MIT SLOAN SPORTS ANALYTICS CONFERENCE presented by ESPN

Bhostgusters: Realtime Interactive Play Sketching with Synthesized NBA Defenses

Thomas Seidl[1], Aditya Cherukumudi[†], Andrew Hartnett[†], Peter Carr[†2], Patrick Lucey[3]
Technical University of Munich[1], Argo AI[2], STATS[3]

1. Introduction

Over the last decade, quantitative methods for analyzing player and team performance have pervaded professional sports, and the NBA has been at the forefront of this movement. With the widespread availability of tracking data, teams can now scout upcoming opponents in a detailed manner and analyze their own performance after the game.

Having readily available analysis for "heat-of-the-moment" decisions, however, is critical to in-game success. Bill Walsh, the hall-of-fame NFL coach of the San Francisco 49'ers, knew this back in the 70's and crafted a "look-up table" that specified which play to run, depending on the yardage, time, and game-context [1].

Due to the continuous and dynamic nature of basketball, having a fixed look-up table of plays is not practical. Instead, coaches often use instinct and experience to create new variations of set plays—Brad Stevens being a great example [2,3]. However, intuiting an effective play is notoriously difficult [4]:

> *"The perfect [play] is part preparation, part matchup(s), part advance scouting work and a large part of spontaneous creativity. How tough is it to draw up a set/action in a small amount of time? Consider that a coach must anticipate how the defense will match up, how it will play screens and what (or who) it will try and take away. So, it's a rhetorical question, kids. It is very tough."*

In this paper, we consider play sketching from a data-driven perspective. What if a coach didn't have to rely solely on intuition, but instead could see instantly how the defending team is likely to respond (see Fig. 1)? Such a tool would still enable spontaneous creativity, but provide realtime objective analysis tailored specifically to the sketched play. Fortunately, we have made this tool a reality[1] by developing a powerful analytics framework embedded within an intelligent user interface.

As Le *et al.* [5] demonstrated at SSAC 2017, neural networks trained via deep imitation learning can model the defensive tendencies of soccer teams. Given tracking data for offensive players and the ball, the neural network hallucinates tracking data of "ghost" players to illustrate how a team is likely to defend. A key drawback of their approach was that the model lacked sufficient fidelity to make realistic predictions. Although it could model the tendencies of specific teams, it didn't

[†] This work was conducted while at Disney Research.
[1] http://www.bhostgusters.com/video

2018 Research Papers Competition
Presented by:

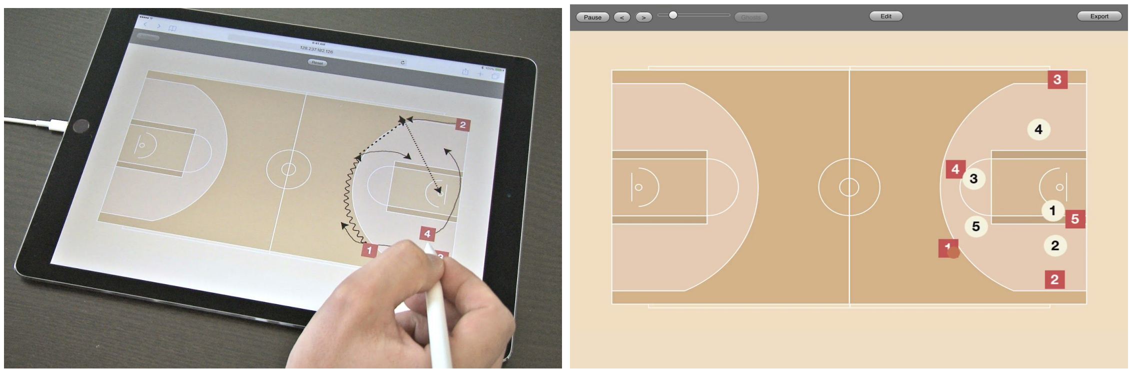

Figure 1: (See video and online demo) (left) interactive sketching, and (right) frame from corresponding synthesized tracking data with "ghost" players shown as white circles

take into account the relevant context, such as the score, the fatigue of the players, etc. In this work, we adapt the ideas of data-driven ghosting to basketball, and address the shortcomings of the previous work. We explicitly model numerous contextual features and illustrate our ability to make accurate predictions of how teams will behave in specific situations. We quantify the predictive abilities of our model by comparing the distance discrepancy between defensive players and ghosts.

Whiteboards[2] are still the de facto method for sketching plays at every level of basketball, from middle school to the NBA. Although apps exist for sketching plays electronically (such as [6]), they all employ complex interfaces that could never be used for in-game decisions. With this in mind, we have developed a web-based application that mimics the familiar marker/whiteboard interface for creating sketches in seconds. Although sketches imply temporal information, no timing data is explicitly defined. As a result, it is impossible to generate ghost responses to a sketched play directly (since data-driven ghosting requires tracking data as input). Instead, we estimate the tracking data that would be recorded if players carried out the sketched play. This hypothetical tracking data is then used as the basis for data-driven ghosting to determine how a specific team will react in turn.

Our system is composed of three distinct modules. The user interface manages sketching input and animation output. Hypothetical ball and offensive player tracking data is generated from the sketch and input to a data-driven ghosting model which predicts how a specific team will defend. Finally, the complete set of hallucinated tracking data is run through an expected points model to quantify the effectiveness of the sketched play.

Because our system is highly intuitive and operates in realtime on a tablet, *it brings analytics out of the back office and places it courtside*. Coaches can sketch plays and instantly see how the opposition is likely to respond. Additionally, our interface can edit existing tracking data, which allows fans to Monday morning quarterback: simulating alternate offensive decisions using real game data and discovering whether these "what if" scenarios can bust the ghosted defenses.

[2] Or yellow legal pads in the case of Larry Costello.

2. Data-Driven Ghosting in Basketball

The concept of "ghosting" was first introduced by the Toronto Raptors in 2013 [7]. Based on thousands of training examples, the analytics staff developed rule-based algorithms to indicate the (x,y) location where defenders *should* have been, instead of where the players actually were. At SSAC 2017, Le *et al.* [5] presented ghosting models learned directly from a season of professional soccer data using deep imitation learning. The neural networks were able to model the tendencies of individual teams and predict their behavior in response to different attacking scenarios. The data-driven approach has many useful qualities, but it lacks sufficient fidelity to make realistic predictions. The context of the game is not taken into account, and dynamic factors like player fatigue are not considered. In this work, we apply deep imitation learning to basketball, incorporate additional features to capture game and player context, and validate our models on a held-out test set.

2.1 Data

In this work, we use STATS SportVU player tracking from the 2016-2017 NBA season[3]. Utilizing the associated event data and other simple handcrafted filters, we divide games into possessions. Possessions consist of ten 2D trajectories (one for each player) and one 3D trajectory (the ball). In addition to these trajectories, each possession has additional metadata: game clock, shot clock, player fouls, and the number of in-game seconds logged by each player thus far. For simplicity, we filter out possessions involving jump balls, free throws and other game delays. In total, the training data consists of 30,764 possessions, with an average duration of 12.64 seconds.

2.2 Deep Multi-Agent Imitation Learning

Recurrent neural networks are naturally suited to modeling variable length sequences. Specific RNN architectures such as long short-term memory networks (LSTMs) and gated recurrent units (GRUs) are particularly well matched to sequences with long-term dependencies. Furthermore, by stacking these units, one can construct extremely expressive models capable of capturing diverse dependencies across many time scales simultaneously. Here, individual trajectories are modeled using a two-layer LSTM. The complete architecture consists of five distinct two-layer LSTMs (one for each role). Each two-layer LSTM acts as a role-specific *policy*: determining where the player fulfilling the specified role should go next.

Training the network of LSTMs can be broken down into three distinct phases: pre-training, single-policy training, and joint-policy training [8]. Pre-training consists of predicting where a player will move in the next timestep (80ms) given the true current positions of all players. This alone will not produce realistic behavior that resembles anticipatory reactions, but it preconditions the model parameters for learning more difficult tasks. Next, policies are trained in isolation by iteratively predicting multiple timesteps into the future, given the true positions of all other players at each timestep. The discrepancy between the predicted and actual location of the specified role are computed over the entire trajectory and used to update the weights of the neural network. This procedure (rollout) allows each policy to learn how to plan into the future and how to recover if out of position. Finally, all five policies are trained together. The procedure is identical to that of

[3] The model has been trained and validated on data dating back to the 2012-13 season. See appendix for additional results.

single-policy training except that each model uses the predictions of the other four models, rather than the true defender positions. This joint training allows the policies to learn cooperative behaviors such as switching on a screen or providing weak-side help.

2.3 Results

The primary objective of our ghosting model is to synthesize realistic defensive behaviors. This realism should extend from individual-level fundamental behaviors such as closing out shooters or boxing out for rebounds, all the way up to team-level coherence to a man, zone, or hybrid scheme. Figure 2 highlights the scope and variety of real basketball behaviors we observe from ghost defenders. Additionally, ghosts transition between these behaviors smoothly, often indistinguishable from the real players.[4]

In order to add a sense of realism to the ghosting model, context features such as elapsed time on court (a proxy for fatigue) and fouls conceded were added as additional input features to the LSTM model. We manipulate these features (see Fig. 3) to observe their effect on prediction (e.g. artificially making a player tired or being close to fouling out).

Beyond passing the "eye test", ghosting models should faithfully represent the behavior of specific teams in specific contexts in order to have true value. We evaluate our models both in terms of accuracy (the ability to predict the (x,y) coordinates of where real players will go) and precision (the confidence interval associated with the predicted locations). Because we determine the parameters of the LSTMs using stochastic gradient descent (picking a random sample from the training data to compute gradient updates), different runs of the training procedure will generate slightly different models. Therefore, we run the training procedure 10 times and produce an ensemble of 10 different models (see Fig. 4a).

We compute prediction error for three different ghosting models: generic league average team, San Antonio, and San Antonio with player-context. In each case, we evaluate predicted defender locations for all test set possessions in which San Antonio is on defense (N=93). As expected, the predicted precision decays with time (see Fig. 4b); we are trying to predict the behavior of inherently stochastic agents. For simplicity, we compute the prediction error over the first 2s (twice the training rollout time) for all models. Introducing a team specific feature significantly increases the accuracy of the ghosts (error decreases from 8.22ft/player to 6.59ft/player). The addition of contextual features increases the error. We conjecture that the model becomes too complex to optimize with only half a season of training data.

To quantify overall prediction error, we trained 10 team-specific models for each of four teams representing elite (SAS, GSW), average (DAL), and below average (LAL) defenses[5]. All 40 models were subsequently evaluated on 25 possessions chosen from the test set to represent a wide range of offensive behaviors: transition, isolation, weave, pick-and-roll, etc. Notably, ensemble averages have lower absolute error than single models (see Fig. 4c).

[4] http://www.bhostgusters.com/video

[5] https://stats.nba.com/teams/defense

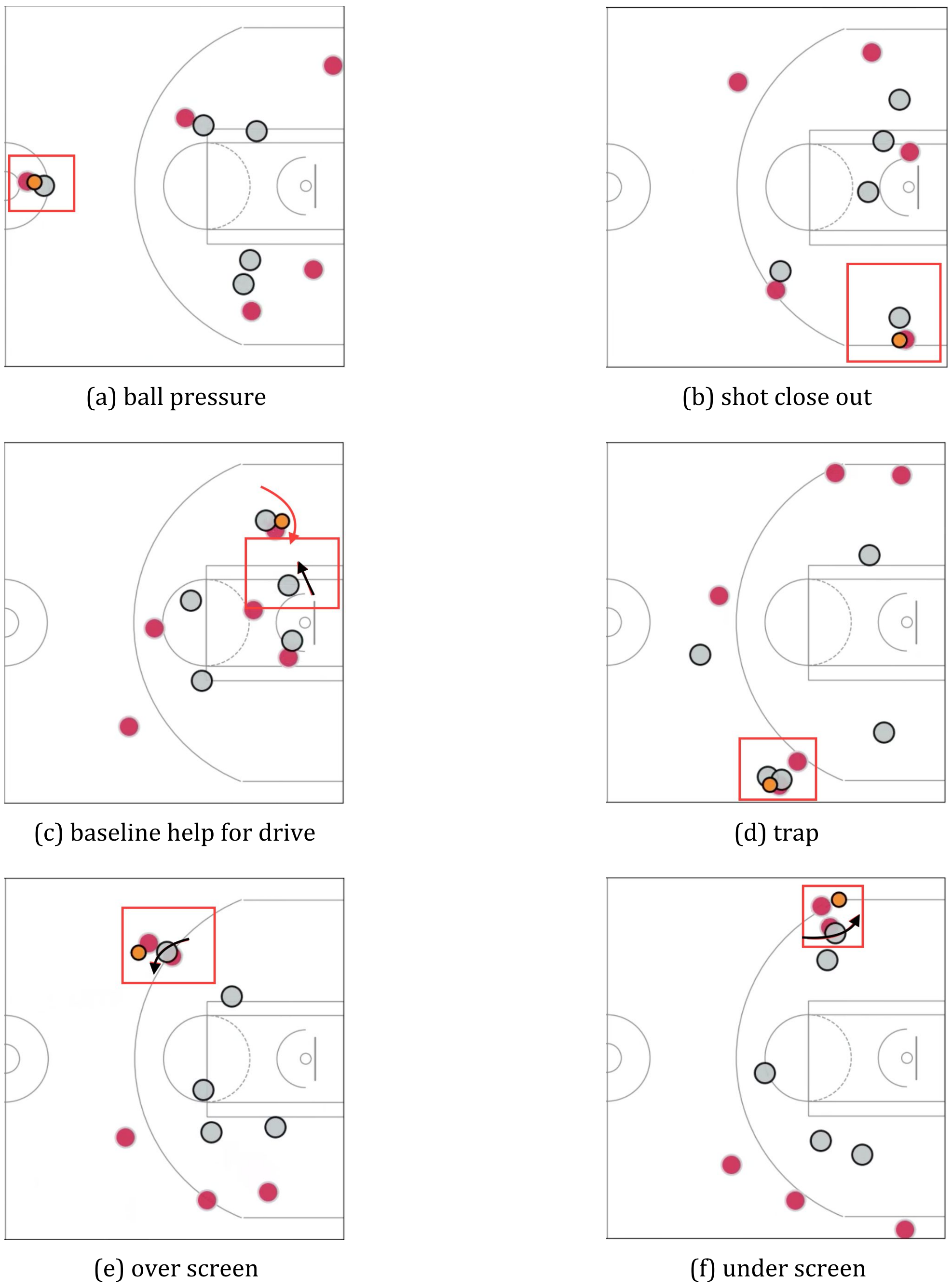

Figure 2: High-level ghost behaviors. See supplementary video for corresponding animations.

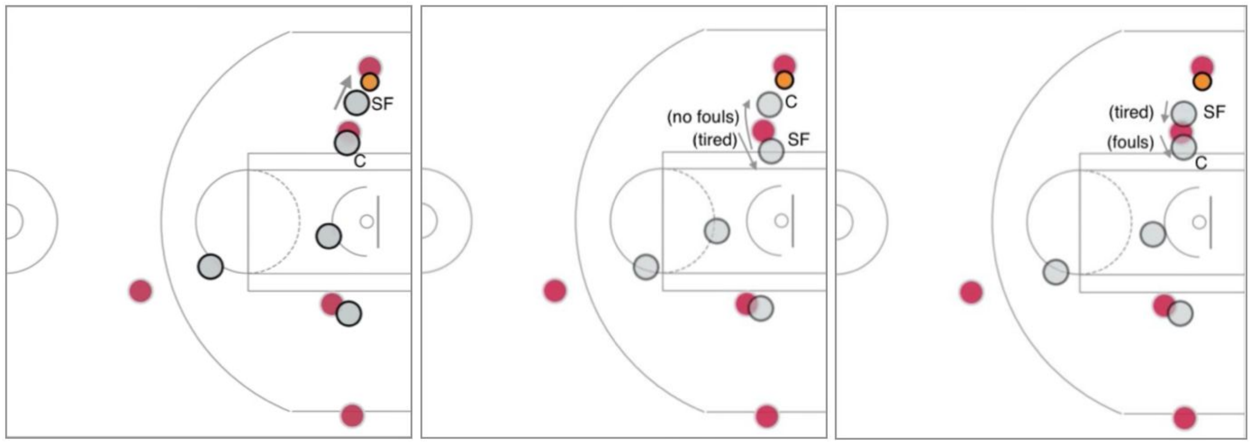

Figure 3: The effect of manipulating player-specific context features. A still frame from a sequence where the PG of the offensive team kicks it out to a shooter in the corner. (left) Small forward ghost (player fatigue clock ~6 minutes) closes out the shooter and ghost center (with 0 fouls) holds the PG tight to collect the defensive rebound. (middle) Small forward ghost (player fatigue clock changed to ~48 minutes, i.e full length of the game) falls back and the ghost center (with 0 fouls) covers for him and tries to close out the open shooter. (right) Ghost center (with 5 fouls) stays low and the tired small forward ghost is slow closing out the shooter.

2.4 Expected Points

In practice, other evaluation metrics may be more useful than discrepancy in predicted player location. For example, precisely predicting the location of off-ball players far from the action may not be important. We develop a simple on-ball model of expected points (EPM), similar to [9]. We have opted for a parsimonious regression model based on few features (shot location, ball state[6], and the distances to the two nearest defenders). This allows our model to be sensitive to high level behaviors such as, closeouts, weak-side help, and double-teams while robustly generalizing to the novel situations we anticipate from user generated sketches. More complex models, such as [10], could be substituted, but these may generate bizarre expectations if the sketched play is not realistic. Our model generates expectations (see Fig. 5a) for league average performance based on shot location which are consistent with other expected points models. Fig. 5b illustrates how the expected points changes over time for a particular possession. Although the positions of the players and the ghosts are not identical, the expected points is very similar; suggesting that the ghost behavior close to the ball is very similar to what the actual players did.

[6] Ball state is explicitly defined in sketches. It can also be easily and accurately inferred from the 3D ball trajectory data. We employ a vanilla gradient boosting classifier, trained on small set (~2000 frames) of manually labeled data.

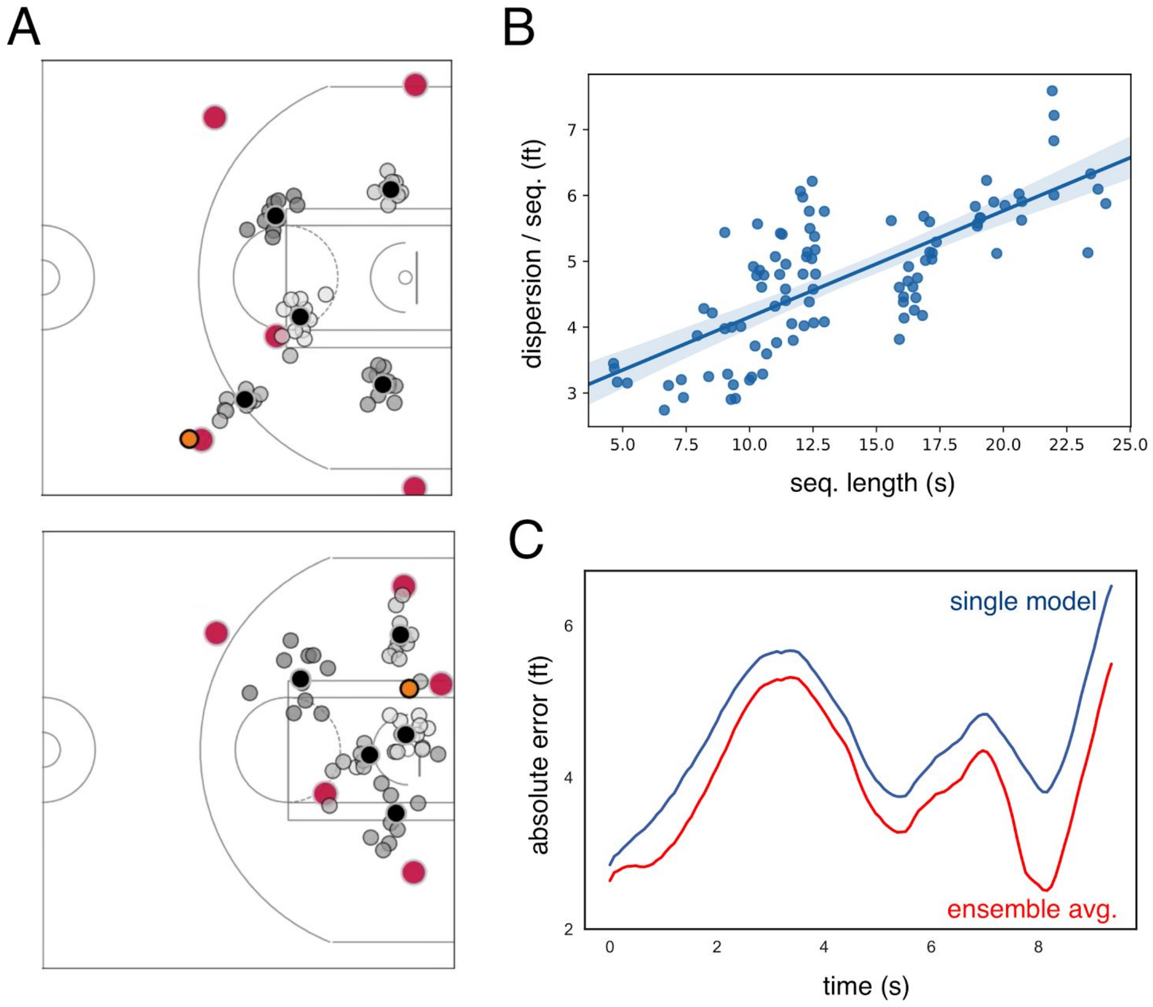

Figure 4: The precision and accuracy of our ghosting models. Panel A shows two different still frames with all ten SAS models (small gray circles) shown along with the ensemble average (black). The average dispersion between models, grows with the sequence length (approximately 0.16ft/s) as individual realizations adopt slightly different positions (Panel B). Dispersion remains low for sequences much longer than the training rollout. The absolute error (Panel C), or the typical distance between ghosts and the true defenders, also increases with time on average. At the level of an individual sequence, however, errors quickly shrink in situations with clear positional cues and grow in others (such as when players prepare for a rebound). Ghosts constructed from an ensemble average (red) have lower absolute errors than individual models (blue).

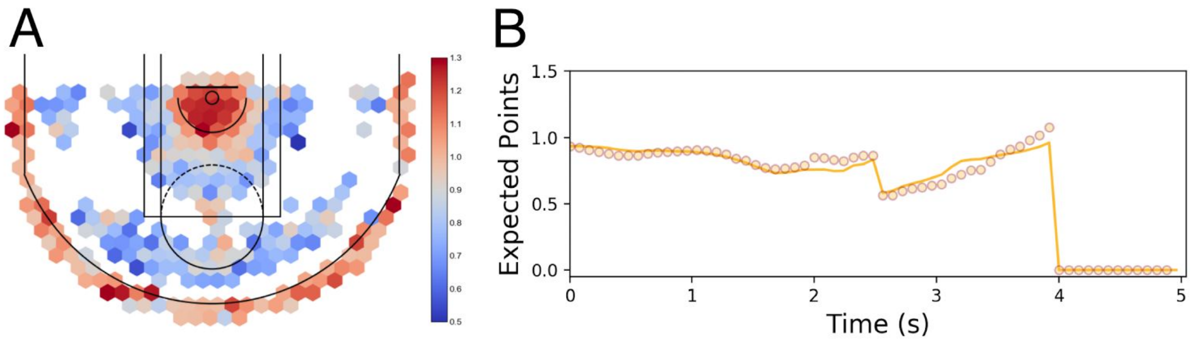

Figure 5: Panel A illustrates our on-ball expected points model evaluated on a test set of shots. Empty hexbins indicate locations with fewer than 20 shots. Red corresponds to high values of expected points (deep red = 1.3pts/attempt and dark blue = 0.5pts/attempt). Panel B shows the time series of expected points for a sample offensive possession. The solid yellow line (dotted line) indicates the expected points given the true positions of the CLE defenders (CLE ghosts). While the ghosts do not necessarily occupy the same positions as the true defenders, they play comparable defense. See supplemental video for the actual sequence.

3. Sketching

Through decades of use, a codified notation has emerged for sketching basketball plays [11]: wavy lines represent dribbling, dashed lines represent passes, etc. On whiteboards, it's necessary for the user to explicitly draw these conventions. In the digital domain, the intention can often be inferred and the appropriate convention drawn automatically (see Fig. 6).

We model a sketched play as a collection of *routes* (one for each player, and one for the ball). Each route is defined as an ordered set of segments, and each segment is an ordered set of (x,y) locations augmented with metadata. For player routes, the metadata indicates whether the player is dribbling the ball, setting a screen, etc. Similarly, ball metadata encodes if the ball is being dribbled, passed or shot.

Just like their analog counterpart, digital sketches are constructed through a series of stylus strokes (or mouse drags). Each stroke either defines a new segment for a particular route, or acts as an interpretable gesture [12] to modify metadata. To minimize the number of gestures, we interpret each stylus stroke within the context of basic basketball rules. For example, if a straight line is drawn between two players where one has the ball and the other does not, the intention is clearly a pass.

3.1 Inferring Timing

Sketches lack explicit temporal information. A route defines the path a player will move along, but no timing information is provided (e.g. it's not clear whether the player sprinted or jogged). Instead, players must learn how to interpret sketches and infer when to move, and how quickly.

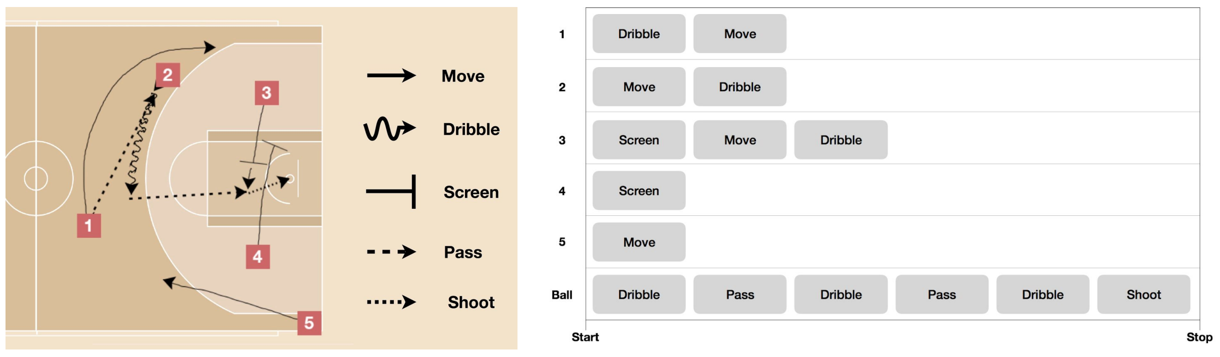

Figure 6: (left) The established notation for sketching basketball plays. (right) The underlying data representation of the sketch. Each *route* is represented as one or more segments. For simplicity, "dribble" represents a player with possession of the ball, but it could be a catch and shoot situation (e.g. player 3). The same dribble segment appears in both the ball route and the corresponding player route. All other segments are unique.

Tracking data, on the other hand, has explicit timing information. Unlike routes, *trajectories* are ordered sets of (x,y,t) data, often occurring at a fixed temporal interval (frame rate). In order to generate tracking data from a sketch, we infer the missing timing information using linear interpolation. Unfortunately, applying linear timing to each route often results in infeasible plays. A pass, for example, may end in empty space because the intended receiver was animated either too quickly or too slowly. If per-route linear timing were applied to the example sketched play in Figure 6, player 3 would only get to the correct catch-and-shoot location by the end of the play. Meanwhile, the ball would be animated such that the pass would already have been received and the shot taken.

Fortunately, because routes are defined as an ordered set of segments, it is easy to identify all spatiotemporal constraints that exist between the players and the ball. Applying linear timing per segment (instead of per route) guarantees a feasible solution—e.g. a pass will always go from one player to another. We estimate the duration of the sketched play by analyzing the route of the ball using empirically derived values for typical dribbling, passing, and shooting velocities. Once the beginning and ending times of each ball segment are determined, the temporal constraints of dribbles are propagated to the player segments (since the same dribble appears in both the ball and corresponding player routes). The timing of the individual player segments can then be determined using linear interpolation.

The inferred tracking data makes it possible to visualize a sketched play as an animation. In this format, the user gets immediate and unequivocal feedback about the design and timing of a play. Additionally, the animation makes it easy to communicate the intention of the set play to those without the requisite experience to interpret a sketch directly (such as novice players or data scientists). See the appendix for an example sketched play with ghosting responses.

3.2 Editing

Generating trajectories from routes is difficult because the missing timing information must be inferred. However, the reverse process (extracting routes from trajectories) is straightforward: drop the timing information and re-sample each trajectory using a fixed spatial resolution (e.g. sampling every half foot). The necessary metadata about the state of the ball can be extracted directly from the tracking data.

Because sketches can be generated directly from tracking data, a user can edit any offensive play that was run in any game. For example, players can erase the actual pass that took place, and visualize how the defense would have reacted if the ball had been passed to a different teammate. Similarly, the routes of the teammates can be modified to better spread the defense.

4. Summary

Answering detailed "what if" questions to help guide decision making is the core purpose of data analytics. Until now, insights extracted from player tracking data were only available post game—primarily because of the complexity of the algorithms and the domain-specific knowledge required to interface with the systems (i.e. scripting/coding). In this work, we bring analytics courtside for use in in-game decisions by combining data-driven ghosting with a digital sketching interface. Our framework is highly intuitive—anyone can draw a play and easily understand how a team is likely to defend against it. More importantly, because the algorithms run in realtime and the interface requires only a few stylus strokes, the system can instantly deliver objective analysis for very specific queries. To create this capability, we adapted data-driven ghosting from soccer to basketball, and devised algorithms to translate sketches into hallucinated tracking data. Our experiments illustrate the ability to which we can accurately predict how specific teams will respond in specific game situations. As a result, coaches, fans and commentators can use our tool to explore an endless number of scenarios: sketching plays to find weaknesses in a particular defensive system, editing existing tracking data to see if a better decision could have been made, or evaluating similarities/differences between how teams react to the same attack.

References

[1] B. Walsh, S. Jamison and C. Walsh. "The Score Takes Care of Itself: My Philosophy of Leadership." Penguin Publishing Group, 2009.

[2] J. MacMullan. "Isaiah Thomas: Brad Stevens could be one of the greatest coaches who ever lived." ESPN, 26 Apr 2016.

[3] R. Mahoney. "Brad Stevens's After-Timeout Genius Keys Celtics' Game 3 Victory." Sports Illustrated, 21 May 2017.

[4] D. Eberhardt. "'ATO' geniuses: Breaking down why some teams execute great out-of-bounds plays." SB Nation, 12 Dec 2013.

[5] H. Le, P. Carr, Y. Yue and P. Lucey. "Data-Driven Ghosting using Deep Imitation Learning." MIT Sloan Sports Analytics Conference, 2017.

[6] FastModel Sports.

[7] Z. Lowe. "Lights, Cameras, Revolution." Grantland, 19 Mar 2013.
[8] H. Le, Y. Yue, P. Carr and P. Lucey. "Coordinated Multi-Agent Imitation Learning." International Conference on Machine Learning, 2017.
[9] Benjamin Morris. "Can The Warriors Break Basketball Again?" FiveThirtyEight, 3rd Nov, 2016.
[10] D. Cervone, A. D'Amour, L. Bornn and K. Goldsberry. "POINTWISE: Predicting Points and Valuing Decisions in Real Time with NBA Optical Tracking Data." MIT Sloan Sports Analytics Conference, 2014.
[11] Hooptactics. "How to Read & Interpret Play Diagrams".
[12] R. Zeleznik, T. Miller, L. Holden and J. LaViola. "Fluid Inking: Using Punctuation to Allow Modeless Combination of Marking and Gesturing." Graphics Interface 2006.
[13] P. Lucey, A. Bialkowski, P. Carr, S. Morgan, I. Matthews and Y. Sheikh. "Representing and Discovering Adversarial Team Behaviors Using Player Roles." Computer Vision and Pattern Recognition, 2013.
[14] basketball-reference.com

Appendix

A.1 Roles

Similar to [5], we organize the tracking data of each possession by strategic *role*, and not player *identity*. Basketball has a consistent set of established positions: center, point guard, shooting guard, small forward and power forward. We learn the template[7] of roles directly from data [13]. For each possession, we infer the role being fulfilled by each player by solving a linear assignment problem using the Hungarian algorithm. Unlike soccer, the template in basketball does not simply translate up and down the court (see Fig. 7). Instead, the role templates are inversions of each other, depending on whether the team is setup for offense or defense. In practice, we use a weighted combination of defense/offense templates where the proportional influence of each model is governed by a sigmoid distribution based on the centroid of the player positions.

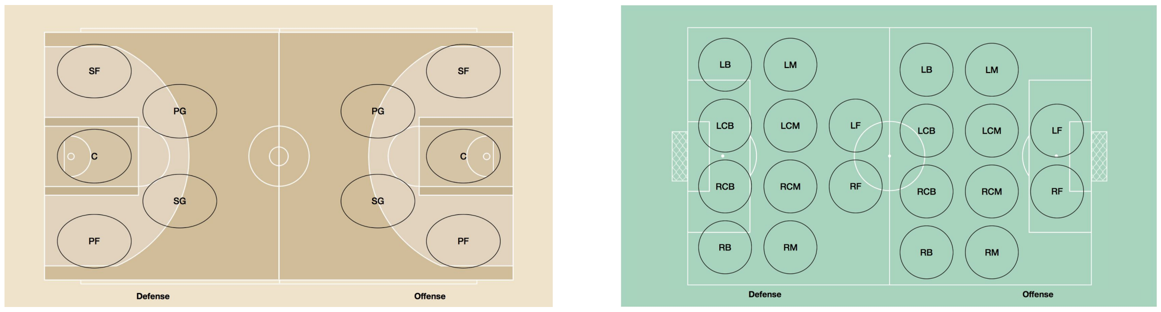

Figure 7: In soccer, the relative spatial arrangement of roles is the same regardless of whether the team is attacking or defending. In basketball, the formations are inverted. When assigning roles to the basketball tracking data, we employ a weighted combination of both templates so that sensible role assignments are generated for every frame of data

A.2 Features

Like soccer, a defender's decision making for where to go next on the court is strongly related to relative distances to the ball, the basket and other players. To simplify neural network training, we handcraft features that contain both the global cartesian coordinates (x, y) and relative polar coordinates (r, $\cos\theta$, $\sin\theta$). In addition to a "one hot" encoding of team identity, we compute additional context features that describe the state of the game (score, game clock, shot clock) and player-specific attributes[8] (fouls committed, cumulative "on court" time). The resulting feature vector is 247 dimensions, and encodes the state of the game from the perspective of a specific role.

[7] While beyond the scope of this paper, it is clear that this template is evolving as NBA teams increasingly utilize stretch-4s and small ball lineups.

[8] Since the starting five log the majority of minutes, team identity implicitly encodes many player-specific attributes like skill. Therefore, we only model dynamic player-specific properties like fatigue and fouls.

A.3 Game Evolution

Since the introduction of the the 3 point shot in 1980, the game of basketball has changed. The "modern" NBA has seen an increase in the number of three point shots attempted. On average, the 2012-13 NBA season had 20 three point field goals attempted, while the 2016-17 NBA season had 27 three pointers attempted [14], with the Houston Rockets attempting almost 40 three pointers per game [14] in the 2016-17 NBA season. This increase has therefore changed the way defenders guard 3 point of shots. Using data from 2012-13 season, we generated a model to predict defensive behavior and ran it on a test set from the 2016-17 season. Fig. 8 clearly shows the ghosts from the 16-17 season model guard the corner three much more aggressively.

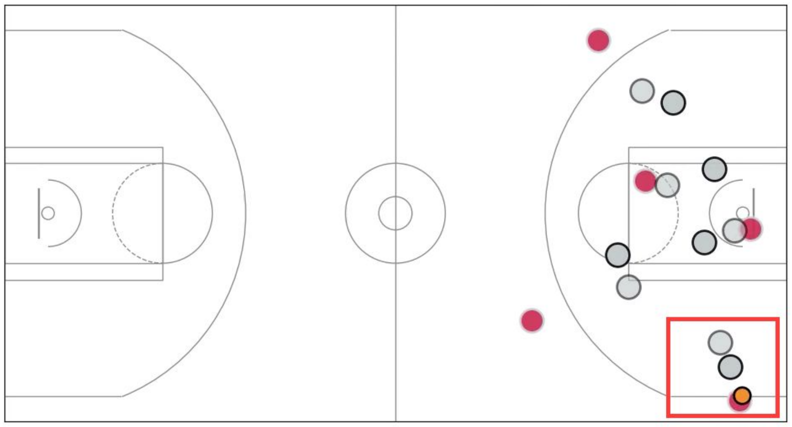

Figure 8: 2016-17 ghosts (grey solid) aggressively guard the corner three point shot as compared to the 2012-13 ghosts (grey transparent) [red circles - team on offense, orange circle - ball].

A.4 Ghosting Response to Sketch

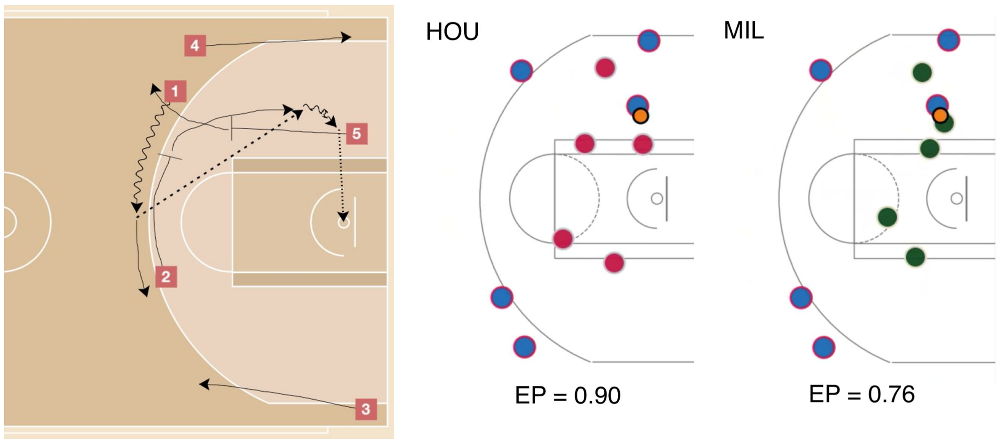

Figure 9: (left) A sketched play. (right) The defensive positions of Houston (red) and Milwaukee (green) at the time of the shot (offense in blue). The Rockets do not contest the mid-range jump shot, whereas the Bucks are anticipated to make an aggressive challenge (resulting in a lower expected points).