Reconnaissance Blind Multi-Chess: An Experimentation Platform for ISR Sensor Fusion and Resource Management

Andrew J. Newman, Casey L. Richardson, Sean M. Kain, Paul G. Stankiewicz, Paul R. Guseman, Blake A. Schreurs, and Jeffrey A. Dunne

The Johns Hopkins University Applied Physics Laboratory
11100 Johns Hopkins Road
Laurel, MD 20723

ABSTRACT

This paper introduces the game of reconnaissance blind multi-chess (RBMC) as a paradigm and test bed for understanding and experimenting with autonomous decision making under uncertainty and in particular managing a network of heterogeneous Intelligence, Surveillance and Reconnaissance (ISR) sensors to maintain situational awareness informing tactical and strategic decision making. The intent is for RBMC to serve as a common reference or challenge problem in fusion and resource management of heterogeneous sensor ensembles across diverse mission areas. We have defined a basic rule set and a framework for creating more complex versions, developed a web-based software realization to serve as an experimentation platform, and developed some initial machine intelligence approaches to playing it.

Keywords: Chess, Kriegspiel, Decision Making Under Uncertainty, Stochastic Control, Sensor Fusion, Sensor Resource Management, ISR, Intelligent Systems

1. INTRODUCTION

This paper introduces the game of reconnaissance blind multi-chess (RBMC) as a paradigm and test bed for understanding and experimenting with autonomous decision making under uncertainty and in particular managing a network of heterogeneous Intelligence, Surveillance and Reconnaissance (ISR) sensors to maintain situational awareness informing tactical and strategic decision making. The intent is for RBMC to serve as a common reference or challenge problem in fusion and resource management of heterogeneous sensor ensembles across diverse mission areas. We have defined a basic rule set and a framework for creating more complex versions, developed a web-based software realization to serve as an experimentation platform, and developed some initial machine intelligence approaches to playing it.

In future conflicts, opposing forces will be engaged concurrently in the sea, land, air, space, and information domains. Each side will attempt to hold the adversary's valuable assets at risk (e.g., combatant ships, mobile land-based weapon systems, aircraft, missiles in flight, communications and reconnaissance satellites, and command and control nodes) while concurrently attempting to protect its own. Moreover, each side will employ and orchestrate a diverse ensemble of sensing assets to collect the information needed to find, identify, and precisely track targets of interest, and more generally, to make timely, well-informed, and confident decisions on defensive and offensive measures. However, each side will attempt, through a variety of means, to counter the adversary's information acquisition through deception techniques and access denial. Commanders will need to assess threats and opportunities (e.g., to strike, to radiate, to move out of harm's way, etc.) without the benefit of full information; and, potentially, under the possibility of false information.

The effectiveness of decisions, tactics, and strategies at different levels of warfare rely on an accurate, current, and projectable picture of the multi-dimensional battlespace. Given the complexity, interconnectedness, and

Further author information:
Andrew J. Newman: andrew.newman@jhuapl.edu, 443-778-4193

DISTRIBUTION A. Approved for public release: distribution unlimited.

Signal Processing, Sensor/Information Fusion, and Target Recognition XXV, edited by Ivan Kadar, Proc. of SPIE
Vol. 9842, 984209 · © 2016 SPIE · CCC code: 0277-786X/16/$18 · doi: 10.1117/12.2228127

dynamics of the many elements of the battlespace, it is possible for even relatively small uncertainties, ambiguities, and errors to severely degrade the effectiveness of decisions, tactics, and strategies over time with consequences that range from temporary or minor to durable or catastrophic. It would be extremely useful, therefore, to have methods that quantify the relationship between uncertainty (or, conversely, information) and decision effectiveness, and that can compute those quantities at a given moment in time for a given state of the battlespace. Given such methods, commanders would have a means of optimizing, with respect to a well-defined and mission-relevant value function, the orchestration of their respective sensing ensembles and the countermeasures they use to disrupt the adversary's information acquisition.

There is a long history of using chess as a paradigm for understanding and developing advanced methods for tactical and strategic decision-making; and in recent decades as a test bed for machine intelligence algorithms. In standard chess, opposing players (commanders) engage on an eight-by-eight grid. Each player attempts to hold the opponent's pieces at risk while concurrently attempting to protect his own. Players assess threats and opportunities with full information in a well-structured domain where the set of potential disposition of forces at the next move is relatively straightforward to compute and remains finite (but explodes combinatorially) over any number of future moves. There is a large body of literature on quantifying the relative value of pieces (assets), the relative advantage and disadvantage of the state of the board (battlespace), and the improvement gained by individual moves, combinations of moves (tactics), and the pros and cons of different strategies.

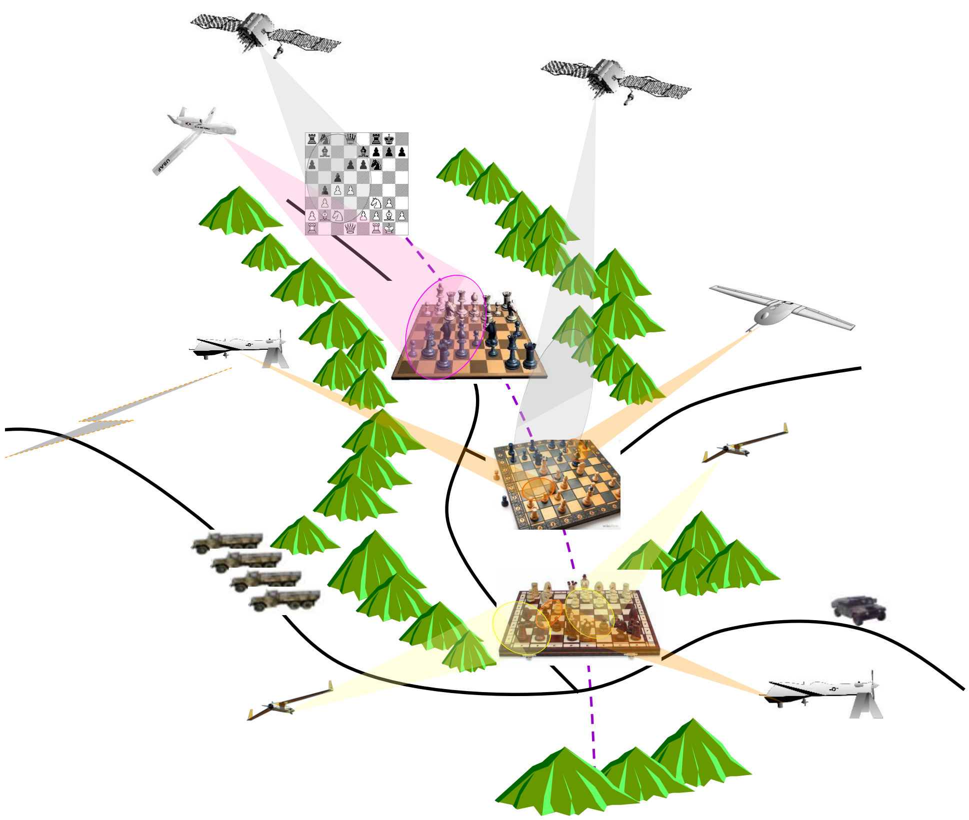

Figure 1. Metaphorical illustration of RBMC as a reference problem for orchestrating and fusing multi-modality sensing ensembles to inform tactical and strategic decision-making in warfare.

However, standard chess is not an entirely suitable experimentation platform for understanding and improving tactical and strategic decision making in warfare because it lacks the key elements of "fog and friction" (as per Clausewitz[1]), management of limited sensing resources, and adjudication among multiple, competing objectives. The game of RBMC introduces these factors as dominant elements of the competition. It is inspired by the rich history of chess variations that add the elements of uncertainty and partial information to the contest. These imperfect information games have been used to intensify the test of cognitive abilities between human players and to study and compete in the art of decision-making under uncertainty. The key differentiators of RBMC are the elements of incomplete information, competing priorities, and taskable sensors. The controllable reconnaissance element is new within the universe of chess variants and is the driver of this research problem. It is envisioned that the tactics and strategy for RBMC are fundamentally different than for chess; and consequently therefore machine intelligence algorithms for RBMC will be a new area of research.

In RBMC, the players are not able to directly view the chess board. The chess board can be observed only through sensing actions, and situational awareness can be achieved only through inferences made on those observations. Each player is allocated a limited supply of sensing resources to interrogate the chess board. The player's sensor ensemble is, in general, heterogeneous and noisy, and it has some predefined spatial, informational, and temporal agility. We have developed a rule set governing definition and allocation of sensing resources, described in this paper.

In addition, the players may be competing in a number of blind chess games running concurrently. This introduces the element of multiple, competing objectives, which is also a key element of warfare. This represents the problem that a commander at a certain echelon has in dividing attention and resources among multiple missions and goals. We can also introduce the element of friction by injecting a degree of randomness into the outcomes of moves. The concept of a controllable multi-modality ISR sensing ensemble collecting information supporting a "multi-chess mission" is illustrated metaphorically in Figure 1.

It is envisioned that RBMC will serve as a common reference or challenge problem in fusion and resource management of heterogeneous ISR sensor ensembles with applications to important mission-relevant problems across military domains and mission threads (land, maritime, air, space, and cyber). Such reference or challenge problems have a long and very useful history in science and engineering. Because each tactical situation or mission is different, specific applications of sensor fusion and resource management techniques derived from the vast literature on that subject tend to be situation specific. The RBMC paradigm provides an experimentation platform where information valuation and the sensor management trade-offs can be studied within a structured, well-understood framework.

2. CHESS AND GAMES OF IMPERFECT INFORMATION

2.1 Computer Chess

This paper is concerned with creating a variant of chess that can serve as an experimentation platform for autonomous perception and decision making under uncertainty with a particular orientation toward those problems encountered in military ISR missions. We first observe that chess has been used for decades as a well understood, convenient benchmark problem in artificial intelligence and machine learning. There is a vast literature and a large, world-wide research and development community in computer chess, which provides an advantageous point of departure this new research problem. The history of computer chess has been thoroughly chronicled in a variety of publications (see, e.g., [2,3]). Some important milestones and phases are:

- 1947: Alan Turing designs the first program to play chess (paper and pencil).
- 1950: Claude Shannon publishes the groundbreaking paper on a relay-based chess machine.[4]
- 1955: Dietrich Prinz implements the first working chess program.
- 1958: Allen Newell and Herbert Simon developed pioneering algorithms.[5]
- 1970's-1990's: Computer chess tournaments and consumer electronics proliferate.
- 1997: Deep Blue defeats human world champion Garry Kasparov.[6]

As stated earlier, standard chess is an adversarial game of perfect information, i.e., the board state is known fully and continually by both players. In any game of perfect information, under the assumption of exhaustive search, the optimal sequence of moves for each player, and the outcome, are deterministic. The solution may be obtained by recursively evaluating the optimal value function in a search tree from every board position (game state). However, exhaustive search is infeasible for games with large numbers of legal moves per board position (referred to as game breadth) and potentially lengthy move sequences (referred to as game depth). Silver and co-workers[7] provide a concise summary of practical solution approaches to optimizing move decisions for games of perfect information (in particular within the context of the game of Go). Essentially, search depth can be reduced by approximating the value function beyond a certain depth of moves and search breadth can be reduced by sampling move actions from a suitably designed decision policy. Most existing chess engines are based on von Neumann's fixed-depth minimax algorithm,[8] which is commonly used with alpha-beta pruning to reduce the search space. More recently, computer chess researchers have experimented with rollout, tree search, supervised learning, and reinforcement learning.[9]

2.2 Chess Variants

There is a very long history and broad diversity of chess variants (see, e.g.,[10]). Blindfold chess[11] and simultaneous chess are variants that augment chess as a test of memory capacity and cognitive processing power. In blindfold chess (first recorded game: Jubair (665–714), Middle East), the players may not see the board and moves are announced and known to the players (who must keep track mentally). In simultaneous chess, one or more of the players competes on multiple boards simultaneously. These variants have often been combined and used by experts to exhibit their mastery over common players. (The world record result for simultaneous blindfold chess is held (in dispute) by Najdorf (1947) versus 45 opponents; 39 wins, 2 losses, and 4 draws.) For human players, these variants are related to the problem of chess with uncertain or partial information in the sense that the player needs to minimize and compensate for information loss due to human memory limitations. But for computers, each board represents a decoupled, standard, full information game.

On the other hand, Kriegspiel (see, e.g.,[12–15]) is a chess variant in which the moves are covert and players must decide and act with incomplete information. It was invented as a chess variant by Henry Temple in 1899 and based upon a simulated warfare game of the same name (the German translation of Kriegspiel is war game) developed by Georg von Rassewitz in 1812 as a tool to train officers in the Prussian army. It was later used successfully by the Japanese navy during the Russo-Japanese war (1905). Kriegspiel itself has numerous variants. Reconnaissance chess may be thought of as Kriegspiel with a controllable, heterogeneous sensing ensemble (and potentially other controllable countermeasure elements).

2.3 Chess With Uncertain or Partial Information

There is relatively little work in the area of computational methods for chess variants with imperfect or partial information when compared to the large body of literature on algorithms for standard chess. The algorithms need to evaluate a huge number of board configurations and game sequences to deal with uncertainty. The existing literature centers on development of computationally tractable approaches for Kreigspiel and Kreigspiel-like chess variants from both a theoretical and implementation point of view.

Due to the complexity involved, the early work focused on playing Kriegspiel in simple endgames (e.g.,[16,17]). Applying these techniques to a more complex position or full games is prohibitive without an equivalent theory of move selection in the general case. A more promising set of approaches that has had success playing whole games involves the use of so-called metapositions. Sakuta and Iida[18] examined search methods that model the uncertain game state as a collection of positions, called metapositions, in a manner that is somewhat analogous to using mixture models. The transitions between these mixed states are then metamoves. They apply this idea, along with graph search techniques, to a game called Tsuitate-Tsume-Shogi, which is a Shogi (Japanese chess) variant where the player must solve a checkmating problem with imperfect information. Ciancarini and Favini[19] extended this approach with a tree search algorithm and minimax policy selection to implement a Kriegspiel algorithm known as Darkboard, which has achieved success in competition, and by developing Monte Carlo methods[20] for constructing and searching the move tree in the framework of Kriegspiel with metapositions. Favini[21] presents an excellent summary of this computational approach to Kriegspiel as well as some additional computational issues.

Bud and co-workers[22] analyzed an interesting information-theoretic viewpoint for a Kreigspiel-like chess variant called Invisible Chess. They estimate the branching factor that is possible from a given (uncertain) board position. These branching factors are then used to estimate the probability that certain configurations are the true position. From this probability density one can define notions such as entropy and the utility of attempting to move certain pieces. Russell and Wolfe[23] extended existing AND-OR searches (such as depth-first or proof-number searches) for game trees. The approach incorporates actual game states (which comprise the metapositions that are being searched) in its expansion selection. This allows particular branches to be disproved in a more efficient manner and translates to performance improvements in the move selection.

2.4 Other Imperfect Information Games

Just as Kriegspiel was derived from chess, imperfect information variations of a number of other games have been developed to study problems in decision making under uncertainty. One of the most significant examples is Phantom Go, a variant of classic Go in which the players do not see the opponent's moves, and which has spawned a playing community, tournaments, and algorithm development efforts .[24] In addition, there are many games designed inherently around the imperfect information aspect, and which offer a reconnaissance element that players must use effectively to learn the board state faster and more accurately than the opponent. Examples include Battleship which involves a search for static targets on a grid and a very simple sensing model; Stratego which involves interrogating dynamic targets with uncertain rank but sensing only via maneuver and direct contact (there are variants that include some longer range sensing); and a similar Chinese game called Luzhanqi. In Bridge, players query and infer about unknown cards through bidding and playing known cards. In Poker, players attempt to infer their opponent's relative strength of position based on betting behavior and behavioral "tells".

It is important to note that in all of the above games, including Phantom Go, a newly discovered piece of information remains valid for the remainder of the game. This makes those games less of a suitable proxy for miliary ISR problems in which the battlespace is rapidly evolving and information goes stale quickly. By contrast, blind chess offers a game of imperfect information in which the dynamics are such that board state and strength of position can change quickly.

The other critical element of reconnaissance chess not present in other variations of formal strategy games is the commanding of sensing assets to perform the reconnaissance element. One interesting example of an adversarial game with a sensor control element is robotic capture-the-flag investigated by Huang and co-workers.[25] They investigated full information and partial informaiton variations; including the introduction of aerial reconnaissance (via quadrotor UAV) and sensor management (controlling the aircraft flight path to optimize the probability of finding the opponent). Their work illustrated some of the interaction between tactical maneuver decisions and sensor allocation decisions and other complexities we expect to discover in RBMC.

3. RBMC GAME DEFINITION AND VARIATIONS

3.1 Rule Specializations for Reconnaissance Chess

There are several rule specializations that are needed to handle situations that appear in chess games with imperfect information but do not appear in standard chess. In most cases, these rules could be defined in different ways. We have adopted rules with the primary goal of keeping the basic game definition as simple as possible.

- The conditions of "check" and "mate" do not apply and are not used. A player cannot establish deterministically whether he or his opponent is in check or mate. The goal, therefore, is to actually capture the opponent's king. The game ends when a king is captured. In addition, move constraints associated with check are eliminated. It is legal for a player to move such that the result leaves or puts his king in check (of which he or his opponent may not be aware).
- It is possible for a player to command a piece (queen, rook, or bishop) to move along a row, column, or diagonal to a square that is actually impossible to reach because another piece (possibly unknown to the player) is interposed. We have adopted the rule that, in such cases, the piece moves along the commanded row, column, or diagonal to the square with the interposed piece and captures it.

- It is possible for a player to command a pawn to capture (diagonally; potentially en passant) where no such opportunity legally exists. We have adopted the rule that an illegal pawn capture attempt results in loss of move.

- It is possible for a player to command a king to castle but an opposing piece is blocking the move. We have adopted the rule that an illegal castling attempt results in loss of move.

- It is necessary to define in the rules whether each player will maintain knowledge of his own pieces throughout the game. We have adopted the rule, initally, that each player maintains perfect knowledge of the status and location of all of his own pieces.

- It is necessary to define in the rules whether the game arbiter will announce captures. We have adopted the rule that the game arbiter will notify both players any time a move results in a capture. However, the identity of the captured piece will not be announced (but is known by the player whose piece was captured due to the previous rule).

In our version of reconnaissance chess, the arbiter rejects any illegal move not covered by one of the above cases and forces the player to command a different move. This prevents players from commanding moves that are guaranteed to be illegal in order to intentionally skip a turn as a tactic.

3.2 Sensing Resources and Actions in Reconnaissance Chess

In reconnaissance chess, the board can be observed directly only through sensing actions. Each player commands one or several sensors to interrogate the chess board. The sensors are potentially different from each other in terms of field of view, precision of information, and other attributes. Each sensor has a predefined spatial and informational profile and agility. Sensors can be tethered to a chess piece (organic) or on a dedicated sensor platform (overhead). Each sensor's ability to acquire information is defined in terms of space (area coverage, i.e., number of squares), information type (detection, localization, identity, and color of pieces), resolution (precision), and time (consumed to reach a particular area and/or to achieve a particular coverage or quality). For example, as analogous to imaging sensors with a natural tradeoff among field of view, resolution, and frame rate, the player might have to choose between using his sensor to:

- count the number of pieces in a large portion of the board (or the entire board); or

- localize to within one square of uncertainty all pieces in a medium portion of the board (e.g., 4x4 area); or

- localize exactly all pieces in a small portion of the board (e.g., 2x2 area); or

- localize and identify the piece in a single square.

The player may have just one or several such sensors at his disposal to allocate over one or more chess boards. The sensing rules for a particular game prescribe the sensor ensemble (number of each type) for each player, and, for each sensing asset, prescribe its: platform motion profile or degrees of freedom, tethered piece (if applicable), sampling capacity or period, field of regard, field of view, detection sensitivity, detection specificity, resolution, localization, recognition, and potentially other attributes. These prescriptions can be quantified and parameterized in standard ways (e.g., using probability of detection, false alarm rate, confusion statistics, and similar quantifications).

We have defined a simple, fixed sensor type to serve as the canonical sensing resource for initial research in reconnaissance chess. This basic sensor has a 3 x 3 square field of view and senses truth without noise. The player commands the aimpoint (center square of the 3 x 3) and is allocated one use of this sensor per turn. Figure 2 illustrates an example of the basic sensor's field of view (left) and resulting updated board estimate (right). In the example, the Black player commands the sensor aimpoint to view square b3. As a result the sensor provides the true contents of squares a2, b2, c2, a3, b3, c3, a4, b4, and c4. The sensing action drastically changes Black's board state estimate within the 3 x 3 field of view (disregarding any further inferencing beyond the sensor's field of view).

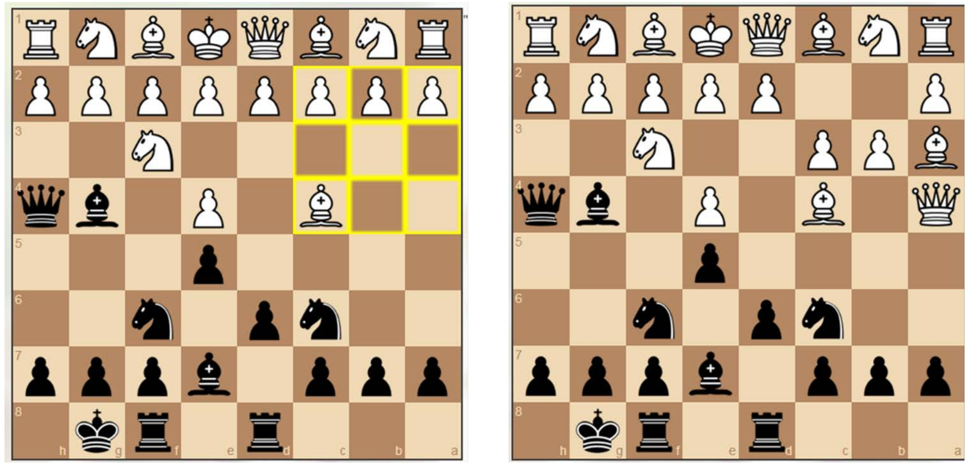

Figure 2. Example of basic sensor's 3 x 3 square field of view (squares a2-c4 outlined in yellow) commanded by the Black player (left) and Black's resulting updated board knowledge within that field of view (right). Note that Black has not yet updated its estimate beyond the sensed field of view, e.g., the estimate contains two White queens. Note also that the ranks (1-8) and files (a-h) are ordered, respectively, top-to-bottom and right-to-left from Black's point of view.

3.3 Multi-Chess Implications

The basic version of reconnaissance chess involves only a singleton game, which alone presents a high degree of complexity. However, the multi-chess option introduces an additional dimension of complexity into the experimentation platform. In multi-chess games, the players must make their resource allocation decisions in the context of multiple, competing objectives, which is also a key element of warfare. This represents the problem that a commander at a certain echelon has in dividing attention and resources among multiple missions and goals and introduces significantly more complex trade-offs and evaluation of opportunity cost into the competition.

In multi-chess the two players compete on multiple boards concurrently. The importance of the multiple boards can be weighted uniformly or according to some other non-uniform scoring system. By prescribing a different "win value" to each board we can represent the case of variable mission priority. It is also possible to adopt nonlinear and time-varying scoring schemes as well as to define rules for coupled or interacting boards. Each player is responsible for controlling his pieces on each board and allocating his sensing resources over the multiple boards. A player may adopt a strategy to allocate his sensing resources evenly over the boards or concentrate more of it on certain boards; or simply make tactical decisions according to his current board estimates.

3.4 Potential Variations

We have adopted a basic, canonical version of reconnaissance chess for the initial web server implementation and player client development. This basic version is singleton chess (vice multi-chess) that uses standard chess rules and game flow (alternating turns) with the rule specializations and basic 3 x 3 truth sensor described above. However, the rule space for RBMC is potentially much richer to achieve a flexible level of game complexity to support different research directions. The rule space potentially diverges from standard chess along several dimensions including game flow (e.g., discrete versus continuous; time limits; termination conditions), chess (e.g., noisy moves deviating from commands), multi-chess (as described above), sensing (as described above), counter-sensing, and communications (e.g., delays in reporting sensor data). We defer a deeper exposition of these dimensions to a future paper.

4. RBMC WEB-BASED SERVER AND EXPERIMENTATION PLATFORM

We have developed a web-based software realization that supports human versus human, human versus machine, and machine versus machine reconnaissance chess games. The RBMC web server currently provides a game arbiter, game display, user controllable game replay, and the capability to assign available machine player clients to the White and Black sides, respectively. The game arbiter enforces the rules, applies player actions (chess piece moves, sensor observations, and eventually countermeasures and other types), and maintains the game state (truth). The game display has a player mode and a spectator mode. The player mode is used by human players and shows only that player's board estimate and console messages. The spectator mode, illustrated in Figure 3, is used for live or replay analysis of games and shows the entire sequence of sensing and move actions along with the true board, White's estimate, and Black's estimate.

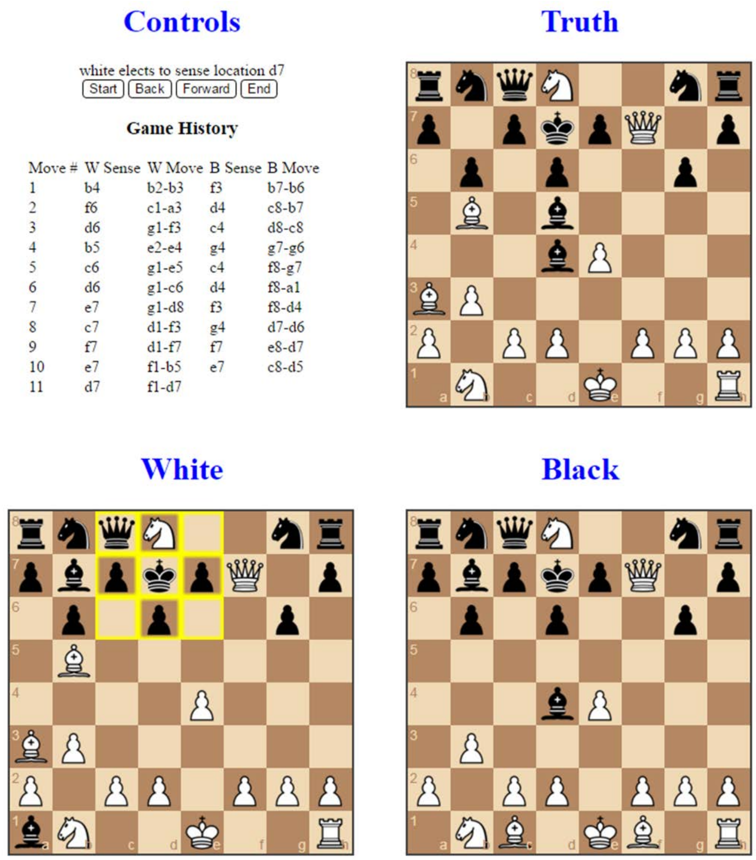

Figure 3. RBMC spectator display is used for live or replay analysis of games. In this example, White makes his sensing action to confirm the position of the Black king just before winning the game by capturing it with his bishop (move 11). Note that the true board, White's estimate, and Black's estimate all differ.

We have developed several exemplar machine player clients, described further in Section 5. Each player client is responsible for its own board state estimation, sensor management, and piece moves. A future version of the web server will provide the capability to configure and run Monte-Carlo experiments with large numbers of games to compile metrics for quantitative analysis. In this manner, different player clients, and hence different solution approaches and algorithms, can compete and be compared over large numbers of games. The web server exposes methods to player clients, which call them according to predefined interfaces. The interfaces will eventually be documented in an Interface Control Document (ICD) that would allow members of the research community to develop and test RBMC player clients by interfacing with the server.

The software is implemented in javascript, which is a high-level, dynamic, untyped, and interpreted programming language. A prime motivation for using javascript is the ability to leverage existing open source, modular chess programming facilities. We have made extensive use of existing open-source javascript packages for standard chess arbitration (e.g., chess.js[26] and stockfish[27]) and chess visualization (e.g., chessboard.js[28]). We considered other options such as python-based (e.g., pychess[29]) and C-based (e.g., xboard[30]) chess packages but they lack the flexibility of the javascript packages.

The software uses Forsyth-Edwards Notation (FEN) to represent the state of a chess board. FEN is a standard notation for encoding the state of a chess game. It represents the game state as a string with six parts separated by a space:

- Part 1: Piece-wise board configuration, row-by-row, with rows separated by the '/' symbol, and a unique symbol for each piece type ('p' for pawn; 'r' for rook; 'n' for knight; 'b' for bishop; 'q' for queeen, 'k' for king), where uppercase represents the White pieces and lowercase represents the Black pieces, and numbers representing the number of unoccupied squares between pieces in a row.
- Part 2: Turn indicator (binary: White or Black)
- Part 3: Castling indicators (player and directions)
- Part 4: List of squares that are potential targets for en passant capture
- Part 5: Half-move counter (to enforce the 50 move draw criterion)
- Part 6: Full-move counter (total moves for each player)

For example, the FEN string below represents a possible game state after White's first move:

rnbqknbnr/pppppppp/8/8/rP3/8/PPPP1PPP/RNBQKBNR b KQkq e3 0 1.

This fairly simple representation allows the full encoding of any game state, from which a game can be fully reconstituted and analyzed. While any game state can be encoded with a FEN string, the converse is not true, i.e., any string in the above form does not necessarily represent a valid game state (e.g., there cannot be three White bishops or any pawns in the back row). An admissible game state must be reachable by some sequence of legal moves.

5. SOLUTION APPROACHES AND STRATEGIES

5.1 The Fundamental Decision Loop

The fundamental decision loop, or perception-action cycle, in reconnaissance chess encompasses the sensing decision (specifically, for the basic game, where to sense), the inferencing process (updating board state estimates based on sensed data, chess rules, and potentially any predictive models of chess moves), and the move decision (which piece and to which square). In the basic game, each player (alternatingly) executes this fundamental Sense-Infer-Move decision loop once per turn. The competing, interacting decision loops are illustrated in Figure 4. Note that the Sense and Infer functions form their own, inner, "ISR" decision loop, in which the inferred game state informs future sensing decisions, which, in turn, produce data that updates the inferred game state. It becomes clear from the diagram that the decision making for sensing and move actions should be highly coupled in any effective strategy.

In the subsections that follow, we offer some general approaches to developing Sense, Infer, and Move decision algorithms for autonomous machine players. To date, we have combined some of the simple approaches to create exemplar player clients for testing human versus machine and machine versus machine games. We have also created a "random" player client (nicknamed Alpha 5 after the fictional Power Rangers robot) to serve as a baseline or default opponent for testing. The "random" player, as the name implies, executes the Move and Sense functions by, choosing, respectively, a random move from the space of legal moves (uniformly) and a random sensor aimpoint within the board interior. We plan to make our exemplar, simple and random player clients available to the community so that other researchers can test their own player clients against baseline or default opponents.

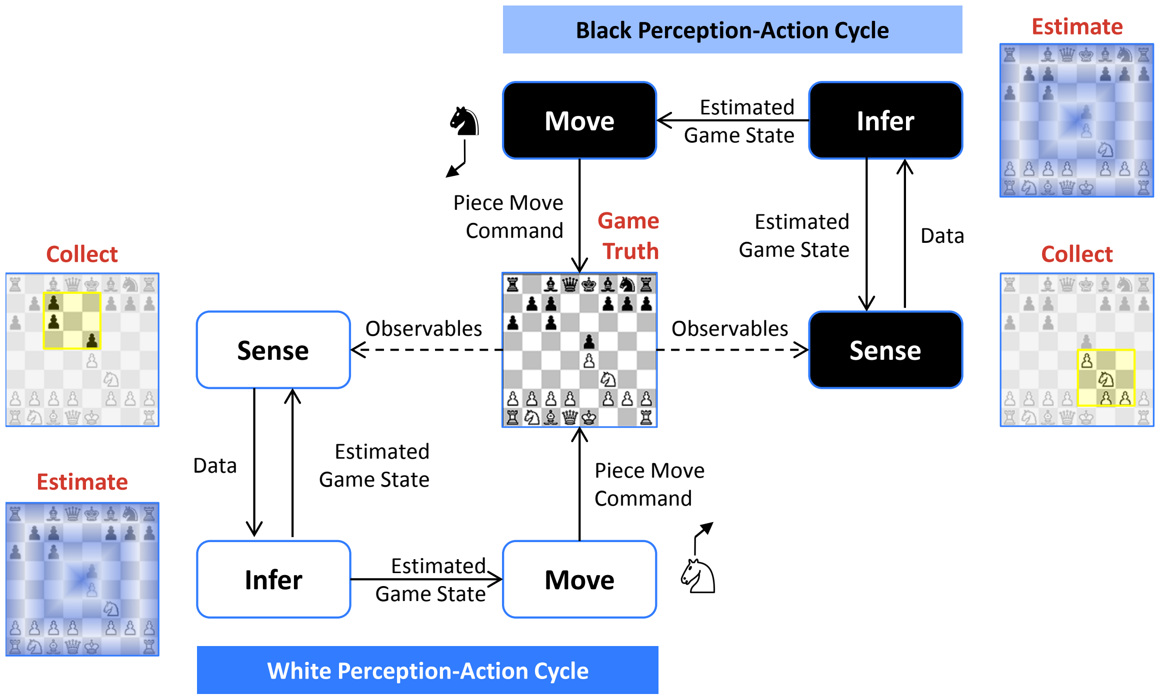

Figure 4. Competing decision processes (or perception-action cycles) studied by RBMC.

5.2 Move Decision Approaches

We begin with the Move function because it has been studied extensively in the context of standard, perfect information, chess. In RBMC, the Move function generates a legal chess move given an uncertain board state (where legality is defined by whatever RBMC rule set has been adopted). As in standard chess, the decision algorithm is typically decomposed into two primary elements: evaluation, which quantifies the utility or effectiveness of candidate moves (e.g., based on material, mobility, and positional factors); and move selection, which chooses an optimal, approximately optimal, or simply good enough move within the space of legal moves, given the pre-computed evaluations.

Despite the enormous volume of literature on algorithms for move decisions in standard chess, and the large number of competing chess engines available in software, there has been relatively very little research in algorithms for chess move decisions under uncertainty. Much of this is contained in the Kriegspiel literature, which is a special case since Kriegspiel has no reconnaissance element and the decisions must take into account that moves and captures are the primary means of sensing.

We begin to develop an approach by representing the uncertain board state as a discrete probability distribution on deterministic board states. The distribution could have infinite support so we truncate and normalize so that we are left with a relatively small number of board states, each with a probability weighting, which sum to unity. We can then use a deterministic chess engine, such as stockfish, to generate the "best" move for each board state in the distribution. As a special case, we could simply use the most likely board state and only generate one move. However, to retain as much fidelity of the distribution as possible, suppose we generate N candidate moves, one each from the top N most likely board states. There is no means of combining these N moves in a mathematical sense. We must pick one. A potential heuristic would be to score and rank them according to the likelihood of the generating prior board state and the win probability resulting from the move.

5.3 Inference Approaches

The Infer function maintains an estimate of the uncertain board state over time (moves) and given new data. It updates the board state estimate after each sensing and move action (for a discrete, alternating action game) or continually (for a continuous action game). In general it involves the standard Bayesian filtering steps: data update, which incorporates information from sensing actions, capture events, and arbiter alerts; and time (move) update, which predicts opponent actions and consequent board states given prior estimates.

The inferencing approach requires a probabilistic representation of the uncertain board state. There are a number of possibilities, which fall into the following general distinctions: single hypothesis versus multiple hypothesis; enumerative versus non-enumerative; and square-based versus piece-based. The most direct representation, although likely not the most practical, is to maintain a record of every possible board configuration and its history. For example, after the first move by White, Black recognizes 18 possible boards with their associated estimated likelihoods (18 because that is the number of possible first moves, and with likelihoods calculated based on whatever algorithm is used to assess opponent actions). The next move would then potentially eliminate some of those boards based on the result of sensing, and expand the remaining board configurations with a second tier of potential options and associated probabilities. Because of the number of possible moves, particularly during mid-game, this approach can quickly grow intractable.

An alternate that is much more constrained is to treat the position of each piece probabilistically, i.e. as being a field of probabilities distributed over the 64 board spaces plus the N number of captured pieces (where $N \leq 15$). In this approach, each move opportunity disperses probability in accordance with the rules of piece movement. The algorithmic details of this style of representation, which are deterministic and computational trivial, have been developed, although an in-depth description is beyond the scope of this paper. This approach has several advantages. It supports a single sense–move temporal field of consideration that matches human intuition closely, and its small memory footprint would permit full history storage. This full memory storage permits backwards evaluation for an even more precise representation of human analysis.

An additional advantage of this piece-based probabilistic treatment is that it can be translated into a square-based representation, where each board square is characterized as having a particular probability of occupancy for a given piece at a particular point in the game. This representation provides yet another approach for machine players to leverage in estimating the relative value of move options and potential threats.

The time (move) update step may be accomplished, simply, by diffusing probability mass uniformly over the space of legal moves for each piece to provide a baseline functionality. For higher fidelity, we can use a deterministic chess engine to generate a set of likely opponent's moves as a pseudo-predict function. Each data update step requires solution of an association problem to associate sensed or captured pieces to prior states. In certain cases there could be considerable ambiguity (e.g., "which pawn (or rook) did I just observe in that square and where did it come from?").

5.4 Sensing Decision Approaches

The Sense function generates one or more sensing actions given uncertain board state, sensor allocation, and sensor states (and according to the defined rules of sensing). It involves the quantitative evaluation of candidate sensing actions in terms of "utility" or "effectiveness" and selection of an action from the space of legal sensing actions given the evaluations. For the basic sensor type defined in our basic RBMC game, there are only 36 possible sensing actions (all inner squares of the board). This renders the optimization search to be trivial once the evaluation is complete. Note that this simplification would not necessarily apply to more complicated sensing rule sets in potential future versions.

One simple approach to the Sensing decision is based on recency of visit. For each square, we maintain a count of moves since it was last sensed. The recency score for a region would then be computed as the sum of the visit recency of the individual squares in the region. The Sense function would choose the least recently searched region at each turn. This policy would result in a relatively even sweep over the chessboard by the sensor.

We are interested in more sophisticated approaches that use entropy, relative entropy, and/or information-theoretic measures or divergences to quantify the information gained by a sensing action (or the expected information gain). However, purely information-theoretic methods do not necessarily account fully or precisely for the tactical or strategic value of the information. We will investigate risk-based sensor management techniques as a means of maximizing reduction in tactical risk (risk of using incorrect estimates in move generation). These techniques quantify and incorporate the magnitude of consequences of bad decisions derived from bad information.

5.5 Strategy Considerations for RBMC

At its root, winning reconnaissance chess is about discovering your opponent's king's location, clearing a path to it (or luring it into an open attack path), and making the capturing move. While this observation seems somewhat obvious, this represents a fundamental difference from traditional chess where checkmate requires a more comprehensively-arranged board position. This elevates the importance of tactical considerations to be on par with the long-term strategic planning of the overall game.

A lack of precise knowledge about adversary piece deployment has, not surprisingly, a profound impact on strategy and tactics. The most significant aspect of this is that, unlike in traditional chess, the threat of the game–ending sudden death can evolve much faster and is not solely a function of oversight (i.e. the "oops, I didn't see that move" loss). Lacking knowledge of adversary piece locations, the player must keep the king safe using a multi-pronged approach that includes blocking access with friendly pieces and monitoring approaches that cannot be blocked physically, such as threats from knights or when too few friendly pieces are available to cover all avenues of approach.

In the game's current incarnation, it is often the case that an opponent's move can be narrowed down to a small subset of possibilities, and consequently a single sense move can serve both defensive and offensive purposes. While this is good when it happens for you, it is preferable that this not be the case for your opponent. Single-minded offensive approaches offer the greatest opportunity for this to happen, and so a diversified attack strategy is generally preferable. Specifically, it is best when your strategy involves the coordination of pieces that are sufficiently far away from each other that they cannot be sensed in a single move.

Unexpected attacks, including those that trade down (taking a lower value piece with a higher value one), become more viable as options to consider, particularly when the attack could have come from multiple places that are widely separated. They require dedication of a sense action to recognize what has occurred, meaning that potentially more critical spots are not being observed. They have the advantage of misdirecting your opponent's attention, something that is considerably more impactful when the game is shrouded in uncertainty, and are particularly effective when arranged such that a subsequent capture results in a vulnerability to the king – a move that is illegal in traditional chess (one cannot put one's self in check), but is often unrecognized in the uncertainty of reconnaissance chess.

5.6 Illustrative Game

Consider the following illustrative human versus human game between one of the authors (White) and an opponent (Black). The corresponding chess boards showing White's and Black's estimates, respectively, after each sensing and move action, are shown in the indicated figures at the end of the paper, with the exception of the board immediately following White's opening move, shown here in Figure 5. In all chess board figures, the squares within the sensor field of view are outlined in yellow, and the squares from which and to which a piece moved are shaded gray.

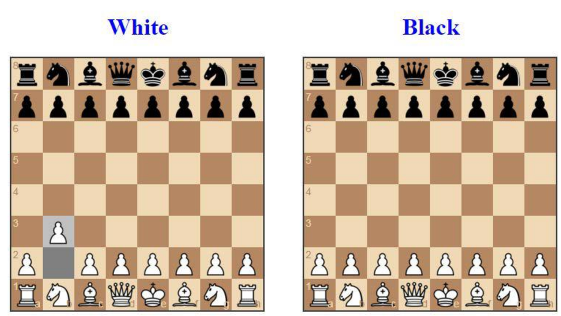

Figure 5. Illustrative game, action 1: White's opening move.

Actions 1-11 (Figure 6): White's opening move is not radically untraditional, but is specifically chosen because it is less likely to be detected by an opponent expecting someone to take control of the center. Black's sense move confirms that it had this effect, but only by luck. Although unknown by White's next sense move, Black is following a similar tactic. White elects to move the queen's bishop out to start putting pressure on some of the board.

Black's sense move, while providing no overt information, actually does potentially give some insight that White is playing a strategy that involves sneaking up the edges of the board. Black also moves a bishop out, a move that goes undetected by White. At this state, neither opponent has been able to positively detect any movement by the other side. White is not aware of Black's sensing effectiveness, however, and moves out a knight on the other side of the board so that Black cannot watch all of his pieces with a single sense. Black's next sense move detects movement, and Black manually updates the b2 square as being empty since it is the only pawn that could have occupied b3. Black's next move, queen to c8 goes undetected for most of the game.

<u>Actions 12-21</u> (Figure 7): White now senses movement too. Seeing Black's interest in that side of the board, moves a pawn to e4. This helps to take command of the center, but its primary purpose is to serve as a blockade of the threat of a bishop at b7 a threat that is only suspected by White at this time, but is in fact the case. Black now becomes aware that White has pieces in motion on both sides of the board and also takes action there.

Looking to confirm his suspicion, White senses to find movement by the White-square bishop. The choice of this sensing move is a calculated risk. Sensing a6-c8 would have guaranteed precise knowledge of the bishop's location if the queen's pawn had not moved, but it was determined that knowledge of that pawn's position was also valuable to understand intent. In this case, that risk paid off.

White now pushes the king's knight forward. The intent of this move is to capture the queen, a plan that White does not realize is no longer feasible. Black's sense provides no additional information, and she continues on with her initial plan of moving bishops into position. As will be seen later, she sees the move of White's pawn to b3 as an opportunity to take the rook at a1. Had she known that the knight was blocking the path at the time, that might have changed her strategy, but it turns out to be irrelevant, as the knight is moved again before it is ever seen.

<u>Actions 22-31</u> (Figure 8): The movement of the knight to c6, a move that would be utterly ridiculous in traditional chess, is more viable here because there is no indication that Black sensed its presence at e5; and c6 is a section of the board that is unlikely to be sensed by Black because it is primarily friendly territory and does not represent an immediate threat to the king (like a knight at d6 would).

Indeed, Black's next sense move does not see it, and Black proceeds to take White's rook. Note that this move provides quite a bit of information to White, filling in at least two prior unsensed moves (the movement of the pawn at g7 and the move of the bishop to g7). On White's next sense move, his board knowledge takes a leap forward. Because of the choice of sensing, White now actually achieves perfect knowledge of Black's piece layout. The queen being at c8 is not shown, but is known with very high confidence. It is theoretically possible that the knight at b8 moved to a6, and the queen could be at b8, but the value of that combination of moves is fairly low so it is not considered likely.

At this point, the presence of the knight in its current location is very low. It would take several moves to attack e7, and the threat of Black moving the bishop at b7 is significant. White decides to move the knight to a position that is again an absurd move in traditional chess, but has strategic value here. The chances of detection at this new location (d8) are very low. There are too many pieces in the way for a queen's side castle, there is no reason to move another piece there, and it so unlikely a source of threat that Black has almost no reason to sense there.

The next sense and move actions were taken by Black for two reasons: to improve strategic position and to move the bishop away from where White knew the piece had been. White now confirmed the presence of the knight and queen being where he believed them to be. In retrospect, this was a wasted sensing move, as it really did not have the potential to provide any useful additional information. A better use of the sense would have been in the center area of the board. White's next move is to capitalize upon the unknown threat to f7. This new location is detected by Black.

<u>Actions 32-41</u> (Figure 9): Black counters by moving her queen pawn forward, providing a potential escape route for the king and enabling her queen to enter the action. This goes undetected by White, who senses to

determine whether the pawn is still at f7 and if the knight at g8 could have moved into a position to block or protect queen-to-f7. Seeing the avenue open, White attacks with his queen, capturing Black's pawn. Black senses to determine which piece took her pawn. This sense move was probably a mistake. The knowledge gained about the squares in the g column provided no useful knowledge, and would have been better used sensing d6-f8.

Believing her opponent was not sacrificing a queen, Black assumes that the queen is protected, and chooses to move her king instead. This move was probably not ideal for traditional reasons - blocking off the queens egress and opening the king to threats along the white-space diagonals. Surprised that his queen has not been taken, White senses to find out if the king is still at e8, or if not, where it has gone. White now learns that d7 had been cleared as an escape route, and that the game is not as close to being over as originally suspected. He moves his bishop to b5 in hopes of gaining a quick victory on the next turn.

Black's next sense is likely more a matter of curiosity than strategy. Black learns of the knight and that the queen has not moved, but not gained enough information to know what represents a safe or productive next move. Black moves her bishop out to threaten the White queen, likely a move of desperation to bring forces to bear on squares around the king. Unfortunately for Black, her position and board knowledge have proven insufficient. The king is being attacked from multiple angles by pieces that cannot be followed at the same time via sensing. Black cannot bring attacks to bear against the White king because all avenues of approach are blocked by pawns, and her knights are too far away. Following a sensing action to confirm that the move will go as expected, White's final move of the game is an attack that comes from a threat that was unknown to the Black player.

6. RECONNAISSANCE CHESS AS AN EXPERIMENTATION PLATFORM

The applicability and utility of reference problems (or benchmark or challenge problems) in research and development is well established. They provide a convenient, accessible, low cost, low maintenance, well-modeled, and adaptable means of producing a surrogate test article and sufficient amounts of empirical data (e.g., produce many Monte Carlo trials quickly) to test hypotheses and gain understanding about the more complex, inaccessible, and/or broadly defined problem they represent. Reference problems should be representative of important elements of the original problem but cannot and should not include every possible element and factor. Examples include the common fruit fly (drosophila) which has been used as a model organism for medical and scientific research for well over a century with applications to genetics, physiology, neurology, evolution, and many other sub-fields; the inverted pendulum which has been used for decades as a nonlinear, inherently unstable benchmark control problem for testing many types of control algorithms including those used in many aerospace and military applications; and DARPA Robotics Challenge tasks such as opening a door, climbing a ladder, and closing a valve, which are more accessible versions of the types of robotic capabilities of prime interest.

The example of primary interest here is chess, which has been used for decades as a well understood, convenient benchmark problem in artificial intelligence and machine learning. As mentioned earlier, the Kriegspiel chess variant has served as a useful experimentation platform for studying decision-making with incomplete or uncertain information. In Kriegspiel, the moves are covert and players must acquire and infer all information through moves, contact, and subsequent inferencing. Reconnaissance chess adds the element of a controllable, potentially heterogeneous sensing ensemble, which significantly enhances its utility as a reference problem for ISR sensor fusion and resource management. Reconnaissance chess is an ideal reference problem and experimentation platform for research in information valuation and collection plan optimization. The primary contributing factors are:

- The large body of literature on evaluating relative strength of chess position with full information (many techniques from a wide range of viewpoints);
- The large body of literature on optimizing chess moves (tactical decisions) with full information;
- The relatively straightforward formulation as a partially observed Markov decision process (POMDP);
- The relatively straightforward state representation which lends itself to computation of information theoretic measures of gain and loss;

- The ability to connect information gain and loss to tactical and strategic value via relative strength of position and game outcomes;
- The ability to play very large numbers of games and sub-game situations (i.e., Monte Carlo trials) to generate sample statistics that quantify the value of specific elements of information collected and the effectiveness of different techniques, algorithms, tactics, and strategies;
- The flexibility to configure sensor allocation trade-off parameters to test a range of orchestrated collection scenarios;
- The representation of concurrent missions and competing priorities via simultaneous boards (multi-chess);
- The flexibility to set multi-chess scoring schemes to test a range of strategic decision-making scenarios.

In ISR problem formulations across domains, the utility of information collection decisions is typically quantified using measures of entropic value or tactical value or some combination of both (see [31,32] for descriptions of a number of different approaches). It is clear that the tactical value of a piece of information is related to the tactical decisions that it is being used to inform. However, each tactical situation or mission is different and the various valuation techniques that have been developed tend to be contrived, ad hoc, and situation specific. The RBMC experimentation platform provides the ability to generate large numbers of trials and outcomes for statistical analysis. The Monte-Carlo functionality will enable empirically determined information valuation and algorithm assessment. A primary goal of this project is to create a community resource capable of running large numbers of experiments, generating large amounts of training data, accomodating player clients from interested parties, testing different approaches and algorithms, and supporting tournaments or other competitions and other means of collaboration.

7. CONCLUSIONS

This paper introduces the game of RBMC as a paradigm and test bed for understanding and experimenting with autonomous decision making under uncertainty and in particular managing a network of heterogeneous ISR sensors to maintain situational awareness informing tactical and strategic decision making. The intent is for RBMC to serve as a common reference or challenge problem in fusion and resource management of heterogeneous sensor ensembles across diverse mission areas. We have defined a basic rule set and a framework for creating more complex versions, developed a web-based software realization to serve as an experimentation platform, and developed some initial machine intelligence approaches to playing it. We have discovered through human versus human and human versus machine games that RBMC is very different than standard chess and offers an entirely new set of stimulating challenges as well as many surprises and a lot of fun. We invite the community to try RBMC and develop your own strategies, algorithms, and player clients to compete against ours and others'.

ACKNOWLEDGMENTS

This work was funded by the JHU/APL Research and Exploratory Development Mission Area Internal Research and Development program with contributions from the JHU/APL Precision Strike, Sea Control, National Security Space, Asymmetric Operations, and Air and Missile Defense Mission Areas. This work is affliated with the JHU/APL Intelligent Systems Center. The authors thank Shannon Dunne for participating in the illustrative game.

REFERENCES

[1] von Clausewitz, C., [*Vom Kriege (On War)*] (1832).
[2] "Chess Programming Wiki." `https://chessprogramming.wikispaces.com` (2016).
[3] Marsland, T. A. and Schaeffer, J., eds., [*Computers, Chess, and Cognition*], Springer-Verlag (1990).
[4] Shannon, C. E., "Programming a computer for playing chess," *Philosophical Magazine, Ser. 7* **41** (March 1950).

[5] Newell, A., Shaw, J. C., and Simon, H. A., "Chess-playing programs and the problem of complexity," *IBM Journal of Research and Development* **2**, 320–335 (October 1958).

[6] Newborn, M., [*Kasparov versus Deep Blue: Computer Chess Comes of Age*], Springer (1997).

[7] Silver, D., Huang, A., Maddison, C. J., and et al., "Mastering the game of Go with deep neural networks and tree search," *Nature* **529**, 484–503 (January 2016).

[8] von Neumann, J., "Zurtheorie der gesellschaftsspiele," *Mathematische Annalen* **100**(1), 295–320 (1928).

[9] Lai, M., *Giraffe: Using Deep Reinforcement Learning to Play Chess*, Master's thesis, Imperial College London (September 2015).

[10] Pritchard, D., [*Encylopedia of Chess Variants*], Everyman Chess (2001).

[11] Hearst, E. and Knott, J., [*Blindfold Chess: History, Psychology, Techniques, Champions, World Records, and Important Games*], McFarland (2013).

[12] Wetherell, C. S., Buckholtz, T. J., and Booth, K. S., "A director for Kriegspiel, a variant of chess," *The Computer Journal* **15**, 66–70 (February 1972).

[13] Li, D., [*Chess Under Uncertainty*], Premier Publishing Company (1994).

[14] Li, D., [*Chess Detective: Kriegspiel Strategies*], Premier Publishing (1995).

[15] "Kriegspiel (chess)." `https://en.wikipedia.org/wiki/Kriegspiel_(chess)` (2016).

[16] Ferguson, T., "Mate with bishop and knight in Kriegspiel," *Theoretical Computer Science* **96**, 389–403 (1992).

[17] Ciancarini, P., DallaLibera, F., and Maran, F., "Decision-making under uncertainty: A rational approach to Kriegspiel," in [*Advances in Computer Chess 8*], van den Kerik, J. and Uiterwijk, J., eds., 277–298, University of Limburg, Maastricht, The Netherlands (1997).

[18] Sakuta, M. and Iida, H., "Solving Kriegspiel-like problems: Exploiting a transposition table," *ICCA Journal* **23**(4), 218–229 (2000).

[19] Ciancarini., P. and Favini, G. P., "Representing Kriegspiel states with metapositions," in [*Proc. 20th Int. Joint Conf. on Artificial Intelligence (IJCAI07)*], (2007).

[20] Ciancarini, P. and Favini, G. P., "Monte Carlo Tree Search in Kriegspiel," *Artificial Intelligence* **174**, 670–684 (2010).

[21] Favini, G. P., "The dark side of the board: Advances in chess Kriegspiel," Tech. Rep. UBLCS-2010-06, University of Bologna, Department of Computer Science (March 2010).

[22] Bud, A. E., Albrecht, D. W., Nicholson, A. E., and Zukerman, I., "Playing invisible chess information theoretic advisors," in [*Proc. 2001 AAAI Spring Symposium on Game Theoretic and Decision Theoretic Agents*], (2001).

[23] Russell, S. and Wolfe, J., "Efficient belief-state AND-OR search with applications to Kriegspiel," in [*Proc. 2001 AAAI Spring Symposium on Game Theoretic and Decision Theoretic Agents*], (2001).

[24] Cazenave, T., "A Phantom-Go program," in [*Advances in Computer Games*], van den Herik, H. J., Hsu, S.-C., s. Hsu, T., and Donkers, H., eds., *Lecture Notes in Computer Science* **4250**, 120–125, Springer (2006).

[25] Huang, H., Ding, J., Zhang, W., and Tomlin, C., "Automation-assisted capture-the-flag: A differential game approach," *IEEE Trans. Control Systems Technology* **23** (May 2015).

[26] Bragg, S., Dao, N., Flaschen, M., and et al., "chess.js javascript chess library." `https://github.com/jhlywa/chess.js` (2016).

[27] Romstad, T., Costalba, M., and Kiiski, J., "Stockfish open source chess engine." `https://stockfishchess.org` (2008-2016).

[28] Oakman, C., "chessboard.js javascript chess board visualizer." `http://chessboardjs.com` (2013).

[29] "PyChess python chess engine." `http://www.pychess.org` (2016).

[30] "XBoard chess graphical user interface." `[https://www.gnu.org/software/xboard` (2016).

[31] Hero, A., Castanon, D., Cochran, D., and Kastella, K., eds., [*Foundations and Applications of Sensor Management*], Springer (2008).

[32] Newman, A. J. and DeSena, J. T., "Closed-Loop Collaborative ISR Resource Management," *Johns Hopkins APL Technical Digest* **31**, 183–214 (January 2013).

Figure 6. Illustrative game, actions 2-11 (sensing actions on left; move actions on right).

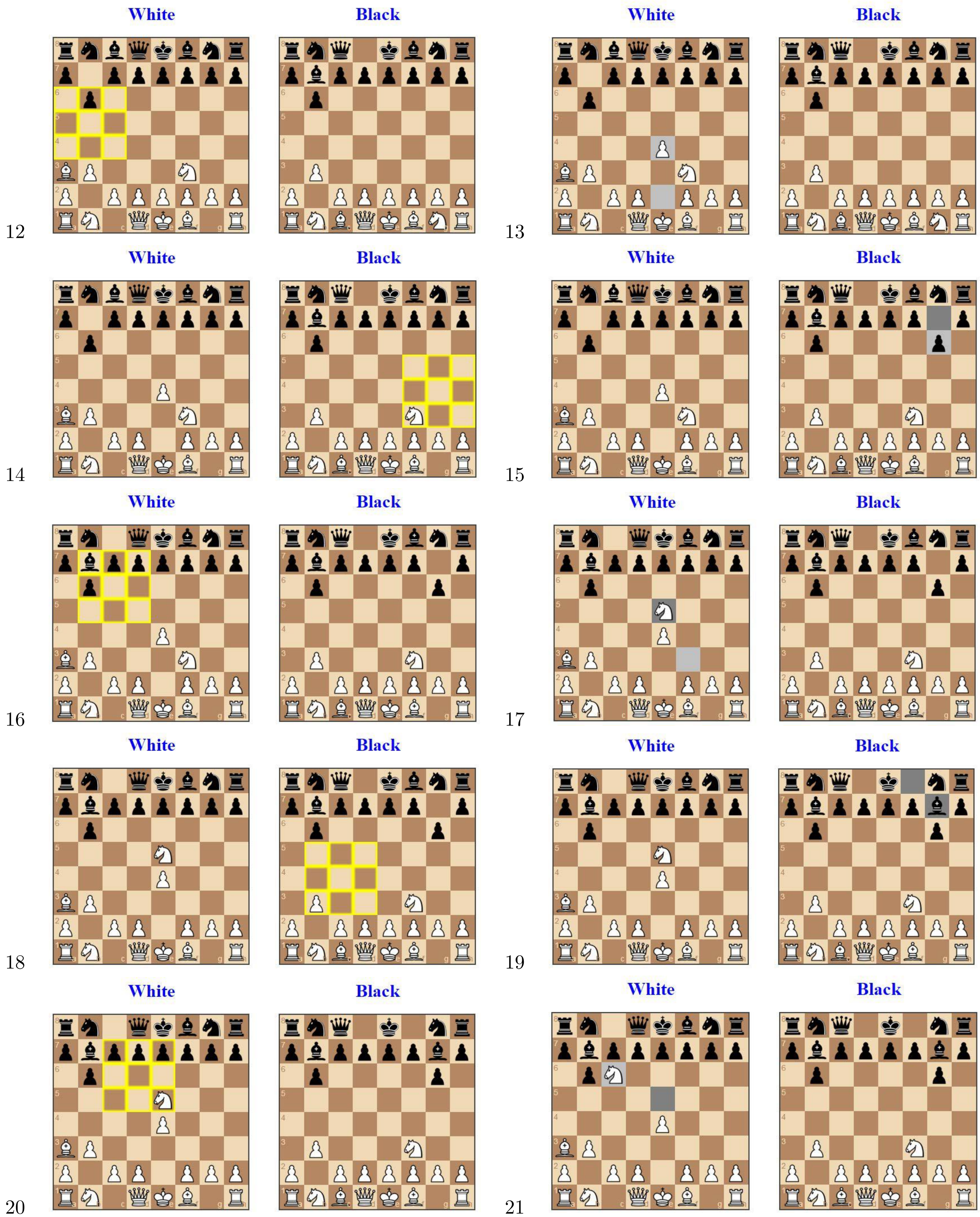

Figure 7. Illustrative game, actions 12-21 (sensing actions on left; move actions on right).

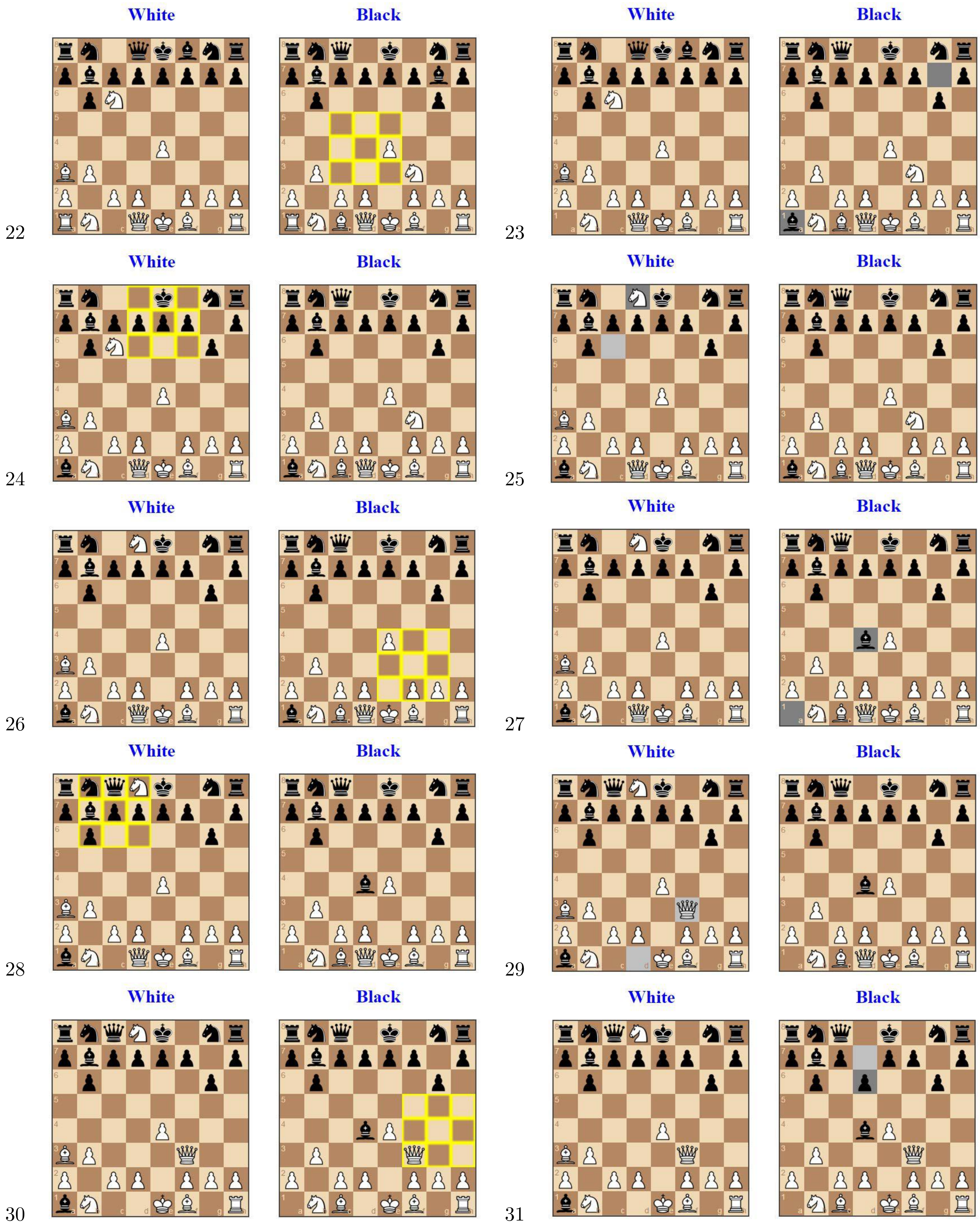

Figure 8. Illustrative game, actions 22-31 (sensing actions on left; move actions on right).

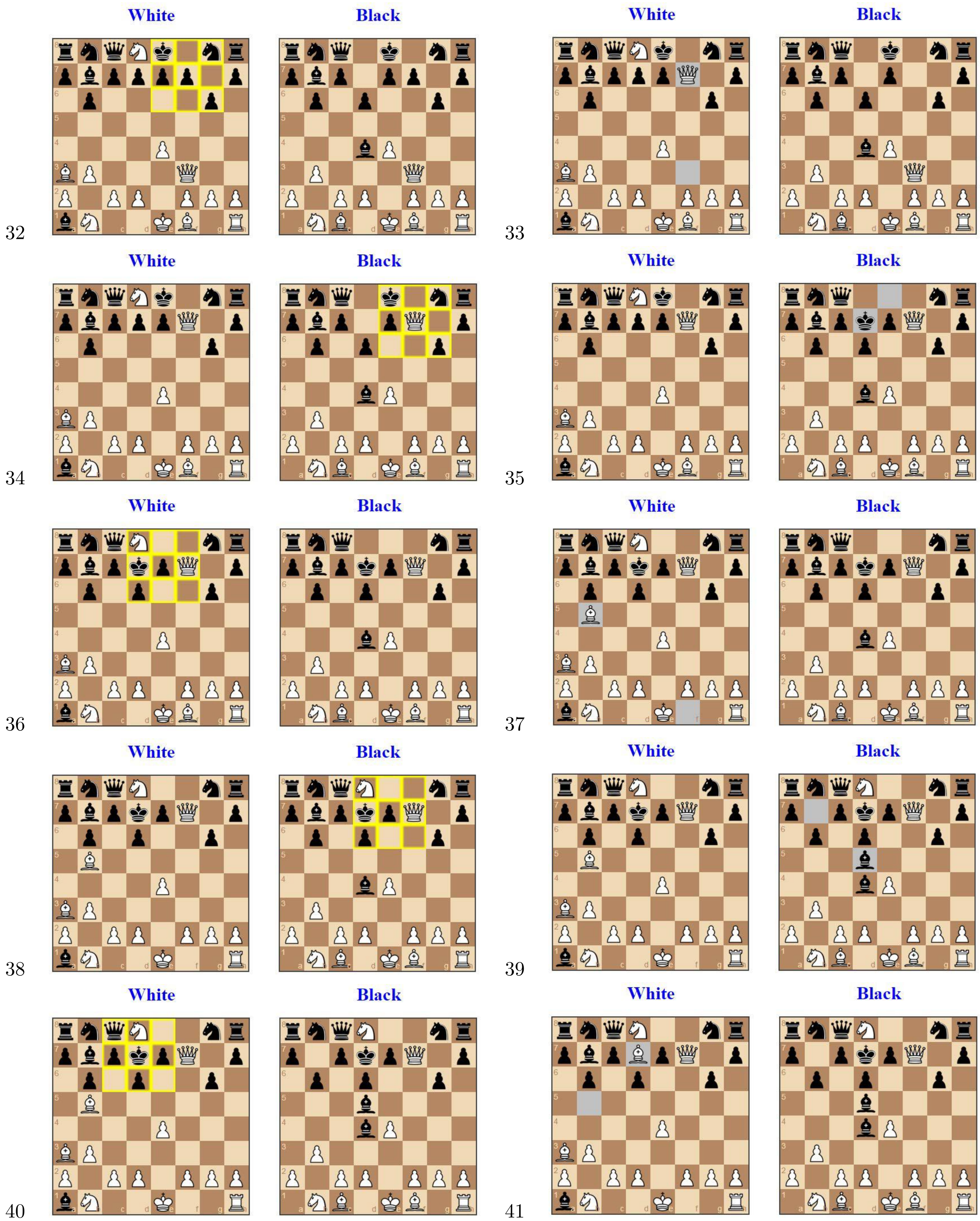

Figure 9. Illustrative game, actions 32-41 (sensing actions on left; move actions on right).