Luo, Y., Wang, Z., Tang, H., Yang, M., Shao, K., et al. (2023). ChessGPT: bridging policy learning and language modeling. *Adv. Neural Inf. Process. Syst.* 36. Available online at: https://proceedings.neurips.cc/paper_files/paper/2023/file/16b14e3f288f076e0ca73bdad6405f77-Paper-Datasets_and_Benchmarks.pdf.

Franka Emika (2020). Panda powertool. Available online at: https://web.archive.org/web/20200102112015/https://www.franka.de/technology.

Franka Robotics GmbH (2023). Franka control interface documentation. Available online at: https://frankaemika.github.io/docs.

Golz, J., and Biesenbach, R. (2015). "Implementation of an autonomous chess playing industrial robot," in *Proceedings of the 2015 16th International Conference on Research and Education in Mechatronics*. Bochum, Germany, 53–56. doi:10.1109/REM.2015.7380373

Gunes, H., Broz, F., Crawford, C. S., Rosenthal-von der Pütten, A., Strait, M., and Riek, L. (2022). Reproducibility in human-robot interaction: furthering the science of HRI. *Curr. Robot. Rep.* 3, 281–292. doi:10.1007/s43154-022-00094-5

He, K., Zhang, X., Ren, S., and Sun, J. (2016). "Deep residual learning for image recognition," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition*. Las Vegas, NV, 770–778. doi:10.1109/CVPR.2016.90

Kołosowski, P., Wolniakowski, A., and Miatliuk, K. (2020). "Collaborative robot system for playing chess," in *Proceedings of the 2020 International Conference Mechatronic Systems and Materials, Bialystok, Poland*. doi:10.1109/MSM49833.2020.9202398

Koray, C., and Sümer, E. (2016). "A computer vision system for chess game tracking," in *Proceedings of the 21st Computer Vision Winter Workshop*. Rimske Toplice, Slovenia. Editors L. Cehovin, R. Mandeljc, and V. Štruc

Kuffner, J. J., and LaValle, S. M. (2000). "RRT-connect: an efficient approach to single-query path planning,"*Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings*, 2, 995–1001. doi:10.1109/ROBOT.2000.844730

Kunz, T., and Stilman, M. (2013). "Time-optimal trajectory generation for path following with bounded acceleration and velocity," in *Robotics: science and systems VIII*. Editors N. Roy, P. Newman, and S. Srinivasa (MIT Press), 209–216. doi:10.7551/mitpress/9816.003.0032

Kuo, M. T., Hsueh, C. C., and Tsai, R. T. H. (2023). Large language models on the chessboard: a study on ChatGPT's formal language comprehension and complex reasoning skills. *arXiv*. doi:10.48550/arXiv.2308.15118

Larregay, G., Pinna, F., Avila, L., and Morán, D. (2018). Design and implementation of a computer vision system for an autonomous chess-playing robot. *J. Comput. Sci. and Technol.* 18, e01. doi:10.24215/16666038.18.e01

Latif, E., Zhou, Y., Guo, S., Shi, L., Gao, Y., Nyaaba, M., et al. (2024). Can OpenAI o1 outperform humans in higher-order cognitive thinking? *arXiv*. doi:10.48550/arXiv.2412.05753

LC, R., Benayoun, M., Lindborg, P., Xu, H., Chung Chan, H., Man Yip, K., et al. (2021). "Power chess: robot-to-robot nonverbal emotional expression applied to competitive play," in *Proceedings of the 10th International Conference on Digital and Interactive Arts*. Aveiro, Portugal. doi:10.1145/3483529.3483844

Leite, I., Pereira, A., Mascarenhas, S., Martinho, C., Prada, R., and Paiva, A. (2013). The influence of empathy in human–robot relations. *Int. J. Human-Computer Stud.* 71, 250–260. doi:10.1016/j.ijhcs.2012.09.005

Luqman, H. M., and Zaffar, M. (2016). "Chess brain and autonomous chess playing robotic system," in *Proceedings of the 2016 International Conference on Autonomous Robot Systems and Competitions*. Bragança, Portugal, 211–216. doi:10.1109/ICARSC.2016.27

Mac, T. T., Dam, T.-N., Nguyen, V. T. T., Hiep, L. H., and Nguyen, T.-D. (2023). "The development of an intelligent robot arm playing chess with human-robot interaction," in *Proceedings of the 2023 1st International Conference on Health Science and Technology*. Hanoi, Vietnam. doi:10.1109/ICHST59286.2023.10565356

Makoviychuk, V., Wawrzyniak, L., Guo, Y., Lu, M., Storey, K., Macklin, M., et al. (2021). "Isaac Gym: high performance GPU-based physics simulation for robot learning," in *Proceedings of the Thirty-Fifth Annual Conference on Neural Information Processing Systems*. Virtual event. Available online at: https://openreview.net/forum?id=fgFBtYgJQX.

Mallasén Quintana, D., Del Barrio García, A. A., and Prieto Matías, M. (2020). LiveChess2FEN: a framework for classifying chess pieces based on CNNs. *arXiv*. doi:10.48550/arXiv.2012.06858

Matuszek, C., Mayton, B., Aimi, R., Deisenroth, M. P., Bo, L., Chu, R., et al. (2011). "Gambit: an autonomous chess-playing robotic system," in *Proceedings of the 2011 IEEE International Conference on Robotics and Automation*. Shanghai, China, 4291–4297. doi:10.1109/ICRA.2011.5980528

Moravec, H. (1988). *Mind children*. Cambridge, MA: Harvard University Press.

MoveIt (2021). IKFast kinematics solver. Available online at: https://docs.ros.org/en/kinetic/api/moveit_tutorials/html/doc/ikfast/ikfast_tutorial.html.

NVIDIA (2019). NVIDIA Jetson Nano. Available online at: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-nano.

OpenAI (2024). OpenAI o1 system card. *arXiv*. doi:10.48550/arXiv.2412.16720

Open Robotics (2020). ROS Noetic installation instructions. Available online at: https://wiki.ros.org/noetic/Installation.

Peer, E., Rothschild, D., Gordon, A., Evernden, Z., and Damer, E. (2021). Data quality of platforms and panels for online behavioral research. *Behav. Res. Methods* 54, 1643–1662. doi:10.3758/s13428-021-01694-3

Pereira, A., Martinho, C., Leite, I., and Paiva, A. (2008). "iCat, the chess player: the influence of embodiment in the enjoyment of a game,"*In Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 3*. 1253–1256. https://dl.acm.org/doi/10.5555/1402821.1402844

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., et al. (2009). "ROS: an open-source robot operating system," in *Proceedings of the ICRA Workshop on Open Source Software*. Kobe, Japan.

Ranasinghe, P., Ranasinghe, P., and Ashan, V. (2023). "ChessEye: an integrated framework for accurate and efficient chessboard reconstruction," in *Proceedings of the 2023 Moratuwa Engineering Research Conference*. Moratuwa, Sri Lanka, 177–182. doi:10.1109/MERCon60487.2023.10355515

Sajó, L., Ruttkay, Z., and Fazekas, A. (2011). Turk-2, a multi-modal chess player. *Int. J. Human-Computer Stud.* 69, 483–495. doi:10.1016/j.ijhcs.2011.04.001

Schwenk, A., and Yuan, C. (2015). "Visual perception and analysis as first steps toward human–robot chess playing," in *Advances in visual computing. 11th International Symposium, ISVC 2015, Las Vegas, NV, USA, December 14–16, 2015, Proceedings, Part II*. Editors G. Bebis, R. Boyle, B. Parvin, D. Koracin, I. Pavlidis, R. Feris, et al. (Cham: Springer), 283–292. doi:10.1007/978-3-319-27863-6_26

Shannon, C. E. (1950). XXII. Programming a computer for playing chess. *Lond. Edinb. Dublin Philosophical Mag. J. Sci.* 41, 256–275. doi:10.1080/14786445008521796

Shin, K., Yamsani, S., Mineyev, R., Chen, H., Gandi, N., Lee, Y. J., et al. (2023). "Exploring the capabilities of a general-purpose robotic arm in chess gameplay," in *Proceedings of the 2023 IEEE-RAS 22nd International Conference on Humanoid Robots*. Austin, TX. doi:10.1109/Humanoids57100.2023.10375209

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., et al. (2018). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science* 362, 1140–1144. doi:10.1126/science.aar6404

Siraj, D. S. (2017). "JudgeMaster: automated chess movement validator," in *Proceedings of the 2017 National Information Technology Conference*. Colombo, Sri Lanka, 112–118. doi:10.1109/NITC.2017.8285662

Srivatsan, R., Badrinath, S., and Lakshmi Sutha, G. (2020). "Autonomous chess-playing robotic arm using Raspberry PI," in *Proceedings of the 2020 International Conference on System, Computation, Automation and Networking*. Pondicherry, India. doi:10.1109/ICSCAN49426.2020.9262351

Standage, T. (2002). *The Turk: the life and times of the famous eighteenth-century chess-playing machine*. New York: Walker.

StereoLabs (2020). Zed 2. Available online at: https://www.stereolabs.com/products/zed-2.

Stockfish (2022). Stockfish 15.1. Available online at: https://stockfishchess.org.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016). "Rethinking the inception architecture for computer vision," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition*. Las Vegas, NV. doi:10.1109/CVPR.2016.308

Tabone, W., and De Winter, J. (2023). Using ChatGPT for human–computer interaction research: a primer. *R. Soc. open Sci.* 10, 231053. doi:10.1098/rsos.231053

Thanh Tra, N. L., Cong, P. T., and Anh, N. D. (2018). "Design a chess movement algorithm and detect the movement by images classification using Support Vector Machine classifier," in *Proceedings of the 2018 4th International Conference on Green Technology and Sustainable Development*. Ho Chi Minh City, Vietnam, 335–340. doi:10.1109/GTSD.2018.8595604

Wölflein, G., and Arandjelović, O. (2021). Determining chess game state from an image. *J. Imaging* 7, 94. doi:10.3390/jimaging7060094

Xie, Y., Tang, G., and Hoff, W. (2018). "Chess piece recognition using oriented chamfer matching with a comparison to CNN," in *Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision, 2001–2009*. Lake Tahoe, NV. doi:10.1109/WACV.2018.00221

Zhelyabuzhsky, I. (2022). Stockfish. Avaliable online at: https://pypi.org/project/stockfish

Zhong, T., Liu, Z., Pan, Y., Zhang, Y., Zhou, Y., Liang, S., et al. (2024). Evaluation of OpenAI o1: opportunities and challenges of AGI. *arXiv*. doi:10.48550/arXiv.2409.18486