Robotic billiards: Understanding humans in order to counter them

Thomas Nierhoff, Konrad Leibrandt, Tamara Lorenz and Sandra Hirche

Abstract—Ongoing technological advances in the areas of computation, sensing and mechatronics enable robotic-based systems to interact with humans in real world. To succeed against a human in a competitive scenario, a robot must anticipate the human behavior and include it in its own planning framework. Then it can predict the next human move and counter it accordingly, thus not only achieving overall better performance but also systematically exploiting the opponent’s weak spots. Pool is used as a representative scenario to derive a model-based planning and control framework where not only the physics of the environment but also a model of the opponent is considered. By representing the game of pool as a Markov decision process and incorporating a model of the human decision making based on studies, an optimized policy is derived. This enables the robot to include the opponent’s typical game style into its tactical considerations when planning a stroke. The results are validated in simulations and real life experiments with an anthropomorphic robot playing pool against a human.

I. INTRODUCTION

As robotic technologies advance, robots are no longer only suitable for automated processes with predefined motion sequences but may accomplish tasks in dynamically changing environments and even in the interaction with humans. Seamless human-robot interaction is one of the most challenging topics in current robotic research with a tremendous application potential ranging from manufacturing via care to training/education. Interaction between humans and robots can be of different type, for example on a physical basis [1]–[3], using gestures or social interaction [4]–[6]. The same accounts for the level of interaction [7], ranging from reactive control strategies [8], [9] to learning based agents [10] enabling them to predict future actions of the interaction partner and to act accordingly. On an even higher level one is interested in the intention of a participating subject, going beyond the set of observable action towards figuring out the underlying action goals [11], either in collaborative scenarios [12]–[14] or competitive scenarios [15], [16].

Competitive games are an appealing test scenario for human-robot interaction for various reasons. On the one hand, most people encounter them on a daily basis whenever doing some kind of competitive sports. On the other hand, they provide an ideal test bed [17]–[20] as game rules are well defined and evaluation is easy - if you win, you’re better than your opponent. Due to its multi-faceted aspects combining motor performance and planning capabilities, billiards (pool) is used as a representative scenario in this work.

For mastering a competitive game its entity, different factors have to be considered. When facing a human opponent, being aware of his individual preferences and limitations can help to improve oneself’s game play. Once the weak spots of an opponent are known, they can be countered to gain a tactical advantage. At the same time a player must be aware of his own flaws to consider them properly when planning ahead. All existing pool planners, e.g. PoolFiz [21], rely on a model-based approach and use a pool simulator based on the underlying pool physics [22]–[25]. To evaluate their performance in a competitive setup, the ICGA computational pool tournament was held three times so far in ’05, ’06 and ’08. Most of the participating planners densely sample the most likely strokes, thus creating a search tree with a large branching factor in order to predict the outcome of every stroke [26], [27]. A different method is presented in [28], [29] where a gradient-based optimization is used to pocket balls and to take position play into account. Besides there exist various planners based on fuzzy logic [30], [31] and deeper theoretical considerations about the existence of Nash equilibria at pool [32]. Yet all those planners assume that both players act according to the same underlying optimality principles and do not account for individual preferences which can occur when facing a human with a pool robot [33]–[36]. In combination with the specific hardware limitations of robotic systems, the question thus becomes whether a robot is capable of overcoming its physical drawbacks through extensive planning and proper incorporation of
the opponents preferences.

The contribution of this article is the consideration of the human decision model to improve the success rate of a robotic system at a competitive game. The presented approach based on a Markov decision process is able to take both hardware limitations and a detailed model of the human decision making process into account. Thus the robot can predict the human’s actions and adapt to them by deriving an optimized policy. Due to the competitive nature of the game, the approach resembles an expectiminimax tree. Experiments are both simulation-based with thousands of games evaluated and hardware-based using a real anthropomorphic robot playing pool against a human. They show how the robot is able to improve its success rate significantly by planning ahead while incorporating the human model. To the best of our knowledge, this is the first work explicitly improving the robot’s game play at pool by considering a human-specific decision making model.

The remainder of this article is organized as follows: Sec. II presents both a detailed problem description and the general planning framework. In Sec. III and Sec. IV it is described how hardware limitations and the human model are included in the planner. Experimental results are presented in Sec. VI. Last, Sec. VII concludes with a summary and possible expansions for the future.

II. PROBLEM STATEMENT AND CONCEPTUAL APPROACH

A. Problem Statement

Following the definition in [37], “8-ball” billiards (from now on called “pool”) can be seen as a competitive interaction scenario where a robot faces a human in a turn-based manner. In order to win the game, every player has to pocket all balls of his own color and finally the black ball before the other player. At each turn, the player in charge can pocket balls by executing a stroke with the cue. A model-based approach is pursued in this article to select an optimal stroke by predicting the outcome of the next few turns. If it’s the robot’s turn, the challenge is to find an optimized stroke (action) \( A_r \) from the action set \( \mathcal{A}_r \) incorporating also the pool state \( S_t \in S \) and possible actions \( A_h \in \mathcal{A}_h \) of the human. Using the depth variable \( t \) accounting for the number of actions to plan ahead, the problem is reformulated as

\[
A_{0,r} = \arg\max_{A_r} Q(S_t, A_{t,r,(r,h)}) \quad t = 0, 1, \ldots, n
\]  

(1)

with the yet unknown cost function \( Q \) encoding the quality of the next \( n \) actions. Through proper choice of \( Q \), maximizing the term in (1) results in an optimal tactic to win the game. Note that we only optimize over \( A_{t,r} \), i.e. the actual and future action of the robot, because the strokes \( A_{t,h} \) of the human cannot be directly influenced.

Differing from simulations, the robot must consider three important aspects in real life: The first aspect, an unknown human decision making, makes it difficult to predict the executed human stroke \( A_{t,h} \) for a given situation \( S_t \). Yet this knowledge is necessary as the optimal stroke of the robot \( A_{0,r} \) also depends on the strokes of the human according to (1). Hence studies must be conducted to derive an approximate model of the human decision making to evaluate (1) precisely. The second aspect, a limited motor skill of robot and human, refers to the deviation between the planned stroke \( A^*_r \) and the executed stroke \( A^*_r \). Caused by limits in perception, uncertainties in motor control and imperfect internal models of both players, the executed stroke is in general not similar to the planned stroke. This turns the deterministic game in simulation into a probabilistic game in real-world situations. When representing both factors well enough with a probability distribution \( f_n \), it is modeled as

\[
A_{t,(r,h)} = f_n(\{A^*_{r,(r,h)}\}).
\]  

(2)

The last aspect, robot kinematic constraints, limits the space of actions \( \mathcal{A}_r \) of the robot as

\[
\mathcal{A}_r \subset \mathcal{A}.
\]  

(3)

As the human kinematic constraints are negligible, it is assumed that \( \mathcal{A}_h = \mathcal{A} \).

All three aspects must be quantified and represented by precise models for incorporating them in (1). Focusing in this article on the human behavior, it is investigated how well the human decision making can be modeled and to what extent a good model improves the win rate of the robot.

B. General Framework

This section develops a mathematical framework using a Markov decision process (MDP) to model the game of pool in its entity and consider all aspects listed in Sec. II-A. Pool can be represented as a sequential stochastic game with a continuous action space and a continuous state space. It is sequential due to the turn-based stroke execution switching between both players and stochastic due to the nondeterministic outcome of every stroke caused by the limited skill of each player. Differing from games with a finite set of states like chess or backgammon, there is an infinite number of possible combinations for the balls to be placed on the table. The same holds for the actions. When discretizing the state and action space, the game of pool is modeled
as a MAXPROB Markov decision process (MAXPROB MDP), see [38]. It is based upon a standard MDP defined by tuples of the form $\langle S, A, T, R, G, s_0 \rangle$ where $S \in S$ is the set of states (position of all balls on the table plus current game situation determined by the rules), $A \in A$ is the set of actions (strokes), $T \in T$ is a transition function $S \times A \times S \rightarrow [0; 1]$ denoting the probability of moving from state $S_i$ to $S_j$ by executing the action $A$, $R \in R$ is a mapping $S \times A \rightarrow \mathbb{R}$ specifying action rewards and $s_0$ is the start state.

For a MDP, the general goal is to find an optimal policy $\pi^*$ maximizing the expected reward when traversing the states of the MPD. By introducing the discount factor $\gamma \in [0; 1]$ and the value function $V : S \rightarrow \mathbb{R}$, the value function $V^\pi(s)$ denoting the expected reward for the policy $\pi$ starting in state $S$ is defined as

$$V^\pi(S) = \mathbb{E}^\pi \left[ \sum_{t=0}^\infty \gamma^t R(S_t, A_t) \right]_{S_0 = S}.$$  \hfill (4)

In addition, the action-value function $Q : S \times A \rightarrow \mathbb{R}$ in (1) is formalized as

$$Q^\pi(S, A) = \mathbb{E}^\pi \left[ \sum_{t=0}^\infty \gamma^t R(S_t, A_t) \right]_{S_0 = S, A_0 = A}.$$  \hfill (5)

The optimal policy $\pi^*$ is the policy that maximizes the value function $V^\pi$ for all possible policies $\pi$ as

$$\pi^* = \arg \max_{\pi} V^\pi,$$  \hfill (6)

resulting in the optimal value function $V^*(S)$ and action-value function $Q^*(S, A)$. Both value function and action-value function are closely related, as such the optimal value function $V^*(S)$ is obtained by finding the action $A$ that maximizes the optimal action-value function $Q^*(S, A)$ as

$$V^*(S) = \max_A Q^*(S, A)$$  \hfill (7)

Note that the used formulation is consistent with existing literature on the topic, e.g. [39].

MAXPROB MDPs are a special class of MDPs with the goal of maximizing the probability of reaching a goal state; that is the probability of winning a game. They are based upon Stochastic Shortest Path MDP (SSP MDP) introducing a set of absorbing goal states $G \in G$. Such a goal state is defined by $T(G, A, G) = 1, \forall A \in A$ and $R(G, A) = 0, \forall A \in A$, allowing only self-transitions in $G$ and accumulating no reward. The only two conditions for SSP MDPs are that for every state $S$ at least one proper policy exists that reaches a goal state with probability $P = 1$. Additionally, every improper policy has a reward of $-\infty$. For pool, SSP MDPs are still not applicable as they only feature goal states (robot wins) but no dead end states (human wins). In case of dead end states, it is not guaranteed anymore that a goal state can be reached from any other state with probability $P = 1$. MAXPROB MDPs on the other hand have a similar structure as SSP MDPs but explicitly account for dead ends by judging the quality of every policy not based on the expected reward but based on the probability of eventually reaching a goal state. By assigning a reward $R = 1$ to every action that reaches a goal state and $R = 0$ to all other actions, the resulting optimal value function $V^*(S)$ reflects the probability of reaching a goal state (e.g. winning the game) when starting in state $S$.

In theory, an optimal policy can be obtained then for an arbitrary start state $S_0$ using heuristic search algorithms [40], [41]. Unfortunately, due to the large state space, those algorithms cannot be used for pool. Instead one has to use approximation algorithms resulting in a near-optimal value function $V^* \leq V^*$ by evaluating only the most promising subset of state-action pairs. In general, those algorithms turn the probabilistic MDP into a deterministic planning problem resulting in a search tree with $S_0$ being the root and solve it using a deterministic planner [42].

Having obtained a policy, an individual action-value function can be assigned to each stroke specifying the expected reward. Then a player simply has to execute the stroke with the highest expected reward. A method to find an action-value function that considers also the human motor skill and decision making is developed in the following sections.

III. Modeling Human/Robot Motor Skill and Robot Kinematic Constraints

As explained in Sec. II-A, the limited motor skills of both players let the executed stroke differ from the desired or planned stroke, thus requiring a measure to describe the percentage of success of a desired stroke. This is described by the pocket probability which incorporates the limited motor skill in the presented planning framework. As it is based upon the stroke difficulty measure, the latter is introduced first.

A. Measure of Stroke Difficulty

The expected reward in (4) is closely related to the stroke difficulty. When playing pool, the expected reward depends on the number of pocketed balls. Intuitively the more difficult it is to pocket a ball, the lower the reward is. When executing a stroke, the player must not hit the ball he wants to pocket (called “object ball”) directly.
Instead, he has to hit the white ball (called “cue ball”) with the cue. From a mathematical perspective, a stroke can be described by five parameters $\alpha, \beta, \gamma, \theta, p$ [43] for a given 2D cue ball position on a flat table. Stroke intensity $p$ and stroke angle $\theta$ have to be considered even by novice players in order to pocket balls. On the other hand, the angles $\alpha, \beta$ and $\sigma$ are mainly varied by advanced players to induce spin on the cue ball for a better position play. A central assumption for the remainder of the article is a well defined spin during the stroke. As such the cue ball is supposed to be hit centrally and the article is a well defined spin during the stroke. As a result, the parameter space can be reduced from five to two dimensions ($\theta$ and $p$).

Different measures are used in literature to define the stroke difficulty. For direct shots where the cue ball hits the object ball without any other ball-ball or ball-cushion collisions, the stroke difficulty can be calculated using basic geometry. Assuming a negligible spin transfer during ball-ball collisions, the stroke intensity $p$ and stroke angle $\theta$ being independent of each other and the stroke intensity $p$ being large enough to pocket a ball, the difficulty of a direct shot is solely specified by the range of values $[\theta_1; \theta_u]$, see [44]. The value $\theta_m = 0.5(\theta_1 + \theta_u)$ denotes the mean angle for which the object ball is pocketed in the middle of the pocket. Intuitively, the smaller the interval $[\theta_1; \theta_u]$ or allowed angular deviation (AAD) becomes the more difficult the stroke will be, see Fig. 2 left side. The same technique can also be used for combination shots including more than one object ball.

Another method to determine the stroke difficulty is based upon expert knowledge [45], [46]. It reveals that the human perception of stroke difficulty depends only on a few parameters: The distance $d_1$ between cue ball and object ball, the distance $d_2$ between object ball and pocket and the cutting angle $\theta_c$, see Fig. 2 right side. Various approaches in literature try to determine an expression for the difficulty of a stroke based on this knowledge [26], [30], one of the most prominent ones is [28]

$$\kappa = \cos(\theta_c).$$

(8)

Analogously to the AAD stroke difficulty, a small value corresponds to a tricky stroke, a large value represents an easy one. The AAD stroke difficulty can be linked in a straightforward fashion to the pocket probability, implicitly considering the parameters $\theta_c, d_1, d_2$ and partially blocked paths due to other balls on the table. The expert stroke difficulty on the other hand suggests that the human perception of difficulty deviates from the AAD stroke difficulty due to a different perspective, thus requiring a new difficulty measure when modeling the human decision making.

Robot kinematic constraints can be incorporated using the AAD stroke difficulty. It is assumed that robot kinematic constraints limit the range of admissible $\theta$-values to the interval $[\theta^k_1; \theta^k_u]$. Then the resulting interval of stroke angles that are both executable by the robot and pocket a ball is given by the intersection $[\theta_1; \theta_u] \cap [\theta^k_1; \theta^k_u]$. Fig. 3 illustrates the influence of the robot kinematic constraints. Shown is a simplified, yet true-to-scale collision model of the used pool robot and the pool table, approximated by bounding boxes. The robot is limited by two constraints. First, it has a stiff body (dark gray rectangle in Fig. 3) which collides with the table if the white ball is placed too far away from any cushion. Second, its right endeffector does not lie on the table but on the cushion, which causes problems if the white ball is placed too close to the cushion. The angle $\theta_r$ specifies the range of possible stroke angles for a given position of the white ball on the table. An angle $\theta_r = 2\pi$ means the white ball can be shot in any direction whereas an angle $\theta_r = 0$ stands for an unreachable white ball. Moreover do gray circles mark the intervals $[\theta^k_1; \theta^k_u]$ of...
stroke angles for specific positions of the white ball.

Figure 3: Kinematic constraints of the robotic system. Due to collisions with the pool table, the robot cannot execute a stroke in an arbitrary direction for a given ball position. The brighter the area is, the larger the range of possible stroke directions is. Gray circle segments mark the possible stroke direction for certain positions of the white ball.

B. Measure of Pocket Probability

This section derives the pocket probability $P_s \in [0; 1]$. It links the stroke difficulty presented in Sec. III-A to the individual motor skill of each player as described mathematically in (2).

It is assumed that the stroke precision of robot and human is represented by normal distributions for the stroke intensity $p$ and stroke angle $\theta$ as $N_{p,\{r,h\}}(\mu_p, \sigma^2_{p,\{r,h\}})$ and $N_{\theta,\{r,h\}}(\mu_\theta, \sigma^2_{\theta,\{r,h\}})$. Here, $\mu_p$ and $\mu_\theta$ encode the desired (planned) stroke $A^*$, the variances $\sigma^2_{p,\{r,h\}}$ and $\sigma^2_{\theta,\{r,h\}}$ represent the individual skill level of robot and human: Novice players are characterized by large variances whereas for professionals the variance is close to zero. Equation (2) is thus reformulated using the 2D normal distribution $N_{\{r,h\}}$ as

$$A = f_r(A^*) = N_{\{r,h\}}(\mu, \Sigma),$$

with $\mu = [\mu_p, \mu_\theta]^T$, $\Sigma = \text{diag}(\sigma^2_{p,\{r,h\}}, \sigma^2_{\theta,\{r,h\}})$.

The limited motor skill of human and robot is encoded by the variances $\sigma^2_{p,\{r,h\}}, \sigma^2_{\theta,\{r,h\}}$, see [44]. When defining the stroke difficulty using the AAD method, the stroke difficulty can be converted for every stroke into a pocket probability $P_s \in [0; 1]$. Similar to the stroke angle, we assume that a ball will be pocketed for stroke intensities within the interval $[p_l; p_u]$. For stroke intensities $p < p_l$ the ball is not fast enough to be pocketed and values $p > p_u$ cannot be achieved due to a limited player strength. With $f_{\{r,h\}}$ as the probability density function of $N_{\{r,h\}}$, the pocket probability can be described by

$$P_s = \int_{p_l}^{p_u} \int_{\theta_l}^{\theta_u} f_{\{r,h\}}(d\theta dp. \quad (10)$$

We further assume that $\sigma_p \ll p_u - p_l, p_l \ll \mu_p \ll p_u$, i.e. the pocket probability is mostly unaffected by the choice of $\mu_p$ and depends primarily on the stroke angle $\theta$. Then (10) is approximated with $f_{\theta,\{r,h\}}$ as the probability density function of $N_{\theta,\{r,h\}}$ by

$$P_s \approx \int_{\theta_l}^{\theta_u} f_{\theta,\{r,h\}}(d\theta, \quad (11)$$

The formulas also hold for combination shots but must be extended if two or more object balls shall be pocketed with a single stroke.

IV. MODELING HUMAN DECISION MAKING

This section focuses on the human decision making to include it in the general planning framework. Differing from a robot for which the assumptions in Sec. III hold and where the pocket probability can be calculated based upon the AAD stroke difficulty, the expert stroke difficulty indicates that humans select their stroke based on different criteria. When modeling human decision making, three subproblems are identified:

1) Compute the human stroke difficulty $\kappa_h$: How difficult a human thinks it is to pocket a ball.
2) Compute the human pocket probability $P_h$: How difficult it is for a human to pocket a ball.
3) Compute the human discount factor $\gamma_h$: How far a human is planning ahead.

Differing from a rationally acting robot who will pocket the ball with the highest action-value function, unknown human likings or limitations may favor a different stroke. As it is also unknown how these preferences affect the pocket probability, one has to specify not just the human stroke difficulty $\kappa_h$ but also the human pocket probability $P_h(\kappa_h)$ as a function of the human stroke difficulty, see [47].

1) Human Stroke Difficulty: In this section an algebraic expression of the human stroke difficulty $\kappa_h$ based on two psychological experiments is derived. Similar to the stroke difficulty in Sec. III-A it measures the perceived human difficulty of each stroke and allows to rank them accordingly. In the first experiment, the most influential factors on the human stroke difficulty are figured out. The result is then used in combination with the second experiment to determine a precise model of the human stroke difficulty.

For the first experiment, 25 participants (15 male, 21-31 yrs, 025 yrs) are shown 24 pool scenarios on a real pool table representing a wide variety of possible strokes and AAD values. In every scenario two playable object balls are placed on the table. For 18 scenarios the AAD stroke
difficulty is similar for the two object balls whereas the three parameters \( d_1, d_2 \) and \( \theta_e \) are varied. For the other 6 scenarios, one of the two object balls is easier to pocket according to the AAD stroke difficulty, but either more difficult to reach (the player has to lean more over the pool table) or partially blocked by a third ball, reducing the interval of admissible stroke angles \([\theta_l; \theta_u]\). Each participant has to judge which ball seems easier to pocket and is asked to specify freely one or more reasons for this decision which are categorized afterwards. The most frequently stated reasons with their number of occurrence for the first 18 scenarios are shown in Table I. The answers show that the variables \( d_1, d_2 \) and include the term \( \cos(\theta_e) \) to account for the increasing difficulty at high cutting angles. The last term is an extension of the existing measure (8). They are

\[
\begin{align*}
\kappa_1 &= a_1 \cos^4(d_1) + a_2 \cos^2(d_2) + a_3 \cos^2(\theta_e) \\
\kappa_2 &= a_1 \cos^4(d_1) + a_2 \cos^2(d_2) + a_3 \cos^2(\theta_e) + \kappa_p, \\
\kappa_3 &= a_1 \cos^4(d_1) + a_2 \cos^2(d_2) + a_3 \cos(\theta_e) \cos(\theta_e) + \kappa_p, \\
\kappa_4 &= a_1 \cos^4(d_1) + a_2 \cos^2(d_2) + a_3 \cos(\theta_e) \cos(\theta_e) + \kappa_p, \\
\kappa_5 &= \frac{\cos(\theta_e) \cos(\theta_e)}{d_1^2(d_1) d_2^2(d_2)},
\end{align*}
\]

with \( a = [a_1, a_2, a_3] \) as constant coefficients and \( c = [c_1, c_2, c_3] \) with \( c_1, c_2 \) as constant coefficients and \( c_3 \) as polynomial of degree one. A two-staged optimization is used to find optimized values for \( a \) and \( c \). The cost function \( \Gamma_1 \) of the first stage consists of two terms related to the first and second psychological experiment. Minimizing the first term corresponds to matching the predicted decision \( \tilde{\Delta}_j(a, c) \) to the dominant decision \( \Delta_j \) of participants of in the first experiments. The factor \( \delta_j \) acts as a penalty term if the predicted decision does not match the decision from the first experiment. By minimizing the second term, one matches the predicted relative ranking \( \hat{\alpha}_k(a, c) \) to the relative rank \( \alpha_k \) of each scenario in the second experiment.

\[
\Gamma_1 = \sum_{j=1}^{18} \delta_j(a, c)^2 + \sum_{k=1}^{12} |\hat{\alpha}_k(a, c) - \alpha_k|^2,
\]

where \( \delta_j = \begin{cases} 2 & \text{if } \tilde{\Delta}_j(a, c) \neq \Delta_j, \\ 0 & \text{else}, \end{cases} \)  

The cost function \( \Gamma_2 \) of the second stage is a measure how much the predicted absolute ranking \( \hat{\kappa}_k(a, c) \) of each scenario in the second experiment differs from the absolute ranking \( \beta_k \). To suppress outliers, the 25% trimmed mean \( \beta_{k,25} \) is used instead of \( \beta_k \). By minimizing the function \( \Gamma_2 \), a measure of the resulting stroke difficulty is obtained.

\[
\Gamma_2 = \sum_{k=1}^{12} |\hat{\kappa}_k(a, c) - \beta_{k,25}|^2,
\]

Final results are displayed in table II. Because the function \( \kappa_5 \) has the lowest cost \( \Gamma_2 \), it is the best approximation to the real human stroke difficulty. Thus the human stroke difficulty \( \kappa_h \) is determined as

\[
\kappa_h = \frac{\cos(\theta_e) \cos(\theta_e)}{d_1^{0.33} d_2^{0.38}}.
\]
2) Human Pocket Probability: This section develops a model of the human pocket probability $P_h \in [0; 1]$ to measure the pocket probability of the human player depending on the human stroke difficulty as

$$P_h = f_p(\kappa_h), \quad (16)$$

thus linking the two measures in a similar way to the AAD stroke difficulty and the pocket probability. To model $P_h$ precisely over a wide variety of $(\theta, d_1, d_2)$-triplets, a Monte-Carlo approach is used that evaluates random strokes occurring during a series of conducted pool games.

In total, four participants (3 male, 24-36yrs, $\bar{\theta} 29$yrs) took part in the third experiment. Two players are intermediate amateurs who frequently play a few pool games per week, two are novice players with hardly any experience. Participants were grouped into dyads of equal playing skill. Their task was to play multiple games while sticking to direct strokes and hitting the cue ball centrally. The stroke results are split into eleven intervals depending on their stroke difficulty. For comparing the influence of AAD stroke difficulty, expert stroke difficulty and human stroke difficulty on the human pocket probability, all measures are normalized to the $[0; 1]$ interval. For every interval, the mean pocket probability and mean stroke difficulty are calculated. Pearson’s correlation coefficient between stroke difficulty and pocket probability for the three stroke difficulty measures is shown in table III. For small values of $d_1, d_2$

Table III: Correlation coefficient between stroke difficulty and pocket probability when considering all strokes

<table>
<thead>
<tr>
<th>player</th>
<th>evaluated strokes</th>
<th>correlation coefficient $\kappa$</th>
<th>AAD $\kappa_h$</th>
<th>$\kappa_h$</th>
</tr>
</thead>
<tbody>
<tr>
<td>one</td>
<td>216</td>
<td>0.46</td>
<td>0.61</td>
<td>0.89</td>
</tr>
<tr>
<td>two</td>
<td>182</td>
<td>0.65</td>
<td>0.72</td>
<td>0.86</td>
</tr>
<tr>
<td>three</td>
<td>164</td>
<td>0.54</td>
<td>0.58</td>
<td>0.88</td>
</tr>
<tr>
<td>four</td>
<td>152</td>
<td>0.76</td>
<td>0.68</td>
<td>0.92</td>
</tr>
</tbody>
</table>

Table IV: Correlation coefficients between stroke difficulty and pocket probability when neglecting very easy strokes

between $P_h$ and $\kappa_h$ can be determined individually for each player. For player one it becomes

$$P_h = f_p(\kappa_h) = \begin{cases} 
0.36\kappa_h + 0.27 & \forall \kappa_h \in [0; 2.02], \\
1 & \forall \kappa_h \in [2.02; \infty].
\end{cases} \quad (17)$$

3) Human Discount Factor: So far only the humans’ skill is discussed, specifying how well they can pocket balls. Another aspect are planning capabilities, defining how much they think about future states when planning their next stroke. Similar to (4) it is modeled by a discounted reward factor $\gamma_h$. Whereas a small $\gamma_h$ value

Figure 4: Plotted results of the normalized stroke difficulty vs. pocket probability for player one with corresponding regression lines.

(a) All strokes considered. (b) Without easy strokes.
implies that the player focuses more on the current situation, a large discount factor $\gamma_h$ indicates that he also takes future table states into account. The data from the third experiment can be used to compute an individual discount factor $\gamma_h$ for every participant by a maximum likelihood estimation (MLE) maximizing the percentage of correctly predicted stroke decisions $\Psi$ for every player over the parameter $\gamma_h$ as

$$\hat{\gamma}_h = \arg\max_\gamma \mathcal{L}(\gamma_h | \Psi)$$  \hspace{1cm} (18)

Results of the MLE for two dyads for a planning depth of 1 are displayed in Fig. 5 and [47]. While the intermediate player attested to think about the outcome of a stroke ($\gamma_h > 0$), the novice player confirmed that he solely focuses on the easiest stroke ($\gamma_h = 0$). This corresponds to the displayed results, with $\gamma_h \approx 0.55$ for the intermediate player and $\gamma_h \approx 0$ for the novice player.

Figure 5: Maximum likelihood estimation of the human discount factor $\gamma_h$ for a novice player and an intermediate player. Only the maximum of each plot is of interest. A discount factor $\gamma_h = 0$ stands for a player not planning at all whereas a discount factor $\gamma_h = 1$ accounts for good planning capabilities.

V. POOL-SPECIFIC ADAPTATION

This section describes how the general framework presented in Sec. II-B is modified and combined with the stroke difficulty and pocket probability measures in Sec. IV and IV to fit pool-specific needs. The two main challenges that must be overcome are a high computational complexity when solving the MAXPROB MDP and the derivation of an optimized policy.

Solving the MAXPROB MDP involves the creation of a search tree. Expanding the search tree means to run a pool simulation that predicts the outcome of a stroke for a given state on the table. Yet running a pool simulation is a time-costly operation and as such an approximate scheme must be used that limits the number of pool simulations. A first step is to simulate only the next $n$ actions, resulting in an approximate value function $\hat{V}^\pi$ as

$$\hat{V}^\pi(S) = \mathbb{E} \left[ \sum_{t=0}^{n} \gamma^t R(S_t, A_{t,\{r,h\}}) \right]_{S_0 = S}$$  \hspace{1cm} (19)

and approximate action-value function

$$\hat{Q}^\pi(S, A) = \mathbb{E}^\pi \left[ \sum_{t=0}^{n} \gamma^t R(S_t, A_{t,\{r,h\}}) \right]_{S_0 = S, A_0 = A}$$  \hspace{1cm} (20)

Yet this causes other problems as the for MAXPROB MDPs a reward $R \neq 0$ is only assigned whenever a goal state is reached, i.e. a game is won. Thus in theory every branch of the tree must be expanded until reaching a goal state or a known dead end to evaluate its outcome, which however may take more than $n$ actions. This is overcome by introducing an reward $R_\eta$ as

$$R_\eta = \begin{cases} +1 & \text{for every pocketed ball of the robot,} \\ -1 & \text{for every pocketed ball of the human.} \end{cases}$$  \hspace{1cm} (21)

which helps to estimate the outcome of an action whenever the first ball gets pocketed. This serves as a bias of the planner towards strokes for which most balls of the own color and least balls of the opponent’s color are pocketed. In addition, the original condition of a reward $R = 1$ when winning a game is relaxed to

$$R_w = \begin{cases} +w & \text{if robot wins,} \\ -w & \text{if human wins,} \end{cases}$$  \hspace{1cm} (22)

with constant $w$. The combined reward $R$ is defined as

$$R = R_\eta + R_w.$$  \hspace{1cm} (23)

The constant $w$ weights the importance of pocketing a ball with respect to a win/loss of the game. A high $w$-value results in an aggressive game style if the planner sees a chance to win the game and a defensive game style if there is a chance for the opponent to win the game. The chosen reward $R$ does have a physical meaning as for $w = 0$ and $\gamma = 1$ the value function $V^\pi(S)$ represents the expected number of balls that the robot will pocket more than the human over the next $n$ strokes following the policy $\pi$. Similarly, the action-value function $Q^\pi(S, A)$ denotes the difference of pocketed balls for a given policy when executing the stroke $A$. During tree creation, pruning is used to reduce computational complexity further: Only the most promising strokes are simulated, i.e. the ones that aim at pocketing a ball of the own color with a direct shot. The combination of all three methods leads to a considerable reduction of computational complexity as it is sufficient
to simulate only a subset of all possible strokes over the next \( n \) turns.

Once a search tree is created, an optimized policy must be derived for computing the next stroke. Backward induction is used to iteratively evaluate the value function \( \hat{V}(S_t) \) backwards in time to find \( \hat{V}(S_{t-1}) \) and the optimized stroke parameters for each stroke. It is started by calculating the action-value function for every leaf node of the search tree of depth \( n \). This is the reward \( R \) weighted with its pocket probability as

\[
\hat{Q}^\pi(S_n, A_n) = \begin{cases} P_{s,i}R_n & \text{if robot's turn,} \\ P_{h,i}R_n & \text{if human's turn.} \end{cases} \tag{24}
\]

In (24) the variable \( P_{s,i} \) stands for the pocket probability of the \( i \)-th ball-pocket combination according to (11) using the AAD stroke difficulty and \( P_{h,i} \) for the human pocket probability according to (16). Based on \( \hat{Q}^\pi(S_n, A_n) \), the value function \( \hat{V}^\pi(S_n) \) is calculated as

\[
\hat{V}^\pi(S_n) = \begin{cases} \max_i P_{s,i}R_n & \text{if r's turn,} \\ \min_i P_{h,i}R_n & \text{if h's turn.} \end{cases} \tag{25}
\]

The value function is split up into two cases, accounting for the competitive nature of the game. As such the robot tries to maximize the value function, whereas the human has to minimize it. During backward induction, the action-value function \( \hat{Q}^\pi(S_{t-1}, A_{t-1}) \) is calculated in an intermediate step to derive \( \hat{V}^\pi(S_{t-1}) \). Because the outcome of each stroke is nondeterministic due to the limited motor skill of each player, it has to be weighted with the probability density function \( f_{\{r,h\}} \), see (10).

This results in

\[
\hat{Q}^\pi(S_{t-1}, A_{t-1}) = \mathbb{E}^\pi \left[ R_{t-1} + \gamma \hat{V}^\pi(S_t) \right]
\]

\[
= \begin{cases} \int \int (R_{t-1} + \gamma_r \hat{V}^\pi(S_t))f_r dp d\theta & \text{if r's turn,} \\ \int \int (R_{t-1} + \gamma_h \hat{V}^\pi(S_t))f_h dp d\theta & \text{if h’s turn.} \end{cases} \tag{26}
\]

with \( \gamma_r \) as the discount factor of the robot and \( \gamma_h \) as the human discount factor according to (18). Note that the integrals in (26) can only be approximated due to a discrete search tree. The value function \( \hat{V}^\pi(S_{t-1}) \) is then calculated as

\[
\hat{V}^\pi(S_{t-1}) = \text{optimize } \hat{Q}^\pi(S_{t-1}, A_{t-1})
\]

\[
= \begin{cases} \max_{\mu_r, \theta_r} \hat{Q}^\pi(S_t, A_t) & \text{if r’s turn,} \\ \min_{\mu_h, \theta_h} \hat{Q}^\pi(S_t, A_t) & \text{if h’s turn.} \end{cases} \tag{27}
\]

Having obtained \( \hat{V}^\pi(S_0) \), the optimized stroke parameters \( \mu_r, \theta_r \) for the actual stroke can be directly derived as they are the ones optimizing \( \hat{V}^\pi(S_0) \). The framework can be seen as an expectiminimax tree [48] with an additional optimization step for each node to calculate the optimal reward.

VI. Evaluation

The evaluation of the planning algorithm is both based upon simulations and by using an anthropomorphic robot facing a human opponent on a real pool table. Simulations allow for a fast evaluation of a large number of strokes to measure the individual influence of various aspects. In contrast, the real world scenario demonstrates the validity of the entire framework. For both experiments the pool simulator described in [44] is used.

The individual parameters of robot and human in table V are determined through hundreds of analyzed strokes recorded by a ceiling camera, indicating that the human opponent outperforms the robot with respect to motor skill (smaller \( \sigma_h, \sigma_\theta \)) but is inferior to the robot concerning planning capabilities (smaller \( \gamma \)).

<table>
<thead>
<tr>
<th>parameter</th>
<th>human</th>
<th>robot</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sigma_h )</td>
<td>0.01</td>
<td>0.03</td>
</tr>
<tr>
<td>( \sigma_\theta )</td>
<td>0.007</td>
<td>0.012</td>
</tr>
<tr>
<td>( \gamma )</td>
<td>0.75</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table V: Parameters for the experimental evaluation

A. Simulations

Simulations make it possible to estimate the individual influence of a variety of factors without the need to perform time-consuming experiments. For all simulations a pool simulator programmed in C++\Qt\ is used [44], whereas the planning framework according to Sec. V is implemented in Matlab R2013b.

A first simulation aims at displaying the influence of a varied search depth \( n \), see (24). As discussed in [27], the effectiveness of a search depth of 2 is still an unsolved question. Differing from deterministic board games like chess or backgammon where the search depth is among the most influencing factors, the probabilistic nature of pool makes it more difficult to answer the question. In [27] the authors find out that both a search depth of 2 and a more densely sampled action space of depth 1 have a similar influence on the percentage to win. In our simulation, a total of 200 strokes for random pool states is processed with a search depth of 0, 1 and 2. The first simulation scenario is similar to the experiment in [27] as it neither includes robot kinematic constraints nor the human model (we use (11) instead of (16) to model the human decision process). In contrast, both the human model and robot kinematic constraint are
incorporated in the second scenario. Simulation results are shown in Venn diagrams in Fig. 6. As such for 17 strokes the planning results for a depth of 0 and 1 are similar. Comparable to [27], most of the strokes - 105 for the first scenario, 77 for the second scenario - are similar for different search depths and a varied search depth influences the decision in just about 20% of all cases. This indicates that a search depth of 2 has only a minor influence on the decision process and can be probably overcome by a more densely sampled action space of depth 1.

![Venn diagrams](image)

Figure 6: Effect of a varied planning depth based upon 200 randomly selected table states using Venn diagrams.

The next simulation illustrates the effect of incorporating the human decision process in the planning framework. Three different runs are performed, depending on the way the robot kinematic constraints of Sec. III-A are treated. For the “without repositioning” run, whenever the robot is unable to reach a ball, it is considered as a foul. In the “with repositioning” run, the white ball is slightly moved such that the robot is able to execute a legal stroke whenever the robot is unable to reach a ball. Last, in the “without constraints” run there are no kinematic constraints, i.e. the robot is able to reach any position on the table. Every run consists of two scenarios evaluating the effect of the human decision process when planning ahead. For every scenario 1200 games are evaluated twice, both with robot and human as starting player. When considering the human decision process, it is modeled according to Sec. IV. Otherwise equation (11) is used instead of (16) to model the human pocket probability. The number of robot wins of every run are checked for significance using a two-sample t-test. Results are shown in Fig. 7. It is visible that for all runs the inclusion of the human decision process significantly improves the win rate.

![Table of results](image)

Figure 7: Number of robot wins for different scenarios after 2400 games played in simulation. Right side: Significance level of every comparison using a two-sample t-test.

The last simulation displays the necessity to include a precise model of the robot’s kinematic constraints as shown in Sec. III in the planner and the effect of a varied search depth of the robot. Similar to the previous simulation, 1200 games are evaluated twice. Two different runs are considered. For both runs it is considered as a foul whenever the robot is unable to reach a ball during stroke execution. The difference is that no kinematic constraints are considered when planning ahead for the “deceived repositioning” run, whereas kinematic constraints are considered when planning ahead for the “without repositioning” run. On can see in Fig. 8 that a search depth of 1 significantly improves the robot’s win rate. Moreover, considering the robot’s kinematic constraints when planning ahead increases the win rate by around 20-25%.

![Table of results](image)

Figure 8: Number of robot wins for different planning depths after 2400 games played in simulation. Right side: Significance level of every comparison using a two-sample t-test.

B. Experimental Evaluation with Robotic Platform

The hardware setup for the experimental evaluation consists of a robot with a pair of 7-DoF anthropomorphic arms mounted on an omnidirectional platform, see Fig. 9. Special gimbal-based endeffectors are designed to hold the cue properly. A Basler acA1300-30gm ceiling-mounted camera with a resolution of 1280×960px and a frame rate of up to 40Hz mounted approximately 2.50m above the table provides information about the cue and ball positions, being able to distinguish between white, black, striped and solid balls. In order to place the cue properly behind the cue ball, combined data of the camera, the robot’s Sick S300 laser range finders, arm pose data and two JR3 endeffector force/torque sensors is
used. The robot is able to move autonomously around the table by detecting the legs of the pool table and execute a stroke independently. Two onboard computers with a 2.66 GHz Intel Pentium i7 920 CPU and 12GB RAM control both arms and the platform. The combination of a proprietary real-time data base [49] for data exchange with a real-time architecture [50] allows an update rate of 1000Hz while sufficing hard real-time constraints.

To create an search tree within reasonable time, pool simulations are parallelized over 40 computers equipped with a AMD Phenom II X6 1075T 3GHz CPU and 8GB RAM. Another computer with similar specifications is used to compute an optimal stroke as described in Sec. V based on the computed search tree. During tree creation, a variable branching factor in the range of 300 is used, resulting in a about 100000 computed strokes for a search depth of 2. The planning capabilities of the robot are evaluated by comparing two scenarios with the same human opponent. In the first scenario the search depth $n$ is set to 0, see (19). The stroke intensity $p$ has been set to a constant value of 0.17Ns. In total 7 games have been played, resulting in 89 strokes of the robot. In the second scenario the search depth $n$ is set to 2 while also considering the human decision making according to Sec. IV. Here 6 games have been played, resulting in 67 strokes of the robot. There are not enough games played for any conclusion about the statistical significance of the number of wins/losses. It is however possible to draw conclusions by looking at the individual strokes. A stroke is considered as “successful” if the robot can pocket a ball and without committing a foul, thus corresponding to the “with repositioning” run in Fig. 7. All other strokes executed by the robot are considered to be “unsuccessful”. Using this definition, the two results differ significantly from each other (based on a two sample t-test) as only 25 out of the 89 strokes are successful for the first scenario whereas 34 out of the 67 strokes are successful for the second scenario.

C. Discussion

The performed experiments show that both excessive planning and learning a model of the opponent significantly improves the outcome of a pool game. Still, there are some more general aspects regarding the human model to be discussed.

A quite unique problem at pool is the coupling between a player’s skill in terms of success rate when pocketing a ball and his planning capabilities. A chess player’s skill is primarily determined by his planning capabilities whereas the success of a golf player depends mainly on his skill when doing a golf swing. Yet with the exception of novice pool player one cannot tell by just looking at the result of a game whether a victory comes from superior skill or planning. The used robotic platform during the experimental evaluation possesses superior planning capabilities but inferior motor skill compared to the human opponent. Thus hardware improvements resulting in lower standard deviations $\sigma_p$ and $\sigma_\theta$ are expected to have a major impact on the outcome of a game.

Only strokes up to a search depth of 2 are simulated. Due to imperfect calibration the deviation between simulation and experimental results increases as the search tree is extended deeper. To account for this problem one may introduce an additional parameter which weights future strokes less. Still this parameter is different from the discount factor $\gamma_{(r,h)}$ that accounts for the planning capabilities of human and robot.

VII. Conclusion and Outlook

The article deals with the question how a robot facing a human in a competitive task can improve its performance through extensive planning and precise representation of the opponent’s behavior. This allows
the robot to adapt to human-specific decision making and the opponent’s individual motor skill while overcoming its own kinematic constraints. Simulations show that including the human model in the planning framework of the robot significantly improves the win ratio. Real world experiments with an anthropomorphic robotic platform prove that extensive planning significantly helps for a better position play.

Future work will be focused on employing reinforcement learning for better online adaption to the human model, resulting in a better overall performance against each individual player.

ACKNOWLEDGEMENT

The research leading to these results has received funding from the ERC Starting Grant "Control based on Human Models (con-humo)" under grant agreement n° 337654 and by the DAAD (www.daad.de).

REFERENCES


Thomas Nierhoff is a research assistant at the Institute of Information-Oriented Control, Technische Universität München, Germany. He received the B.Sc. and Dipl.-Ing. degree (M.Sc. equivalent) in electrical engineering from the same university in 2009 and 2010. From 2012 to 2014 he has been a visiting researcher at Nakamura Labs, Tokyo, Japan. His research interests are in the field of robotics, motion planning and machine learning.

Konrad Leibrandt received the Dipl.-Ing. degree (M.Sc. equivalent) in electrical engineering from Technische Universität München (TUM), Germany in 2012 and the M.Res. degree in Medical Robotics and Image Guided Intervention from Imperial College London (ICL), United Kingdom in 2013. He is currently working towards a Ph.D. degree jointly at the Hamlyn Centre for Robotic Surgery and the Department of Computing at Imperial College London. His research interest includes active constraints and virtual fixtures, ergonomic control for human-robot interaction in surgery, continuum robot control, as well as high-performance parallel computing for surgical applications.

Tamara Lorenz is a research assistant at the Institute of Information-Oriented Control at Technische Universität München. Her research focus is on the dynamics and rules that underlie human-human interaction. She is also interested in how human interaction can be modeled to improve the interaction between humans and robots. As her work is very interdisciplinary, Tamara Lorenz was additionally affiliated with the Experimental Psychology Department at Ludwig-Maximilians Universität (LMU) from 2009 to 2013 and got her PhD form the Graduate School of Systemic Neurosciences, LMU, in 2015. Tamara Lorenz holds a diploma (M.Sc. equivalent) in Mechanical Engineering from Technische Universität München (TUM), Germany, specializing in human-machine interaction and medical engineering.

Sandra Hirche received the Diplom degree in Mechanical Engineering and Transport Systems from the Technical University Berlin, Germany, in 2002 and the Doctor of Engineering degree in Electrical Engineering and Information Technology from Technische Universität München, Munich, Germany, in 2005. From 2005 to 2007 she was awarded a Postdoc scholarship from the Japanese Society for the Promotion of Science at the Fujita Laboratory, Tokyo Institute of Technology, Tokyo, Japan. From 2008 to 2012 she has been an associate professor at Technische Universität München. Since 2013 she is the TUM Liesel Beckmann Distinguished Professor and has the Chair of Information-oriented Control in the Department of Electrical and Computer Engineering at Technische Universität München. Her main research interests include cooperative, distributed and networked control with applications in human-robot interaction, multi-robot systems, and general robotics. She has published more than 150 papers in international journals, books and refereed conferences. Dr. Hirche has served on the Editorial Boards of the IEEE Transactions on Control Systems Technology and the IEEE Transactions on Haptics. She has received multiple awards such as the Rohde & Schwarz Award for her PhD thesis in 2005, the IFAC World Congress Best Poster Award in 2005 and together with students Best Paper Awards of IEEE Worldhaptics and IFAC Conference of Manoeuvring and Control of Marine Craft in 2009.