Detecting Attack Patterns in Football Trajectory Data & Home Region Computation

Frank Staals[*] *Joachim Gudmundsson*[§]

This is a report of a Reciprocal Short Term Scientific Mission (RSTSM) within the framework of COST Action IC0903: MOVE. During this STSM Frank Staals visited Joachim Gudmundsson in Sydney. The goal of the RSTSM was to develop efficient algorithms to aid in automatic analysis of football trajectory data. Some progress was made on this topic. However, the techniques used are not sufficiently innovative to be published at a conference or journal. Thus, part of the time was spent on a different topic, namely that of computing the home-region of a moving entity, based on its trajectory. Although more work is required in this topic, our initial results are promising.

Since we worked on two topics, the remainder of this document is split into two parts. The first part is about the automatic football analysis. The second part about home region computation.

1 Detecting Attack Patterns in Football Trajectories

In the last ten years there has been a lot of research in extracting player and ball trajectories from videos of football matches [6, 8, 11, 13]. These trajectories can provide a wealth of information to coaches, scouts, and media. We study how to support automatic analysis of such football trajectory data using tools from Computational Geometry. In particular, we investigate detecting attacks and attack patterns in the trajectory data. The questions that we study are:

- When is team A attacking?
- Which players are participating in this attack?
- How many attacks of type X are there?
- Which attacks are most similar to this one?

Although we will always refer to football trajectories, we note that our methods are also applicable in the analysis of attacks, or more general, collective motion, in other ball-sports like hockey, rugby, (water)polo, etc.

Modeling. We start by formalizing the terms involved. We consider the trajectory data of a football match between two teams, a *red team* $\mathcal{R}$ and a *blue team* $\mathcal{B}$. We denote the trajectories of the red and blue players by $R_1, .., R_r$ and $B_1, .., B_b$, respectively. The trajectory of the ball is denoted by ⚽. Since segmenting football trajectories into semantically meaningful pieces or *moves*, for example passes, throw ins, shots on goal, has been studied before [8], we can consider the trajectories as sequences of such basic moves. With slight abuse of notation we will also write T for the sequence of moves of trajectory T.

[*]Department of Information and Computing Sciences, Utrecht University, f.staals@uu.nl
[§]School of Information Technologies, University of Sydney, joachim.gudmundsson@sydney.edu.au

We now define a *ρ-attack* $\mathcal{A}$ of the blue team as a subtrajectory of the ball ⚽ that has the following properties:

(i) when $\mathcal{A}$ starts and ends the blue team has the ball,

(ii) $\mathcal{A}$ ends in the *goal region* $\mathcal{G}_\mathcal{B}$,

(iii) the blue team has the ball for at least $\rho\|\mathcal{A}\|$ time during $\mathcal{A}$, where $\|\mathcal{A}\|$ denotes the length (in time) of attack $\mathcal{A}$,

(iv) $\mathcal{A}$ is maximal with respect to these properties.

Since $\mathcal{A}$ is maximal this means $\mathcal{A}$ is a proper subsequence of the basic moves of ⚽, that is, $\mathcal{A}$ starts with a move in which blue receives the ball, and ends with a move after which blue loses the ball. The goal region $\mathcal{G}_\mathcal{B}$ can be any region of the field. For example, a radius r-disc around the goal of the red team. The only restriction we impose on $\mathcal{G}_\mathcal{B}$ is that we can efficiently determine if a point lies inside or outside of $\mathcal{G}_\mathcal{B}$. Finally, the parameter ρ allows us to model robustness: we allow that the blue team temporarily loses control of the ball during the attack. Attacks of the red team are defined analogously.

To answer the first two questions we now need to find all attacks of the blue (red) team. Once we have this information, it is easy to compute which players are associated with the attack.

Results. In case $\rho = 1$, that is, we require the blue team to have the ball during the entire attack, we can use a simple greedy algorithm to compute all attacks in $O(n)$ time, where n is the number of basic moves in ⚽. In the general case we can no longer use this greedy approach. Instead, we use an approach using the *start-stop diagram* introduced by Aronov et al. [1]. The start-stop diagram captures all possible subtrajectories of ⚽ as the upper half of the unit square (see Fig. 1).

In our case the segmentation of the trajectory into basic moves partitions the diagonal of the start-stop diagram into pieces. Each piece corresponds to a single move, and is either red or blue, depending on the team who has control of the ball. We can overlay the start-stop diagram with a grid derived from these pieces. Since every attack is a maximal subsequence of moves, its subtrajectory corresponds to a vertex of the grid.

Each grid vertex is either *free*, in case its subtrajectory satisfies properties (i), (ii), and (iii), or *forbidden* otherwise. The set of free vertices now characterize all attacks. Let n denote the number of basic moves in the ball trajectory. The grid has complexity $O(n^2)$. If we process the vertices starting with the vertices closest to the diagonal, and move towards the top left corner then we can check in $O(1)$ time per grid vertex if conditions (i) and (iii) are satisfied. Suppose that querying whether or not a point lies in $\mathcal{G}_\mathcal{B}$ takes $O(Q)$ time, then the total running time is $O(n^2Q)$. When $\mathcal{G}_\mathcal{B}$ is simply a disc such queries take constant time, and hence we can compute all attacks in quadratic time.

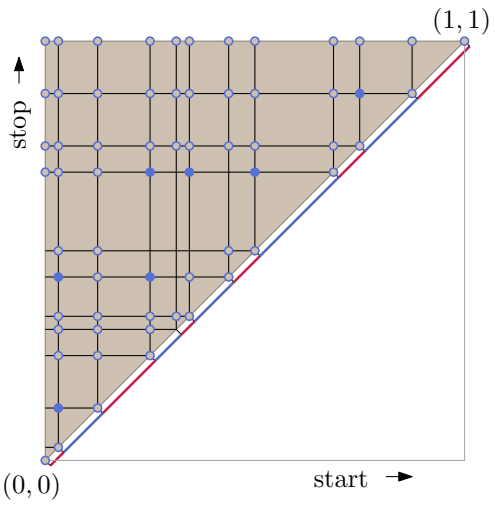

Fig. 1: The start-stop diagram. Forbidden space is indicated in brown. The free vertices are the blue filled dots.

We can now compute all attacks, however some of these attacks may still overlap, or there may even be attacks that are contained in an other attack. We may be interested in selecting a set of non-overlapping attacks that maximizes the total attack time.

We model this using a *conflict graph*: a weighted graph $G = (V, E, w)$ in which we have a vertex $v_{\mathcal{A}} \in V$ with weight $w_{\mathcal{A}} = \|\mathcal{A}\|$ for each attack $\mathcal{A}$. Two attacks $v_{\mathcal{A}}$ and $u_{\mathcal{C}}$ share an edge if and only if their attacks do not overlap, that is, $\mathcal{A} \cap \mathcal{C} \neq \emptyset$. See Fig. 2 for an illustration. The problem now reduces to computing a maximum weight independent set. Since G is an *interval graph* we can compute such a maximum weight independent set in time linear in the number of vertices [10]. Since we may have $O(n^2)$ attacks, computing a set of non-overlapping attacks takes $O(n^2)$ time as well.

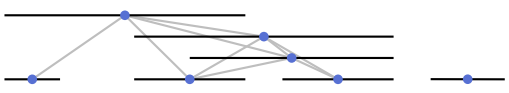

Fig. 2: The conflict graph for the free vertices in Fig. 1.

By using the start-stop diagram approach we always spend at least quadratic time. We investigated if there is an output sensitive algorithm with subquadratic running time. So far, we did not manage to find such an algorithm.

Attack similarity. For the third and fourth questions we need an similarity measure between attacks. However, it seems difficult to characterize what it means for two attacks to be similar. It may be required to involve a domain expert in order to answer these questions.

Implementation. The above algorithms are currently not implemented. However, this should be an easy task. Since the segmentation of the ball trajectory into basic moves is already available, we only need to implement the algorithm to compute all attacks, and an algorithm to compute an independent set on an interval graph. For both algorithms we can simply use dynamic programming. Hence, this should be straight forward.

2 Home Region Computation on Trajectory Data

Two important concepts in ecology are *home range* and *territory*. The home range of an animal, or a species, is the area where it lives and travels in [4]. The territory is a sub area of the home range over which the animal has exclusive or priority use, that is, the area that it consistently defends against others [2].

Several methods exist to compute an animal's home range from a set of points in the plane. The most basic approach is to simply take the convex hull of the set of points. More advanced methods use kernel density estimations [14], α-hulls [3, 5], k-nearest-neighbors [7], or even Brownian bridges [9]. Nowadays, many animals are tracked using systems such as GPS. This provides researchers a wealth of trajectory data capturing the movement of the animals. All methods mentioned above consider the input to be a set of points in the plane. An obvious approach is to simply pick a number of sample points on the trajectory, and use this set of points to compute the home range. However, to get an accurate image of the home range we may need a lot of sample points, thus increasing the running time of the algorithms. A second issue arises if we want to compute an animal's territory. We now may need to know how much *time* an animal spends inside a region, an aspect which has mostly been ignored in the home range computations.

We present efficient algorithms to compute a *home region* from the trajectory data: a region in which the animal spends most of its time.

Modeling. Given a piecewise linear trajectory T in $\mathbb{R}^2$ describing the movement of an entity ϱ, we wish to compute the home region $\hat{\square}$ of ϱ. We will focus on the case $\hat{\square}$ is a square, and extend our results to other shapes later. Let $E[T]$ denote the set of n edges of trajectory T. We assume that on each edge $e \in E[T]$ the speed, and thus the pace γ_e, is constant.

Let $\hat{\square}(c, r)$ denote the square with center c, and side length $2r$. We refer to r as the *radius* of $\hat{\square}(c, r)$. We denote the total trajectory length inside $\hat{\square}(c, r)$ by $H(c, r) = \sum_{e \in E[T]} \gamma_e \|e \cap \hat{\square}(c, r)\|$, where $\|\overline{pq}\|$ denotes the length of line segment $\overline{pq}$.

In case either c or r is clear from the context, we will simply write $\hat{\square}(r)$ and $\hat{\square}(c)$, respectively. We do the same for H. We now study the several problems related to the computation of $\hat{\square}$ and H.

Given Center and Given Radius. In the easiest case we are given a home region $\hat{\square}(c, r) = \hat{\square}$, and we simply want to compute $H(c, r)$.

Consider T as a curve in $\mathbb{R}^3$, where the z-coordinate represents time. To compute $H(c, r)$ we find (the z-coordinates of) all intersections of T in $\mathbb{R}^3$ with the vertical square cylinder $C = \{(x, y, z) \mid (x, y) \in \hat{\square}\}$ that we obtain by extending $\hat{\square}$. Computing H is now straight forward. All of this can be done in $O(n)$ time.

Given Center and Given Minimum Length. We are given the center c of the home region, and a minimum threshold δ on the trajectory length inside the home region. The goal is to compute the minimum radius r for which $H(c, r) \geq \delta$.

We order all vertices on increasing distance from c. Let $v_1, .., v_n$ denote the vertices in this order, and let $r_1, .., r_n$ denote the smallest r values for which vertex v_i lies on the boundary of $\hat{\square}(c, r_i)$. Using binary search we find the values r_i and r_{i+1} such that $H(c, r_i) < \delta$ and $H(c, r_{i+1}) \geq \delta$.

In between r_i and r_{i+1} $H(c, \cdot)$ is a piecewise linear function, that changes when the set $E_{\hat{\square}}$ of edges that intersects $\hat{\square}$ changes. We iterate through these pieces/sub intervals, maintaining a description of $H(c, \cdot)$. This allows us to find the r value for which $H(c, r) \geq \delta$ in $O(n \log n)$ time.

Unknown Center and Given Radius. We are given the radius r. The goal is to compute the center c of a home region $\hat{\square}(c, r)$ that maximizes $H(c, r)$, that is, the length of the trajectory inside the home region.

We present two solutions for this variant of the problem: a $(1+\varepsilon)$-approximation algorithm in the radius, and an exact algorithm.

The basic idea behind the $(1 + \varepsilon)$-approximation algorithm is to select a set of points P, and compute $H(p, r(1+\varepsilon))$ for each $p \in P$. We then simply take the point that maximizes H. We can show that we can easily construct such a set P such that there always is a point $p \in P$ that is close to an optimal center c^* (that is, a point c^* such that $H(\cdot, r)$ is maximal). Therefore, the optimal home region $\hat{\square}^* = \hat{\square}(c^*, r)$ is contained in $\hat{\square}(p, r(1 + \varepsilon))$, and hence $H(c^*, r) \leq H(p, r(1 + \varepsilon))$.

For the exact algorithm we construct a subdivision of the plane into maximally connected regions such that for all points in a region, the set of edges intersected by the home region is the same. This means we can compute a description of $H(\cdot, r)$ in each face of this subdivision. By computing $\frac{\partial z}{\partial c} H(c, r)$ we can find an optimal placement in each face, and thus a (global) optimal placement.

Unknown Center, Unknown Radius, and Given Minimum Length. In the most advanced case we are only given a minimum threshold δ on the trajectory length inside the home region. The goal is to compute a (the center c of a) home region $\hat{\square}(c, r)$ with minimum radius r for which $H(c, r) \geq \delta$.

The basic idea is that to use the algorithm for unknown center and given radius as a decision algorithm, and use parametric searching to find the minimum radius r[12]. To turn this into an efficient algorithm we develop a parallel version/implementation of the "unknown center and given radius" algorithm.

Other Home Region Shapes. Most of our algorithms can be generalized to home regions of different, but given, polygonal shapes. Using more complex shapes will increase the running time of the algorithms. We are planning a more detailed analysis in the near future.

Such a generalized algorithm would also lead to a $(1 + \varepsilon)$-approximation algorithm for the case that the home region is a disk. A more detailed analysis will be given at a later time.

Future Work & Expected Research Outcome

We are currently in the stage of writing down the details and proofs for the above algorithms. With the exception of our the parametric searching algorithm, our algorithms should be relatively easy to implement. We are planning to do so, and experimentally evaluate the results.

The algorithm that simply computes H should be an easy implementation exercise. Once we have that algorithm, the $(1 + \varepsilon)$-approximation algorithm for the unknown center case should also be very simple. Our exact algorithm for the unknown center, given radius, problem is more difficult. However, since it uses the same techniques as for the given-center, unknown radius algorithms. That may algorithm may serve as a stepping stone. The parametric searching algorithm will be very hard to implement. However, instead we may consider using a simple binary search. This will not give us an exact algorithm, and the running time will be larger than that of the parametric searching algorithm. However, we expect that the home region found by this algorithm should not be too different from an optimal solution. Finally, we note that in terms of implementation it may even be possible to use the $(1 + \varepsilon)$-approximation algorithm as the decision algorithm in the binary searching.

One further option for future work is to investigate if and how our results can be extended to trajectories in $\mathbb{R}^3$.

We expect at least one publication as a direct result of this research. We will most likely target algorithms conferences like ISAAC and WADS, or GIS conferences like ACM SIGSPATIAL. We consider the RSTSM to be a starting point of our joint research, which we hope will result in further collaborations and publications.

References

[1] B. Aronov, A. Driemel, M. van Kreveld, M. Löffler, and F. Staals. Segmentation of trajectories on non-monotone criteria. In *Proc. 24th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1897–1911. SIAM, 2013.

[2] L. Boitani and T. Fuller. *Research Techniques in Animal Ecology: Controversies and Consequences*. Columbia University Press, 2000.

[3] M. A. Burgman and J. C. Fox. Bias in species range estimates from minimum convex polygons: implications for conservation and optiyons for improved planning. *Animal Conservation*, 6: 19–28, 1 2003.

[4] W. H. Burt. Territoriality and home range concepts as applied to mammals. *Journal of mammalogy*, 24(3):346–352, 1943.

[5] H. Edelsbrunner, D. G. Kirkpatrick, and R. Seidel. On the shape of a set of points in the plane. *IEEE Trans. Inform. Theory*, IT-29:551–559, 1983.

[6] A. Fujimura and K. Sugihara. Geometric analysis and quantitative evaluation of sport teamwork. *Systems and Computers in Japan*, 36(6):49–58, 2005.

[7] W. M. Getz and C. C. Wilmers. A local nearest-neighbor convex-hull construction of home ranges and utilization distributions. *Ecography*, 27(4):489–505, 2004.

[8] J. Gudmundsson and T. Wolle. Football analysis using spatio-temporal tools. In *Proc. 20th International Conference on Advances in Geographic Information Systems*. ACM, 2012.

[9] J. S. Horne, E. O. Garton, S. M. Krone, and J. S. Lewis. Analyzing animal movements using brownian bridges. *Ecology*, 88(9):2354–2363, 2007.

[10] J. Y. Hsiao, C. Y. Tang, and R. S. Chang. An efficient algorithm for finding a maximum weight 2-independent set on interval graphs. *Information Processing Letters*, 43(5):229–235, 1992.

[11] C.-H. Kang, J.-R. Hwang, and K.-J. Li. Trajectory analysis for soccer players. *2010 IEEE International Conference on Data Mining Workshops*, pages 377–381, 2006.

[12] N. Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *J. ACM*, 30(4):852–865, 1983.

[13] T. Taki and J. ichi Hasegawa. Visualization of dominant region in team games and its application to teamwork analysis. In *Computer Graphics International*, 2000.

[14] B. J. Worton. Using monte carlo simulation to evaluate kernel-based home range estimators. *The Journal of wildlife management*, pages 794–800, 1995.