Analysis of Player Actions in Selected Hockey Game Situations

Fahong Li  Robert J. Woodham
Department of Computer Science
University of British Columbia
Vancouver, BC, Canada V6T 1Z4
{fhli, woodham}@cs.ubc.ca

Abstract

We present a proof of concept system to represent and reason about hockey play. The system takes as input player motion trajectory data tracked from game video and supported by knowledge of hockey strategy, game situation and specific player profiles. The raw motion trajectory data consists of space-time point sequences of player position registered to rink coordinates. The raw data is augmented with knowledge of forward/backward skating, possession of the puck and specific player attributes (e.g., shoots left, shoots right). We use a Finite State Machine (FSM) model to represent our total knowledge of given situations and develop evaluation functions for primitive hockey behaviours (e.g., pass, shot). Based on the augmented trajectory data, the FSMs and the evaluation functions, we describe what happened in each identified situation, assess the outcome, estimate when and where key play choices were made, and attempt to predict whether better alternatives were available to achieve understood goals. A textual natural language description and a simple 2D graphic animation of the analysis are produced as the output. The design is flexible to allow the substitution of different analysis modules and extensible to allow the inclusion of additional hockey situations.

1. Introduction

Many books on hockey coaching use diagrams and textual descriptions to represent and explain play formations and strategies [9, 11]. Instructional hockey videos [13] provide drill and selected game clips. Recently, computer software [14] has been developed to show play plans with graphic animation. All of these media help coaches and players enhance play. However, to our knowledge, there has been no framework which automatically or semi-automatically integrates play knowledge, video data and computer graphic animation to demonstrate how to play hockey effectively in various game situations. In such a framework, users (coaches, players and fans) could browse and build diagrams and textual descriptions illustrating effective play. They could watch actual play in video clips identified as instances of a particular situation. They could simulate alternative play graphically for more detailed interpretation of the actual play data extracted from video.

We have developed such a semi-automatic framework, with two long-term objectives. One is to produce an interactive, effective tool for hockey enthusiasts to illustrate and experiment with hockey strategies. The second is to produce a system that can automatically provide textual descriptions of play to annotate large databases of hockey video data. As our first step, we have focused on the representation for general hockey play, on the description and evaluation of particular play elements, and on the exploration of better alternatives to the actual play observed. The raw motion trajectory data is extracted automatically from video using software developed by others involved in the hockey analysis project. Manual intervention is required only in situations where the rink registration fails. In principle, knowledge of forward/backward skating, possession of the puck and specific player attributes also could be determined automatically. Currently, the raw data is annotated manually by the user. In the thesis from which this paper is drawn [7], examples from three well-defined game situations are considered: 2-on-1 offensive attack, power play shot from the point and defensive zone breakout. In this paper, we illustrate the work using only two examples: one for a simple 2-on-1 offensive attack and the other one for a simple 5-on-4 power play shot from the point.

In any 2-on-1 offensive attack, we can identify a start, an end and an outcome (good, bad or neutral) from the perspective of the offence (or the defence). We analyze player trajectory data, augmented with information on possession of the puck and players’ profiles (e.g., shoots left, shoots right). We describe where the two attacking players skate. Is it wider to set up a pass or does the puck carrier skate directly for a shot on goal? Throughout, we calculate the fea-
sibility of passing the puck, based on the geometric configuration of all three skaters and their profiles. If the puck carrier shoots towards the goal, we evaluate the effectiveness of a shot from that particular location, based on the distance between the location and the goal, the shooting angle, the position of the goalie, and the positions of the other two players. We also explore possible better outcomes by re-analyzing the situation assuming the puck carrier makes the choice (pass or shoot) at slightly different times (sooner or later) or at slightly different locations (e.g., nearer/further or at less/greater angle to the goal). Similarly, we can analyze play effectiveness of the attackers (or the defensive players) in any power play shot from the point.

We also provide a flexible architecture to support the addition of other analysis modules. Thus, in principle, we can analyze a given game situation from different perspectives (offence versus defence) and at different levels of detail (differences, say, in both general and specific skill levels of the individual players involved). The architecture also is extensible so that other hockey game situations can be incorporated once knowledge about play in those situations is represented in the format required.

In summary, we have developed a flexible and extensible system to reason about hockey play effectiveness. It takes as input players’ trajectory data, augmented by individual player profiles and labelling of skating mode (forward/backward, falling down) and possession of the puck. The system includes a Finite State Machine representation for general play knowledge, data structures for augmented trajectories, and a mechanism for generating play descriptions, for analyzing the actual play and for predicting alternative actions in the play. The system is specialized to hockey. We believe the design is generalizable not only to other hockey game situations but ideally also to other domains, such as soccer, basketball, traffic flow control, and people surveillance.

We organize the rest of this paper as follows. Section 2 lists relevant work. Section 3 presents the architectural overview and the assumptions we have made in this paper. Section 4 shows the results of analyzing the two examples mentioned above. Section 5 concludes. We discuss some implementation problems and point to future work.

2. Relevant work

The System SOCCER [1] automatically and simultaneously generates natural language description from soccer game video sequences (recorded with a static TV-camera) for listeners who were not watching the game. It takes as input geometrical description of the scene, including a stationary background and trajectories of dynamic objects. The trajectories are provided by a vision system with a special trajectory editor. Propositions about what is happening at the moment are produced by the incremental event recognition module. The selection/linearization component chooses relevant propositions, sorts them and passes them on to the encoding processes, which further transform non-verbal information in those propositions into ordered written or spoken words in German. All the processes in the system run in parallel. One limitation of the SOCCER system is that it cannot recognize collaborations among several players, such as a team’s attack. Players’ intentions are added into the output description of the SOCCER system by another component, REPLAI (REcognition of PLans And Intentions) [10], which argues that intention is the uncompleted parts of a plan that has been started. The knowledge about standard goals and plans in REPLAI is then formalized as a hierarchy.

Lashkia et al. [6] present a software tool for assisting team play analysis in soccer. Based on a probability model using colour information, it detects the ball and players in video sequences recorded with a single digital video camera, registers their camera coordinates to a field model, simulates the image scenes with 2D and 3D graphic animation, and then analyzes dominant areas of each team. As Lashkia et al. point out, this tool does not integrate players’ personal abilities and needs to improve the accuracy of detecting players and the ball when it is off the ground.

Kawashima et al. [5] propose an algorithm to analyze qualitatively group behaviour in soccer video sequences. The algorithm uses the histogram backprojection to detect and classify each player in a colour image. A group hierarchy is formed by applying Gaussian filters to the output image of the histogram backprojection. Finally an interpretation to the group behaviour is derived from the temporal aspect of qualitative relations (e.g., disconnected, overlapped, or part of) among groups. The interpretation does not include exact motion information such as velocity, direction, or its distribution.

We use the automatic player tracker developed by Okuma [8], another researcher involved the hockey project. Registration with Okuma’s tracker works well with panoramic overhead views because a sufficient number of rink landmarks will be visible in each video frame. Commercial broadcast video often includes close-up views. Manual registration adjustment proved necessary in sequences when there were frames in which not enough rink landmarks were visible. As an aside, we note that it also is feasible to get accurate trajectory data by instrumenting players, the puck and the rink itself, as was done in FoxTrax [15].

3. The framework and assumptions

Figure 1 is the architectural overview. Rectangles with dashed sides (e.g., Hockey Game Videos, Augmented Tra-
trajectory Data, Event Evaluation Functions) stand for storage items such as databases, data structures and function libraries; rectangles with solid sides (e.g., Digitize, Configure, Analyze, Visualize) represent activities or modules; solid arrow lines are flows of information items. Input includes hockey game videos and hockey domain knowledge. All the videos tested are recorded TV broadcast professional matches. Our domain knowledge is that which generally applies in professional matches. We focus on designing storage items and modules identified within the large box with dashed sides, which mainly include Augmented Trajectory Data (ATD), Sorted Play Events (SPEs), Finite State Machines (FSMs), Event Evaluation Functions (EEFs), Situation Analysis Modules (SAMs), Describe, Analyze, Predict, Configure and Visualize.

This paper makes the following assumptions:

- The video data have been pre-segmented and classified into individual clips belonging to one of the three identified game situations: 2-on-1 offensive attack, power play shot from the point and defensive zone breakout. We do this manually.

- For each segmented video clip, players’ (and the puck’s) trajectory data have been accurately acquired and registered to a rink model. We assume we can also extract players’ identity, players’ skating mode (i.e., forward, backward or falling down), and information on possession of the puck from the video clips. Given player identity, we assume we have access to ancillary player profile data which include individual attributes such as shoots left, shoots right, maximum speed, etc. We manually (or, at best, semi-automatically) augment trajectory data with player profile data, skating mode and puck possession information. We name the result Augmented Trajectory Data (ATD).

- High-level play events (such as pass, shot) have been recognized and extracted from the segmented video clips. We also do this manually in our experiments. Papers on video analysis addressing similar problems automatically in other sports [2, 4, 12, 16] have been published. We define the list of these play events as Sorted Play Events (SPEs).

- Knowledge on how to play in each identified game situations is provided as input to the framework in the form of a Finite State Machine (FSM) and associated Event Evaluation Functions (EEFs). EEFs evaluate the effectiveness (feasible or not) of players’ taking primitive actions (play events) under particular geometric configurations during some time interval. We have created examples for several game situations [7].

Situation Analysis Modules (SAMs) embody analysis strategies. An SAM consists of functions in three categories: description functions, analysis functions and prediction functions. They respectively describe the actual play, analyze possible better alternative actions, and predict results of potential changes. Functions belonging to the same category have the same signature (i.e., number, order and data types of input/output parameters). Within one SAM, an analysis function can use results produced by descrip-
tion functions, and a prediction function can access results produced by description functions and analysis functions.

We have implemented one FSM based puck carrier centered analysis strategy. In this strategy, the meanings of the words description, analysis and prediction (corresponding to the functionalities of the SAM’s description, analysis and prediction functions) are given as follows. Description is a parse of the SPE list according to the FSM selected by the user. The result is the transition path augmented with time information indicating when each transition in the path occurs in the video clip. Analysis identifies feasible alternative paths in the FSM which start from the same initial state as the path parsed in the description and end up in a final state whose outcome is as good as or better than the outcome in the actual path. Prediction follows the same transition path as in the description but varies the players’ spatio-temporal trajectories and checks whether there are more favorable outcomes resulting from these trajectory perturbations. Thus prediction is limited to determining whether a slightly different positioning or timing of what actually happened might have led to a more favorable outcome.

The Configure module provides users the interface (Figure 2) to dynamically assign general system options. For each of the built-in situations, users can choose which FSM to use in the analysis process and which evaluation function to call for each of the primitive play events. Users can also assign which description, analysis and prediction functions and their parameters (if any) to be used for the selected analysis strategy (SAM). Options for the Visualize module include whether to analyze and display in each frame the riskiness of passing the puck to the puck carrier’s teammates and the feasibility extent of shooting the puck towards the goal.

The Visualize module visualizes the ATD (either the original one or the one altered in the prediction process) in one window using a simplistic 2D graphic animation, synchronized with the video clip displayed in another window (see Figure 5). The riskiness of passing and shooting can also be visualized in the animation upon the user’s request (through the Configure module). Meanwhile, textual descriptions of the SPEs and the paths generated in the description and analysis process also are output.

4. Analysis of two hockey game situations

Consider one video clip of a 2-on-1 offensive attack as an example. In this clip (clip1) the 2-on-1 situation starts from Frame #72 (Figure 3(a)), when the attacker (Player #2) on the right begins to carry the puck forward in the neutral zone and the other attacker (Player #3) follows up on the left. The opposite defenceman (Player #1) keeps skating backward between these two attackers until Frame #152 (Figure 3(d)), when he falls down in front of the goal to block the potential pass from Player #2 to Player #3. It ends when the offensive team loses possession of the puck at Frame #164 (Figure 3(f)). We acquired the players’ trajectories, registered them to the rink coordinate frame, and augmented them with necessary information to form the ATD for this clip. Then we manually identified and sorted the higher-level play events (SPEs) of interest from the ATD. The FSM (Figure 4) representing play knowledge in a simple 2-on-1
offensive attack is provided as input, in a formatted text file. Figure 5 is part of the final output. The interface has 4 windows: the left window on the top shows the original video clip; the middle window on the top displays textual output; the right window on the top contains control buttons; the window at the bottom visualizes the ATD and the evaluation results. In the bottom animation window, areas (i.e., possible passing paths) filled with green horizontal lines represent intervals during which it is feasible to pass, areas filled with yellow diagonal crossing lines represent intervals risky to pass, and areas filled with red vertical lines represent intervals unwise to pass.

Major textual components of the description include:

At Frame #120, Player #2 crossed the blue line.
At Frame #152, Player #1 fell down.

At Frame #164, the offensive team (Player #2) lost possession of the puck.
The clip goes through the path:

1. q0[Frame #72]: The two attackers are outside their offensive blue line.
2. q1[Frame #120]: The puck carrier is inside his offensive zone.
3. q2[Frame #164]: The puck carrier lost possession of the puck.
The outcome is bad for the offensive team.

Major textual components of the analysis include:
According to the selected FSM and the return values of evaluation functions, i.e.,

Since Frame #73, EvPassV1() returns: 1.
Since Frame #114, EvPassV1() returns: 0.
Since Frame #115, EvPassV1() returns: 1.
Since Frame #116, EvPassV1() returns: 0.
Since Frame #118, EvPassV1() returns: 1.
Since Frame #119, EvPassV1() returns: 0.
Since Frame #120, EvPassV1() returns: 1.
Since Frame #123, EvPassV1() returns: 0.
Since Frame #130, EvPassV1() returns: 1.
Since Frame #130, EvShootV2() returns: 3.
Since Frame #133, EvPassV1() returns: 0.
Since Frame #134, EvPassV1() returns: 1.
Since Frame #136, EvPassV1() returns: 0.
Since Frame #137, EvPassV1() returns: 1.
Since Frame #138, EvPassV1() returns: 0.
Since Frame #139, EvPassV1() returns: 1.
Since Frame #141, EvPassV1() returns: 0.
Since Frame #156, EvShootV2() returns: 2.
Since Frame #160, EvPassV1() returns: -1.
the offensive team could have played the following path for a better outcome:
1. q0[Frame #72]: The two attackers are outside their offensive blue line.
2. q1[Frame #120]: The puck carrier is inside his offensive zone.
3. q5[Frame #130--155]: The puck carrier shot towards the goal in his offensive zone.

In the analysis process, two evaluation functions, i.e., EvPassV1() and EvShootV2(), are used to estimate possible passing/shooting path(s) and to check the feasibility of taking the two primitive play actions/events (PASS and SHOOT) from the offensive team’s perspective.

- EvPassV1() takes into account the players’ current geometric configuration, the defensive players’ defending coverages and the potential receiver’s most likely next position after a short fixed time interval. In the version we implemented, it returns 1, 0 or -1, indicating it is feasible, risky or unwise to pass respectively.
- EvShootV2() takes into account the distance between the puck carrier and the center of the goal line, the shooting angle (i.e., the acute angle between the goal line and the line decided by the puck carrier’s position and the center point of the goal line), and positions of both offensive and defensive players in front of the goal. It returns a value between 1 and 5, indicating to what extent it is feasible to shoot (a bigger value means a greater extent).

The evaluation results are visualized (frame by frame if applicable) and the visualization effect in every frame is kept from the start to the end in the bottom window of Figure 5. The possible pass path area is filled with horizontal lines in green, diagonal crossing lines in yellow, or vertical lines in red to represent the case of feasible, risky or unwise to pass respectively. The analysis results show the offensive team could have shot towards the goal after Frame #130 (Figure 3(c)), instead of making a potential pass, in order for a more favorable outcome.

As for prediction, we apply several predefined perturbations to the two attackers’ trajectories and redo the analysis. Figure 6 is the output after applying one of the perturbations. By comparing it with Figure 5, we can see that the feasible region (filled with green horizontal lines) for a pass has grown larger and longer and that the unwise region (filled with red vertical lines) for a pass has disappeared. This suggests a better outcome than what actually occurred for the offensive team. The animated versions of Figure 5 and Figure 6, as well as other examples, are available through <http://www.cs.ubc.ca/nest/lci/thesis/fhli/videoclips>.

Another example (clip3, Figure 7) is an instance of power play shot from the point. In this clip, the situation starts from one point player (Player #8) has possession of the puck at Frame #1 (Figure 7(a)) in a 5-on-4 power play formation. The puck carrier (Player #8) passes the puck to the other point player (Player #7) at Frame #90 (Figure 7(d)). The situation ends when the puck carrier (Player #7) shoots the puck towards the goal at Frame #181 (Figure 7(f)). Figure 8 is the FSM representing play knowl-

![Figure 7: Key image frames in clip3](image)

(a): the situation starts; (b), (c), (e): frames where the return value of EvShootV2() changes; (d): the puck carrier passes the puck to a teammate; (f): the puck carrier shoots towards the goal.

![Figure 8: The FSM for simple 5-on-4 power play shot from the point](image)
At Frame #90, Puck carrier (#8) passed the puck to his teammate (#7).

At Frame #181, The puck carrier (#7) shot towards the goal in his offensive zone.

The clip goes through the path:
1. q0[Frame #1]: The point player has possession of the puck.
2. q0[Frame #90]: The point player has possession of the puck.
3. q3[Frame #181]: The puck carrier shot towards the goal from the point.

The outcome is dependent on EvShootV2().

Major textual analyses include:

According to the selected FSM and the return values of evaluation functions, i.e.,

Since Frame #37, EvShootV2() returns:3.
Since Frame #41, EvShootV2() returns:2.
Since Frame #177, EvShootV2() returns:1.
Since Frame #181, EvShootV2() returns:2.

the offensive team could have played the following path for a better outcome:
1. q0[Frame #1]: The point player has possession of the puck.
2. q3[Frame #37-40]: The puck carrier shot towards the goal from the point.

It tells that the point player (Player #8) could have shot since Frame #37 (Figure 7(b)) instead of passing the puck to Player #7 at Frame #90 (Figure 7(d)) in order to get a more favorable outcome. The reason is that at Frame #37 (compared with Frame #181) no opponent player (except the goalie) was inside the possible shot path area in front of the goal; the algorithm we used in EvShootV2() treats this as an opportunity for the offensive team to score. No perturbation/prediction is applied to this clip.

5. Conclusion, discussion and future work

We have presented a flexible and extensible framework to describe, analyze and predict player actions in selected game situations, based on Augmented Trajectory Data, Sorted Play Events, Event Evaluation Functions, and Finite State Machines which represent play knowledge in those situations. The results show that augmented player trajectory data support description, analysis and prediction in hockey game situations. Although our FSM model does not represent the complete range of feasible geometric arrangements of players in each situation, it is adequate to represent specific game situations and the discrete events/actions that occur in those situations. With event evaluation functions, we can identify alternative paths in the FSM which might result in better outcomes than the actual one if the players played the situation by following those paths. Analysis of alternative geometric positioning of the players can predict more advantageous outcomes for a given path through the FSM. This analysis is limited in depth in that we do not predict how players on the other team might react to the alternative positioning proposed. The framework provides a foundation for development of more elaborated hockey analysis applications.

We manually recognized the higher-level play events that occur in each clip. We have not yet attempted to develop an automatic event recognizer. Our event evaluation functions make general assumptions about player skill level. Overall, we assume a high skill level. We do not yet take into account differences in skill level among individual players. (Our event evaluation functions also would be different for a system designed for novice hockey. A pass or shot that is feasible for a professional is not necessarily so for a ten year old player). Our goal was to provide a basic proof of concept, which we have done. At the same time, we believe that it is feasible to build and to use a database of individual player skating, defending and/or attacking skills. This would help to customize the evaluations for specific groups of players. Different FSMs can be defined for the same game situation. This is desirable since description, analysis and prediction certainly depend on the perspective and the skill level of each player involved. They also depend on the particular coaching style and strategies employed. Each FSM is one view of a game situation. It is not expected that analysis based on a single FSM will be relevant to all parties. Ultimately, we expect users to be able to easily build their own FSMs so that a given video clip can be analyzed
from different perspectives and at different levels of granularity.

A program which can predict a player’s next action on the fly will interest more people. We may achieve this by seeking a better spatio-temporal configuration for the offensive (or defensive, whichever is the analysis perspective) players to complete their understood goals. However, it is hard to identify on the fly what are the players’ understood goals and how they will react to opponent players’ interference. In an ideal case, i.e., if the event evaluation functions are continuous, we can find an improved configuration by estimating the spatio-temporal gradients of these functions. Nevertheless, we can perturb the puck carrier’s (or one of the key players’) current position to somewhere nearby that can be reached by the player within a short time interval $\delta_t$, given her or his maximum skating speed. Assuming other players keep still or moving with their own previous velocities, now we get a new players’ geometric configuration after $\delta_t$. Then we can call the event evaluation functions to redo the evaluation. One way to choose a reasonable position perturbation is to introduce uncertainty and use statistical methods to find the most likely positions the player will move to based on the player’s history of play behaviour. Another one is to allow the user to adjust players’ positions interactively and show the evaluation results on the fly in the 2D graphic animation. In our experiments, several fixed perturbation modes are defined and each of them perturbs key players’ positions according to heuristic factors which play a role in computing the event evaluation functions.

We would also like to develop more analysis strategies (SAMs) such as using multiple FSMs in parallel to deal with complicated game situations. Statecharts [3] may be introduced to supersede FSMs when the multitude of states grows exponentially and/or when the notions of hierarchy, concurrency and communication among game situations need to be dealt with to a much greater degree of granularity.

Finally, we want to extend the system in two separate ways. First, we want to enhance the user interface by providing user-friendly tools to input, debug and summarize hockey domain knowledge. We need better dynamic illustration of overlapping analysis and prediction results on sequentially displayed image frames. We would like to use advanced 3D (rather than toy 2D) graphic animation in the animation window. Second, we want the system to function in batch mode, without graphic user interface. Batch mode is required to process large volumes of video data autonomously to add appropriate textual description, analysis and prediction annotation to a very large video database.

Acknowledgements: Support for the work described in this paper was provided by NSERC and by the IRIS TRA project, Raymond Ng (Project Leader). The authors thank Jim Little, David Lowe, Valerie McRae, Kenji Okuma and Michael Zhang for their contributions.

References