Action Selection for Hammer Shots in Curling

Zaheen Farraz Ahmad, Robert C. Holte, Michael Bowling

Department of Computing Science, University of Alberta

Hong Cheng
chenghong@kw.ac.kr
Advisor: Prof. Hyukjoon Lee

Computer Intelligence Network Laboratory, Dept. of CE, KwangWoon University

CINe Lab

Introduction

- **Curling**
  - an Olympic sport played between **two teams**(each team with **4 people**)
  - **Ends** : The game is played in a number of **rounds** (usually **8**)
  - When each team has thrown **8** rocks, the score for that end is determined and added to the teams' scores from the previous ends. The team with the highest score after the final end is the winner.
- **Hammer shot**
  - **Last shot of an end**
  - It is of the **utmost importance** as it **heavily influences** the **score** for the end.
- **Focus**
  - On the problem of **selecting the hammer shot**.
  - **Removes the need** to reason about the opponent,
    - while still leaving the substantial challenge of efficiently identifying a near-optimal action in a continuous state
    - And action space with stochastic action outcomes and a highly non-convex scoring function.

Introduction

- **Heatmap**
  - Illustrate the **difficult nature** of this restricted **optimization problem**
  - **Shading** represents score as a function of two main continuous action parameter
    - Angle θ
    - Velocity v
  - it shows the exact score if shot (θ, v) is executed without error
- **Finding an optimal shot**
  - Means finding the darkest parts of this heatmap

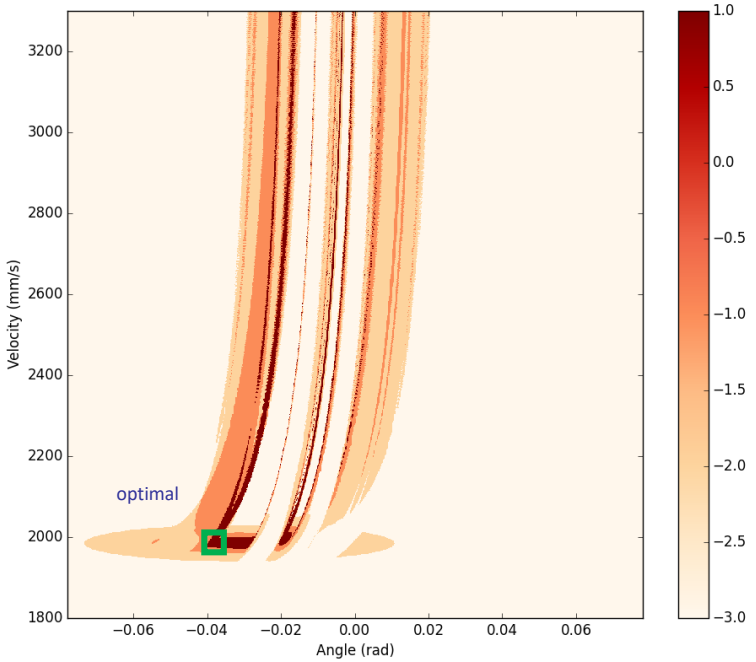
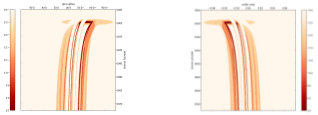
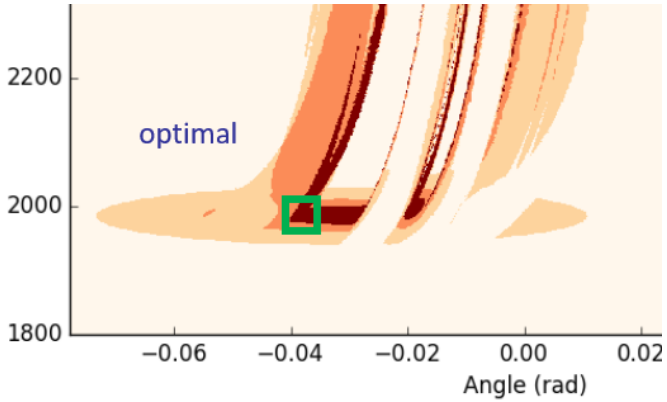


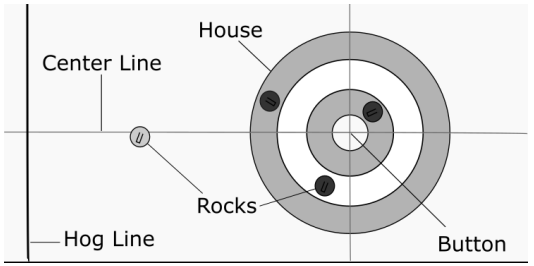
Figure 1: Heatmap showing the highly non-convex scoring function for a counterclockwise turn in the state in Figure 2.

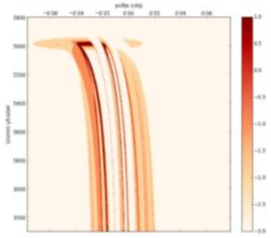


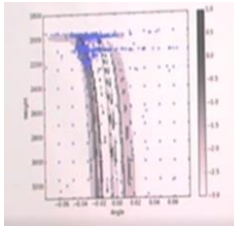
Introduction

- (θ, v) and (θ', v')
  - Because **action execution** is **stochastic**, the expected value of shot (θ, v) is the average noise-free values of shots,
  - (θ', v'), weighted by $p((\theta', v') \mid (\theta, v))$
    - Given that (θ, v) is the intended shot
    - The probability that (θ', v') is executed
  - This essentially **blurs the deterministic heat map**
    - making the **darkest regions** even **smaller**
    - only the central region of the **rectangle**
  - where $\theta \approx 0.04$ and $v \approx 2000$ is optimal in expected value in the situation shown in Figure 1









Introduction

- **Two main contributions**
  - **Delaunay Sampling (DS)**
    - **Use Delaunay triangulation which is an non-convex optimization method**
    - on a set of sampled points (*Surovik and Scheeres [2015]*) to discretize the continuous action space and focus subsequent sampling in regions that appear promising.
    - Add a final step, in which **a shot** is selected by treating the most promising regions as "**arms**" in a **multi-armed bandit problem**.
  - **Evaluate the effectiveness** of **DS** and a representative set of existing algorithms for non-convex function optimization for hammer shot selection

The Hammer Shot in Curling

- **The *state* of hammer shot is determined by**
  - score differential
  - number of ends left to play
  - (x, y) positions of rocks in play
- **Curling rule**
  - Rock is **thrown** with an **initial linear** and **angular velocity** at some angle relative to the center line
    - angular velocity causes the rock to **travel in an arc** ("**curl**")
    - deviating from the straight line path that the rock was initially on
  - The **direction** of the curl is **determined** by
    - the **sign(**clockwise or counterclockwise) of the angular velocity,
    - but is little affected by the magnitude of the angular velocity
  - **Sweep**
    - Affect the **deceleration** and the amount it **curls**

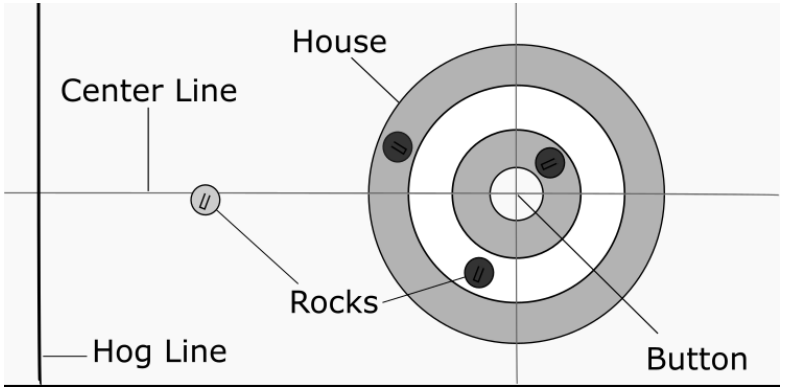


Figure 2: Curling state with 4 rocks in play

- **Score**
  - The team with the rock in the **scoring area** that is **closest** to the center **scores**.
  - The **number** of **points** is equal to the **number** of **rocks** *in the scoring area* that are closer to the center than any rock of the opposing team.

The Hammer Shot in Curling

- **Intended shot is not realized**
  - **Human error**
    - The player **throwing** the stone might **not perfectly** deliver it at the required angle or velocity.
    - the **skip**(team leader or captain) may **incorrectly judge** the rock's path or the sweepers its speed, resulting in **sweeping being misapplied** to achieve the desired outcome.
  - **Variability in the ice and rocks**
    - **ice conditions** can change as a game goes on
    - every **rock** is slightly **different** in how it interacts with the surface of the ice.

The Hammer Shot in Curling

2.1 Modelling the Hammer shot

- using the **continuous** (x, y) **coordinates** for each of the **previously thrown 15 rocks** still in play, as well as the **score differential** and **number of ends remaining**
- **Action space with two simplifying assumptions**
  - we **treat the angular velocity** of the hammer shot **as a binary variable** (clockwise or counterclockwise)
  - **do not have any parameters related to sweeping in our action space.** Instead we integrate the effects of sweeping into the execution model of our curling simulator
- **Action space**
  - two **continuous dimensions** (θ, v) [$(x, y)_n$ $(ends, score)$ is state]
  - one **binary dimension ("turn")** [binary]

The Hammer Shot in Curling

- **The execution model**
  - using the Chipmunk 2D rigid body physics library
  - A rock's trajectory is modeled by **a deterministic simulation given an initial linear velocity, angle, and turn.**
  - Represent the **variability** in **outcomes** in the execution of **an intended shot.**
    - The variability treated as a stochastic transformation
    - If (θ, v) is the intended shot, outcome is the deterministic simulation of a perturbed shot (θ', v')sampled from a predetermined conditional probability distribution whose mode is intended shot
  - Purpose of **sweeping**
    - to correct for human error and variability in the ice conditions.
    - This is implicitly incorporated into the execution model as a reduction in the execution noise
  - Do not model **ice** and **variability**
    - Assume all shots are subject to the same execution error

Related Work

3.1 Continuous Bandits

General Bandit Problem

- an **agent** is presented with a set of **arms**
- Each **round**, the agent selects an **arm** and receives a **reward**.
- The reward received from an arm is an i.i.d (independent and identically distributed) **sample** from the unknown distribution associated with that arm.
- The **objective** of the agent is to select arms to maximize its cumulative reward.

Related Work

3.1 Continuous Bandits

- **UCB(Upper Confidence Bounds)**
  - an **arm selection policy** that associate with each arm an upper bound on the estimated expected value given the entire history of interaction

$$v_i = \bar{r}_i + C\sqrt{\frac{\log N}{n_i}}, \qquad (1)$$

  - $\overline{r_i}$ is the average reward observed by the arm i
  - N is the total number of samples
  - n_i is the number of times arm i was selected
  - C is a tunable constant
  - On each round the agent chooses an arm i that maximizes v_i
- The two terms in the upper bound balance between
  - **exploiting** an action with high estimated value
  - **exploring** an action with high uncertainty in its estimate.

Related Work

3.1 Continuous Bandits

- generalizes this framework to continuously parameterized action spaces *[**Bubeck** et al., 2009; Kleinberg, 2004].*
- The agent is presented with a set of arms X, which form a topological space
- On each round, the agent selects a point $x \in X$ and receives a reward determined by a randomly sampled function from an unknown distribution, which is then evaluated at x.
- With some continuity assumptions on the mean function of the unknown distribution, one can devise algorithms that can achieve a similar exploration-exploitation tradeoff as in the finite arm setting.
- One such algorithm is **Hierarchical Optimistic Optimization(HOO)** *[**Bubeck** et al., 2009]*

Related Work

3.1 Continuous Bandits

- **Hierarchical Optimistic Optimization(HOO)**
  - HOO creates a **cover tree** spanning the action space which successively divides the space into smaller sets at each depth.
  - The **nodes** of the cover tree are **treated as arms of a sequential bandit problem**.
- **Exploitation** is done by traversing down **previously-explored, promising nodes** to create a finer granularity of estimates
- **Exploration** is performed **by sampling nodes** higher up in the tree covering regions that have not been sampled adequately
- Under certain **smoothness assumptions**, the average reward of HOO's sampled actions **converges to the optimal action**.

Related Work

3.2 Gaussian Process Optimization

- *[Rasmussen, 2006; Snelson and Ghahramani, 2005; Snoek et al., 2012; Lizotte et al., 2007]*
- Based on Gaussian processes
  - Which are **priors over functions**
  - parameterized by a **mean function** $m(x)$ and a **covariance kernel function**, $k(x, x')$
- Suppose there is an unknown function f and one is given N observed data points $\{x_n, y_n\}_{n=1}^{N}$
- Where $y_n \sim N(f(x_n), v)$ with v being a noise term
- The **posterior** distribution on **the unknown function** f is also a Gaussian process with a **mean** and **covariance** kernel function computable in closed form from the data.
- **GPO** seeks to find a **maximum** of an unknown function f through carefully choosing new points x_{n+1} to **sample a value based on the posterior belief from previous samples**.

Related Work

3.2 Gaussian Process Optimization

- The optimal selection mechanism for a non-trivial sampling horizon and prior is **intractable**
- However, good performance can often be had by instead choosing a point that maximizes some acquisition function as a proxy objective
- **choosing the point with the maximum probability** of improving on the largest previously attained value.
- **Drawbacks**
  - **the choice of prior** can have a significant impact on its performance that **shifts the problem to selecting hyperparameters**
  - **computation** to **identify a new sample point** is still **growing** with the number of previous samples, which can become intractable when allowing a large number of samples.

Related Work

3.3 Particle Swarm Optimization

- *[Eberhart and Shi, 2011; Shi and Eberhart, 1998]*
- a population-based, stochastic approach to optimization.
- Each particle in the population keeps track of its **own best score** and **the global best score** evaluated at each time step.
- The particles (samples) are **initially evaluated at uniform random points** in the action space.
- At each iteration of the algorithm, the particles take a step toward **the current global best.**
- The **step size** depends on **weight and velocity parameters**, which are decided by the **practitioner**, and on the **particle's own best score**.
- Unlike HOO and GPO
  - which make **a global convergence guarantee** under smoothness assumptions
- PSO is only expected to converge to a **local optimum**, and so results **depend** on the **quality** of the **initial sampling of points**.

Related Work

- **3.4 Covariance Matrix Adaptation – Evolution Strategy**
  - (CMA-ES) is **iterative algorithms** that attempt to optimize a function by **introducing stochastic variations at each iteration.** *[B¨ack et al., 1991]*
  - proceeds by drawing a set of samples from an (initially uninformed) **multivariate Gaussian** *[Hansen and Ostermeier, 1996; Hansen, 2016]*
  - A **new multivariate Gaussian** is then **constructed**.
  - The **mean** is **the weighted mean** of the sampled points, where higher weights are given to samples with larger reward values.
  - The covariance matrix is modified from the previous covariance so as to encourage high variance in the direction that the mean is observed to be changing.
  - This procedure is then repeated using the new multivariate Gaussian to sample points.
  - As with PSO, successive generations produced will lead to convergence to a local optimum of the function, with no guarantee with respect to the global optimum.

Related Work

3.5 Curling and Billiards

- Strategies for **curling** *[Yamamoto et al., 2015].*
- Work in **billiards**, a game which is similar to curling with **continuous actions** and **states along with stochastic outcomes**, has received some attention in the literature *[Archibald et al., 2009; Smith, 2007; 2006].*
- However, in all of this work, the authors use **domain knowledge** to **create a finite set of discrete actions**.
- This makes it possible for search to be employed without addressing the challenges of optimization in continuous settings.
- **Our work** seeks to **forego domain knowledge** as much as possible, in favor of a **more domain-independent** approach.

Delaunay Sampling

4.1 Sampling with Delaunay Triangulation

- 1. Sample the (stochastic) **objective function** at points **distributed uniformly** over the range of values of the action parameters.
- 2. Apply **Delaunay triangulation** to the sampled points to partition the continuous action space into a set of disjoint regions.
- 3. **Assign** each region a **weight**, which we discuss below.
- 4. Sample the set of regions with replacement, with the probability of a region proportional to its weight.
- 5. Each time a region is selected in the previous step, uniform randomly sample a point within it and add it to the set of sampled points, with its value sampled from the (stochastic) objective function at that point.
- 6. Repeat from step 2 for a fixed number (T) of iterations.

Delaunay Sampling

4.1 Sampling with Delaunay Triangulation

- **specify** the **weight** w_i used for **region** i
- Let $t \leq T$ be the current iteration of the algorithm
- a_i refer to the area of region i
- v_{ij} refer to the observed reward of vertex j of the region i
- Define **the weight for region** i as

$$w_i = a_i^{1-t/T} \times s_i^{\sigma t/T} \qquad s_i = e^{\max_j(v_{ij})}$$

  - σ is a tunable parameter that controls the rate of exploration
- We call s_i **the *score* of the region** since it will be large if one of the vertices has been observed to result in high reward.

Delaunay Sampling

4.1 Sampling with Delaunay Triangulation

$$w_i = a_i^{1-t/T} \times s_i^{\sigma t/T} \qquad s_i = e^{\max_j(v_{ij})}$$

- Note that if a_i is large then w_i can be large, encouraging the algorithm to refine large regions;
- if s_i is large then w_i can be large, encouraging the algorithm to focus on regions with higher observed rewards.
- As t increases, more weight is put on refining the high-valued region, since ultimately only the high-value regions are used in the selection stage of the algorithm
- In Step 1 triangulations are done separately for each value of any discrete action parameters (**specifically, the binary "turn" parameter in curling**).
- The other steps use the union of regions from all triangulations. This allows the algorithm to allocate more samples to the most promising discrete parameter values.

Delaunay Sampling

4.2 Selection of the Final Action

- This stage chooses the action that will actually be executed using the following procedure.
- 1. Assign new weights to the regions,

$$\hat{w}_i = \frac{1}{|v_{ij}|} \sum_j v_{ij}. \qquad (4)$$

- 2.Select the k regions with the highest weights.
- 3. Run T iterations of UCB using the chosen regions as the arms of a bandit. When UCB samples a region, the sample is taken at the region's incenter, and its value is a (stochastic) sample of the objective function at that point.
- 4. Return the incenter of the region that was sampled most by UCB.

Delaunay Sampling

- **The first stage is exploratory**
  - Try to find promising regions that are relatively small.
  - For this purpose it makes sense to use an optimistic score for a region.
- **The second stage's aim**
  - **identify the region with the largest expected value.**
  - For this purpose it is appropriate to repeatedly sample the point within a region that would actually be returned.

Experimental Setup

5.1 Objective Function

- *WP*(win percentage) not **EP**(expected point)
- ~~find a shot with the maximum (EP) differential?~~
  - Consider choosing the very last shot of a game in which the team with the hammer is losing by **2** points. And suppose that :

Shot	Shot A	Shot B
Succeed Probability	100%	20%
Succeed Point	1	3
Failed Probability	0%	80%
Failed Point	0	-1
EP	1	-0.2
WP	0%	20%

Experimental Setup

5.1 Objective Function

- Game state variables before the hammer shot
  - n, the number of ends left to play
  - δ, the number of points by which the team with the hammer is leading (δ is negative if the team with hammer is losing)
- We call a pair (n, δ) the resulting game state, or g
  - For example if $g = (1, -2)$, $WP(g)$ is the probability of the team with hammer winning if it enters the final end down by two points
- We then fit this WP function to data using 28,000 curling games played between 2011 and 2013

Experimental Setup

5.1 Objective Function

- For the final end of a game, $g = (1, \delta)$, we estimated $WP(g)$ using simple frequency statistics for the final ends from the dataset.
- For the second last end of a game we used the same data to estimate the transition probabilities from $g = (2, \delta)$ to a game state $g' = (1, \delta')$ for the final end.
- This tells how frequently it happened that the hammer team having a lead of δ when there were two ends to play was followed by the hammer team in the final end having a lead of δ'
- With these transition probabilities and the already-computed values of $WP((1, \delta'))$ for all δ', it is easy to compute $WP((2, \delta))$ for all δ.
- The same process can then be applied to compute $WP((3, \delta))$, $WP((4, \delta))$, etc. With δ defined for all values of n and δ, the "score" we return when a hammer shot with x ends remaining results in a lead of δ for the team with the hammer in the next end is $WP(x-1, \delta)$.

Experimental Setup

5.2 Experimental Design

- The **data** used for our experiments was drawn from the hammer shot states captured by hand from the **2010 Winter Olympics men's and women's curling** competition.
- A set of **397** hammer shot states from these logs were used in the parameter sweeps mentioned below.
  - The parameter **sweep** for **Delaunay Sampling (DS)** chose **a value of 14 for σ** .
- A separate set of **515** hammer shot states (the "**test states**") from these logs were used to **evaluate** the systems (including the humans).

Experimental Setup

5.2 Experimental Design

- For each of the test states, we gave **DS** a budget of between **500** and **3000** samples. This budget includes
  - **100** samples for initializing the triangulation over each turn
  - **100** samples per iteration of the first stage
  - **100** samples for the final UCB stage.
- After **DS** selected a shot, **10** outcomes were sampled from the simulator, with the outcome's sample mean used as the resulting estimate of **WP**.
- This procedure was repeated **250** times for each test state.
- The values we report in **Table 1** are the average over the **2500** evaluations for each test state.

Experimental Setup

5.2 Experimental Design

- **HOO** selected a sample by choosing a *turn* using **UCB** and then expanding the **cover tree** for that *turn*.
- We ran **HOO** for **250** trials over the test states using the same sampling budgets as **DS**.
- The parameters for **HOO** were set by a parameter **sweep** to
- $\rho = \frac{1}{\sqrt{2}}, v = 2\sqrt{2}$ and **UCB** constant $C=0.01$

Experimental Setup

5.2 Experimental Design

- **PSO** was tested slightly differently
- Since the particles move in a continuous space, having a discrete parameter (“turn”) required us to run **PSO** on each turn separately.
- For each test state, **PSO** was run using one turn value and then the other with each run being provided the full sampling budget.
- The best shots found with each turn were compared and the one with the higher average WP was evaluated one final time to compute its expected WP.
- This was performed 250 times for each test state. **PSO**’s parameters were set to $c1 = 1.5, c2 = 1.5, w = 0.7$ and 50 particles

Experimental Setup

5.2 Experimental Design

- We used the **BIPOP-CMA-ES** *[Auger and Hansen, 2009]* version of **CMA-ES** for our tests.
- The experimental setup was the same as for **PSO**.
- The parameters for the number of offspring and parents were, respectively, $\lambda = 4 + \log 2$ and $\mu = \lambda / 2$ with 4 restarts.
- We set the initial standard deviation to $\sigma_0 = 0.25$ and normalized the action parameters θ and v to range between 0 and 1.
- The parameters for step-size control and the covariance matrix adaptation were set in accordance to the values recommended by *Hansen [2016].*

Experimental Setup

5.2 Experimental Design

- **GPO** was tested in a manner similar to **PSO**.
- However, since **GPO** is *considerably slower* we implemented it with only a budget size of **250, 500 and 1000** samples per turn with **50** *restarts*.
- We chose the squared-exponential function as our **GP** *kernel* and set the **noise variance**, $\sigma_n = 0.01$.

Experimental Setup

5.2 Experimental Design

- **Wilcoxon signed-rank test (WSRT)**
- To determine if **DS**'s average ***WP*** on the test states was **statistically significantly different** than the average ***WP*** of the humans or other systems tested,
  - with one pair of values for each of the ***515*** test states
    - one of the values was **DS**'s ***WP*** on the state
    - the other was the ***WP*** of the system **DS** was being compared to
- Unless otherwise noted, **WSRT** produced ***p***-values sufficiently small to conclude with greater than ***99%*** confidence that **DS**'s superior average ***WP*** (for any specific sample budget) is not due to chance.

Results and Evaluation

- **Comparison with different algorithm**

DS Budget	DS	HOO	PSO	CMA	GPO
500	0.4956	0.4273	0.4853	0.4932	0.4857
1000	0.5203	0.4787	0.4856	0.4958	0.4868
1500	0.5277	0.4968	0.4858	0.4954	-
2000	0.5310	0.5115	0.4849	0.4933	0.4867
2500	0.5331	0.5176	0.4858	0.4954	-
3000	0.5343	0.5212	0.4854	0.4948	-

Table 1: Average WP for different sample budgets.

Results and Evaluation

Comparison to Human Olympic Teams

- The hammer shots actually taken by the Olympic teams on the test states had an average WP of 0.4893.
- With a sample budget of 500, **DS** is not significantly better than the curlers (**WSRT** p-value = 0.1509)
- But for all larger sample budgets **DS** is significantly better ($p < 0.01$).
- Nevertheless, on approximately 20% of the test states the Olympic teams chose a shot that was superior to the shot chosen by **DS**.

Results and Evaluation

- **Comparison to Human Olympic Teams**

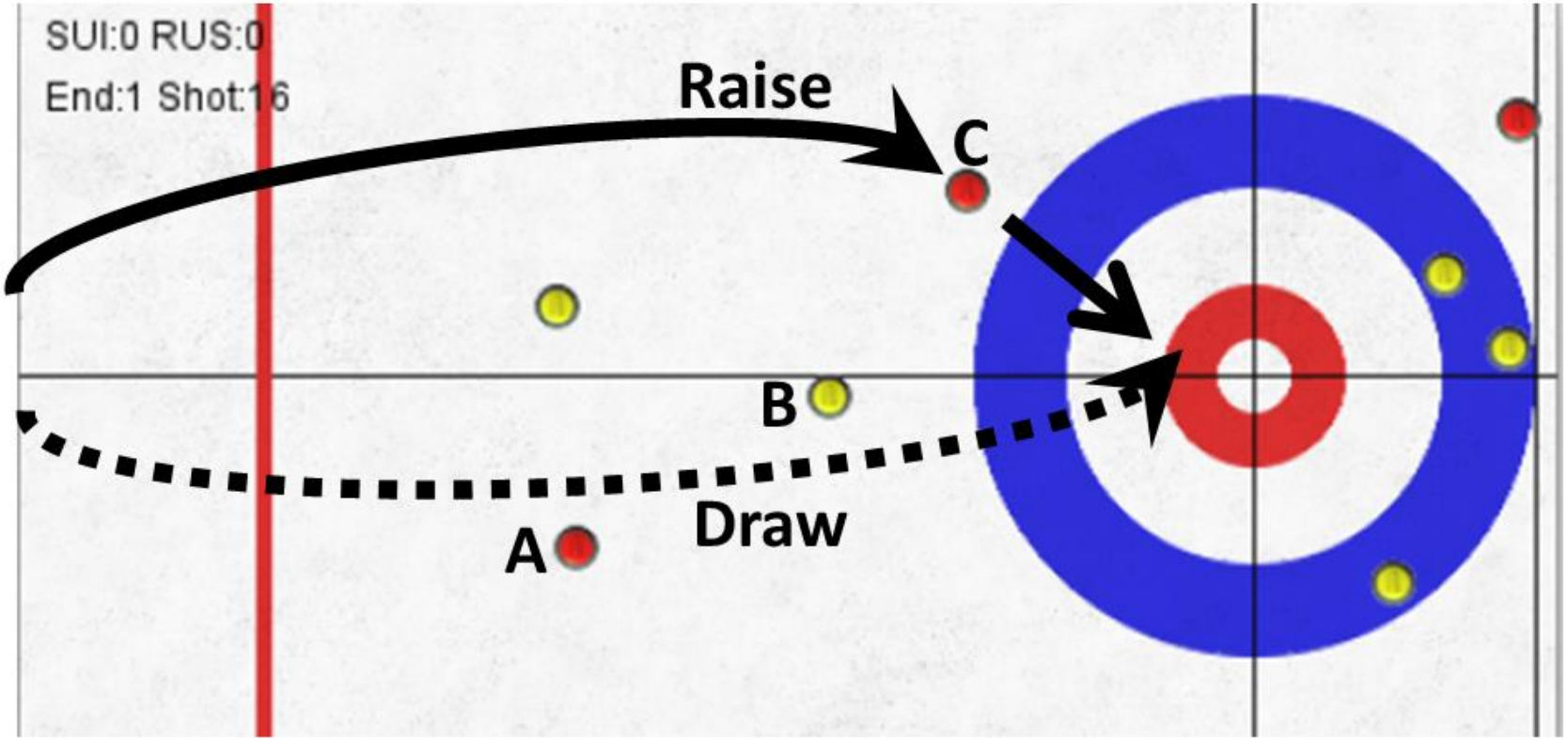


Figure 3: Two shots that score one point for the dark team

Results and Evaluation

- **Limitations of this comparison**
  - we were **not able to** repeatedly **sample human executio**n of a shot to obtain its **expected value**.
  - Our physics simulation is not a perfect replica of the true physics of curling.
  - the data from the Olympics was logged by hand, so the positions of the rocks, as recorded in the logs and used by DS in our study, might not be exactly the positions faced by the curlers during the Olympic games.

Conclusion

- Selecting a hammer shot in curling is a challenging **low-dimensional non-convex optimization problem**.
- We have tested several existing methods for non-convex optimization on hammer shot states
- All of them were significantly **inferior**, in terms of average win percentage (WP), to the Delaunay Sampling (DS) method
- DS also achieved a significantly higher WP than the Olympic curlers.