ggViz: An Interactive Esports Analytics System

Peter Xenopoulos
New York University
Brooklyn, NY
xenopoulos@nyu.edu

João Rulff
New York University
Brooklyn, NY
jlrulff@nyu.edu

1. Introduction

Esports is one of the fastest growing sports in the world, attracting viewership as high as the FIFA World Cup or College National Football Championship. Accordingly, there is increased interest from investors looking to create teams, gamblers seeking to wager, and media conglomerates pursuing new, captivated audiences [9]. As esports have grown, there is an increasing need for analytical tools and methods, which have successfully proliferated in conventional sports like baseball, basketball, soccer and American football. Thus, the increased analytical focus has revolutionized team operations and fan experience.

While most previous work in esports has focused on League of Legends (LoL), there is a dearth of methodology and tools that are competitive first person shooter games specific. For example, consider Counter-Strike: Global Offensive (CSGO) – even though the game has the second biggest prize pool and the most tournaments out of any other esports [3]. Additionally, CSGO is the most watched esports on Twitch, popular streaming platform [1]. Despite the public interest, there have been little to no academic works concerning CSGO, especially in visualization. Part of the lack of academic work in CSGO has been due to data availability problems.

Previous work in sports analytics has focused on the development of models used to predict outcomes. For example, [20] present a model to predict win probability in Counter-Strike: Global Offensive (CSGO) rounds. Typically, these win probability models use low level events, such as damages or bomb plants, to drive player valuation and analysis. However, these models fail to capture high level "macro events", such as coordinated player movements that describe tactics. Because tactics are so complex, they are also hard to identify solely through a numerical technique, like clustering. Additionally, they are rarely annotated in the data. Understanding tactics are important as different combinations of tactics may lead to differing distributions of outcomes. Understanding these outcome distributions is important for teams to adjust their play style, for gamblers to properly orient their odds and bets, and for media companies to generate unique content for fans.

In this paper we introduce *ggViz*, a visual analytics system which guides a user to understand the effect of teams tactics. We provide a system that utilizes coordinated views to not only help users annotate interesting tactics, but also search the space of similar tactical events. Furthermore, we provide a collection of unique charts and views that users can explore spatiotemporal CSGO data. We organize the paper as follows. In section 2, we cover relevant background, such as related work on sports analytics and visualization, along with an explanation of Counter-Strike game mechanics. Section 3 covers our win probability methodology. In section 4 we discuss the ggViz system. In section 5 we show a few case studies outlining the system and further validating the underlying win probabiltiy model. Finally, in section 6 we conclude the paper and provide avenues for future work.

2. Background

2.1. Related Work

Tactic identification and exploration has a varied literature among conventional sports. Bialkowski et. al. [5] present methodology for extracting tactics from spatiotemporal tracking data in soccer. To do so, they dynamically update each player's role during each frame of the game. To discover each player's role, they use a minimum entropy data partitioning method. As data access increased, visual systems became favored for tactic identification. For soccer, Stein et. al. [17] present a visual analytics methodology to identify tactics on set plays. They find that their visual methods allow an analyst to generate global strategies to attack and defend free kicks. Recently, Wang et. al. [19] introduce a tactic exploration tool for table tennis based on simulated data. In order to identify the tactics, the users had to visually inspect the sequences of events to find interesting tactic combinations. For a comprehensive survey of visualization in sports, see Perin et. al [15].

Concerning esports, Li et. al [11] introduce a visual system to identify player and team behavior in Massive On-

line Battle Arena (MOBA) games. They find that their system allows an analyst to uncover player strategies over the course of a match. In follow up work, Li et. al. [10] also introduce a system to visually identify and learn interesting segments of a match. While there has been work on MOBA games, such as LoL, first person shooter (FPS) games are virtually untouched by visualization community, to the best of our knowledge. We seek to bridge that gap with this paper.

Estimating the win probability of a round or match in esports is an emerging area of research. Yang et al. [21] first estimated win probabilities in Defense of the Ancients 2 (DOTA2) using logistic regression, considering both real time and pre-match factors in their model. Hodge et al. [8] attained similar performance on DOTA2 data and suggested that ensembles may provide the best performance when trying to predict win probability. For Counter-Strike: Global Offensive (CSGO), Makarov et al. [12] presented an ensemble approach using TrueSkill, a method commonly used to rank players and teams in gaming, decision trees and logistic regression to predict round winners. Specifically, they employ TrueSkill to produce pre-round odds, and then decision trees, ensembled with logistic regression, for post-plant scenarios. One of the advantages of their approach is that it allows for the assignment of individual player ratings via TrueSkill. However, more spatially informed models have proven to perform better. The first attempt to use spatiotemporal information for CSGO win probability modeling was in Bednarek et al. [4], which utilized the spatial information to cluster death locations to create player ratings. They argue that encounters vary in importance by location and cluster deaths using k-means to create death heatmaps. Xenopoulos et. al. [20] present the most recent attempt to model round win probability for CSGO. They draw from win probability modeling methodology in contemporary sports, and utilize a large and unique set of CSGO matches, to estimate continuous-time win probability in CSGO rounds.

2.2. Game Mechanics

Counter-Strike: Global Offensive is the third rendition of the Counter-Strike series, and has been a mainstream esport for roughly two decades. The game is a first-person shooter (FPS) which features two sides, the counter-terrorist (CT) versus the terrorist (T), and takes places on a variety of maps, which are virtual worlds. The goal of the CT side is to defend two bombsites, designated A and B. The goal of the T side is to plant a bomb at one of the two sites. Either team can win by eliminating the members of the other team.

A professional game is structured as follows. Typically, two teams decided on a set of maps they wish to play, and typically choose one, three or five maps. Then, teams play 15 rounds as either T or CT, and then the teams switch sides. The first team to win 16 rounds wins the map. The team that wins the majority of maps wins the game.

At the beginning of each round, teams buy weapons, such as guns and grenades, along with armor. Each kill or action a player commits, such as planting or defusing the bomb, grants them money. A side also gains a money bonus every round, whose value is determined by if the side won or lost the previous round. The amount of money a team has is called their *economy*.

3. Win Probability

Xenopoulos et. al. [20] show how a team's chance of winning a round, their *win probability*, can be estimated at any point of the round by using *game states*. A game state is essentially a freeze frame of the round at a particular moment in time. A game state has a collection of attributes, such as each team's remaining players, their health points and each team's distance from the bomb site, among others. Using a game state as a unit of observation, we can build a model to predict the outcome of the round. Specifically, we use the following features for a game state:

Map: the map where the game state occurred

Ticks Since Start: how many ticks since the round start

Equipment Value: the total round start equipment value for T and CT

Players Remaining: the total numbers of players remaining on T and CT

HP Remaining: the total health points remaining for T and CT

Bomb Planted: a flag indicating whether or not the bomb is planted

Bomb Plant Site: if the bomb is planted, at which site (A or B)

Team Bombsite Distance: Closest player distance to A and B bombsites for T and CT

3.1. Data Acquisition

First, we need to extract CSGO data. Each map in a game produces a *demofile*, which is a binary file that can be parsed to extract game events. Game events are written to the demofile every *tick*, which is a unit of time equal to roughly .007 seconds. After each tick, the updates (events) sent to the game server are synced with the connected clients (the players). We use the `csgo` package in Python to parse over 3,000 professional games since 2016[1]. We downloaded these demofiles from HLTV.org, a leading CSGO fan site.

[1] `csgo` package GitHub (https://github.com/pnxenopoulos/csgo)

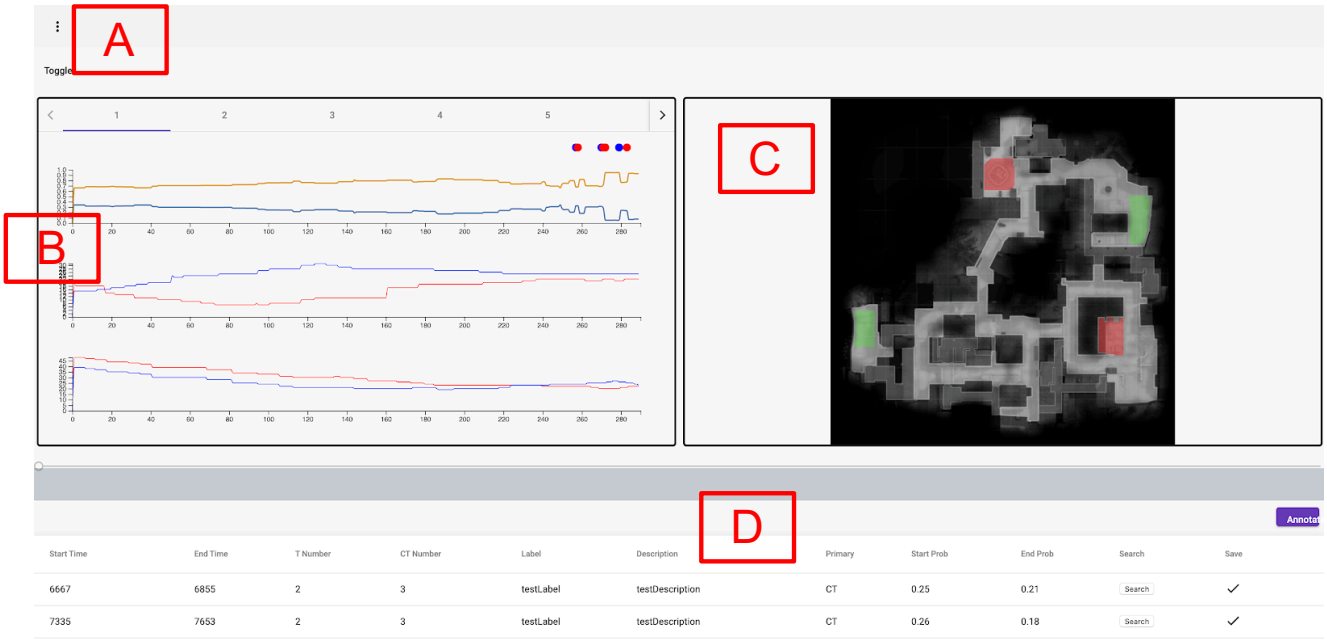

Figure 1. ggViz interface for the first round of a match between Astralis and Natus Vincere, two top teams. **A** shows the game and round selection tool. **B** shows the win probability graph and the bombsite distance graphs for the selected round. **C** is the map view for the round, which shows trajectories up to the selected frame. The view's selected frame can be chosen via the slider underneath the graphs and the map view. **D** shows the annotation interface, where users select on a timeline the start and end points of an event. Below, they can search the round's annotated events and find similar ones.

We used only matches that occurred over a local area network (LAN). These matches are usually indicative of a high level of play and lack networking issues that may affect gameplay for online matches. We parse the demofiles for different events, such as damages, bomb plants or player movement, to construct of 70 million game states. For more information on game states and a CSGO data model, see [20].

3.2. Model

To model win probability using game states, we consider an assortment of models, such as logistic regression, CatBoost [16] and XGBoost [7]. We focus on the latter two, CatBoost and XGBoost, since boosted ensembles have a demonstrated tendency to produce well calibrated probabilities [13, 14]. Additionally, models like CatBoost and XGBoost handle categorical variables well, and we have an collection of categorical variables such as map and bombsite. Since we are interested in estimating a probability, we define performance using both the log loss and Brier loss, which are standard in probabilistic prediction problems, as well as AUC [18]. We consider a benchmark model which predicts a game state's map average CT round win rate. Our data is set up such that each data frame k from round j in game i, $F_{i,j,k}$, is assigned $Y_{i,j} = 1$ if round j was a CT win and $Y_{i,j} = 0$ if the CT side lost the round. We used $\frac{2}{3}$ of games in our data for training and $\frac{1}{3}$ to test. We show the results of our model training in Table 1. We find that XGBoost performs the best across all metrics, and each model significantly improves above the map average, which proves to be a poor predictor of round outcomes.

Method	Log Loss	Brier Score	AUC
Logistic Regression	0.5539	0.1912	0.7743
CatBoost	0.5443	0.1875	0.7851
XGBoost	**0.5353**	**0.1842**	**0.7913**
Map Average	0.6917	0.2493	0.5303

Table 1. Win probability model performance on test data. XGBoost outperformed all other candidate models across all metrics.

4. ggViz System

In this section, we present and outline the various components of our system, ggViz. We present the interface of ggViz in Figure 1. Our interface includes a game selection tool **A**, win probability and bombsite distance graphs **B**, a map view for player trajectories **C** and an annotation tool for marking events **D**.

To use the system, the user first selects a match by clicking the three dot icon in the upper left. The following interface, in Figure 2 appears, and a user can search the database

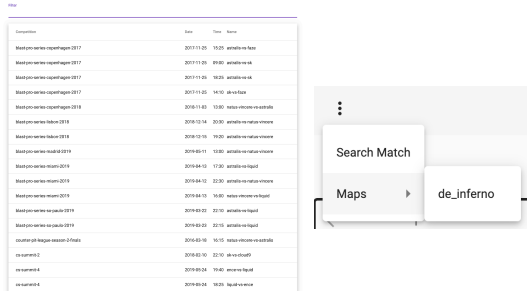

Figure 2. The game search and map selection interface allows the user to narrow down their search to a specific game-map pair.

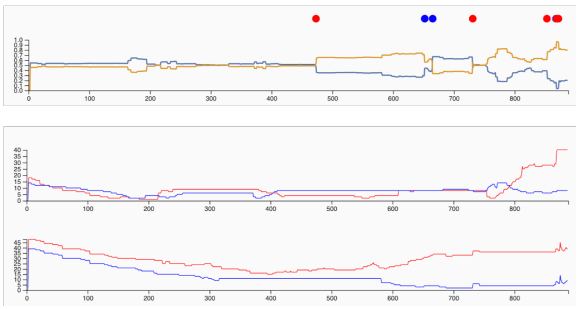

Figure 3. Win probability graph (top) along with bombsite distance graphs (bottom 2, A bombsite on top, B bombsite on bottom)

of games, which autofills based on the user's query. Additionally, once a round is selected, the user can change what map to view from that game. After a user selects the game and map, the rest of the views are populated.

4.1. Win Probability and Bomb Distance Graphs

One of the primary uses of ggViz is that a user can observe the changes in win probability, gain an understanding of the context via the map view, and mark interesting events in the annotation view. Thus, we present the win probability graph in the upper left. We show an example win probability graph in Figure 3.

The CT win probability is shown in blue, and the T win probability is shown in orange. Although we only need to show one team's win probability, as a team's win probability is just one minus the other team's, we show both, which is in line with most win probability visualization found in other sports. As kills and damage events greatly affect win probability, we indicate kills above the graph, using blue circles for a CT kill and red circles for a T kill.

Because the bombsites are such focal points of CSGO, we also provide two graphs that show the team's minimum distance to both bombsites. CT distance is shown in blue, and T distance is shown in red. We define distance using the same graph based measure in [20], as euclidean distance may be less appropriate for the constrained space nature of CSGO.

Figure 4. The ggViz map view shows player trajectories over the course of a round. T are shown in red and CT in blue. Bombsite are red shaded areas and buy zones are green shaded.

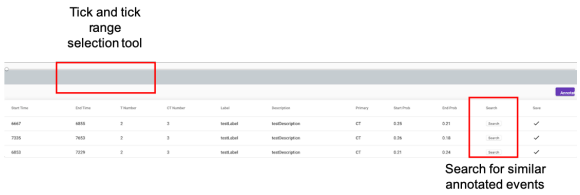

Figure 5. Users can annotate periods of the games to identify tactics and identify similar tactics from other games.

4.2. Player Trajectory View

One of the benefits of ggViz is that it is the first public system, to the best of our knowledge, to provide visual exploration of CSGO trajectory data. This is due to our ability to effectively parse CSGO player positions, which are changed on "Footstep" events in our parser. While the win probability graph, along with bombsite distance graphs, provide some sense of what is happening in a round, the trajectory data allows for a user to understand broader team tactics and movement.

We present ggViz's map view in Figure 4. On each game tick, every player's trajectory up to the current tick is plotted on the map view. Users control the tick through a slider located underneath the map view. While a user cycles through different ticks, a marker appears on the win probability and bombsite distance graphs to show the corresponding tick.

4.3. Event Annotation

While win probability can help a user identify kills and damages, they are often the result of overarching tactics that teams employ on the offensive and defensive phases of the game. These tactics vary by round, team economy levels and score differential, to name a few. Oftentimes, they are hard to identify via clustering techniques, rapidly changing and are best explored visually. To this end, we provide an annotation functionality for users to record tactics. We

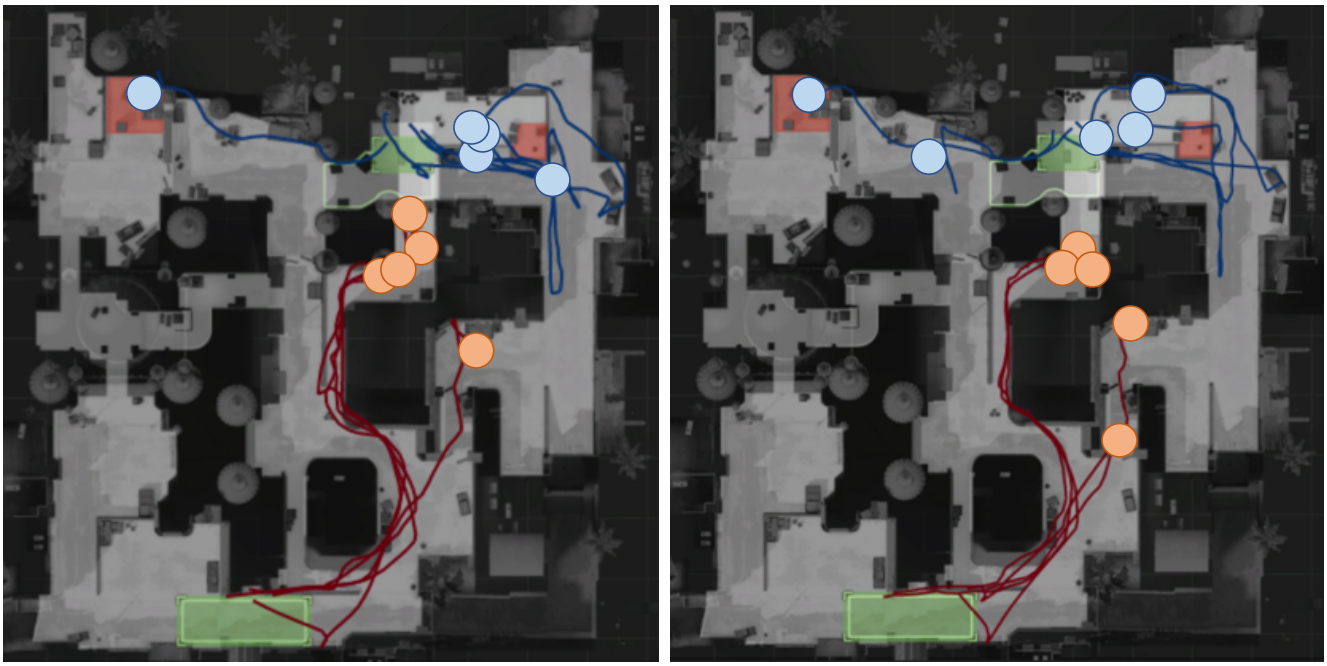

Figure 6. These are two separate "catwalk" tactics on the map Dust 2. Circles indicate player positions. On the left, which is round 2, the T side (orange) increased their win probability from 64% to 99% after using their strategy. On the right, which is round 8, the T side decreased their win probability from 83% to 48%.

show this view in Figure 5. A user can select to search the similar play of events by clicking a "search" button. Then, the ggViz system will search the entire space of user annotations time find similar ones. Currently, the system only takes into account map, annotation label, and players remaining.

4.4. Implementation

ggViz is implemented using a client-server architecture. The client interface is written using the framework AngularJS and libraries such as D3.js [6] and Processing.js to render the graphs and map view. The client makes requests to an API written in Flask to get all relevant game data, such as round information, damages and player trajectories. The Flask API reads data from a SQLite database containing around 100 professional matches from 2019. All of the information for a game is loaded when a game is selected, aside from the trajectories. This is because a game only contains around one to two MB of data, whereas round trajectories can greatly increase the amount of data by 10x. Thus, player trajectories are queried from the API on a round by round basis to make sure that the user experience is not compromised by long loading times. This is important, as our system relies heavily on user interaction.

5. Evaluation

In this section, we present an evaluation of ggViz through the use of case studies. Due to time limitations, we forewent a user study, however we believe this to be a useful future endeavor. We focused on case studies that highlight the intended nature of ggViz, which is to identify interesting tactics and analyze the differences in results.

5.1. Finding Explanations for Outcome Differences

The coordinated views in ggViz allow us to understand *what* is happening in the round, via the map view, and the implications of those actions in the win probability view. As such, a user is easily able to postulate explanations for changes in win probability.

Through the use of the event annotation and search tool, a user is able to quickly view the space of similarly annotated events. This functionality can prove useful for game tactical analysis. For the following case study, we use the a match between Astralis and Natus Vincere, two of the world's top teams, at the BLAST Pro Series Madrid on the Dust2 map.

Consider the two rounds, round 2 and round 8 in Figure 6. In both, we see a standard tactic on the map, which is to attack the part of the map designated "catwalk", which is a central avenue to the A bomb site. An analyst can make a few distinctions between the two scenarios and strategies employed. For starters, it is easy to note that in round 2, the T side dedicated four players to catwalk and a lone player on a part of the map called "Long". However, in round 8, they only dedicated three players to catwalk and two players to long. Another difference is in the positions of the CT

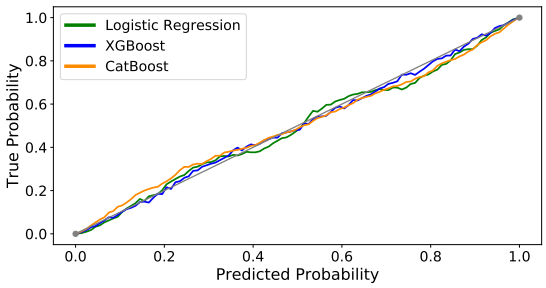

Figure 7. Calibration curves on test data for each model show that XGBoost produced the most well calibrated win probabilities, as it closely follows the center gray line.

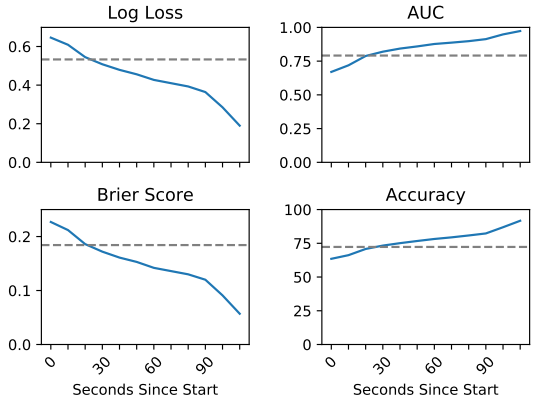

Figure 8. Win probability model performance by round time indicates that our win probability model becomes more accurate, produces lower log loss, and has a higher AUC on predictions deeper in rounds.

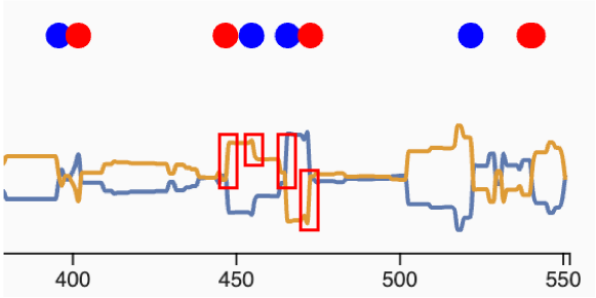

Figure 9. Win probability can change drastically over the course of the match. We see that it is especially driven by kills and damage events. Additionally, there exists variation in the differential those events provide.

defenders (in blue). We see that in round 2, the CT players are farther to the right, and hence, deeper in the bombsite. In round 8, they are further to the left, giving them a more comprehensive view of the catwalk, allow them to see the T players as they round the corner. From these observations, an analyst may deduce that large numerical advantages over your opponent are useful in a catwalk strategy, and to defend against a catwalk attack, the CT side may want to preempt their opponents.

5.2. Assessing Model Calibration

It is important to assess our model performance beyond traditional metrics like AUC or accuracy. This is because we are ultimately concerned with how well the model produces *probabilities* rather than *classifications*. To this end, we show calibration plots of the various models in Figure 7. Calibration plots are useful as many classifiers may perform well on metrics such as accuracy, yet produced probabilities that don't reflect the actual distribution of outcomes. We see that the most well calibrated models closely follow the gray line in the middle of Figure 7. Visually, we see that XGBoost produces the most well calibrated model. We can also assess our model is producing better predictions the deeper it predicts into a round. This is an intuitive check on our model, since we should be better able to predict the end of a round the closer in time we are to it, as there is less opportunity for variability to creep in. We show that our model increases its performance as the time in a round expires in Figure 8. Such a finding lends credibility to the model's efficacy.

5.3. Investigating Win Probability

We can also investigate how our win probability model reacts to game events. In the ggViz system, kills are indicated in the win probability graph via colored circles – blue for a CT kill, red for a T kill. In Figure 9, we show an example of a the variability in win probability over the course of a round. We especially take note of how much win probability appears to be driven by kill events. Additionally, it is clear how there is variation in the win probability change for a kill. In Figure 9, we show changes in win probability due to kills in red boxes. In the first indicated change, the manpower changes from 4 CT vs. 4 T to 3 CT vs. 4 T. Then, the second kill, which evens the game to 3 vs. 3, provides a much lower win probability change. Using the system, we can investigate this change to see that the player low on health in and in a less valuable part of the map.

6. Conclusion

In this paper, we introduce ggViz, a novel visual analytics system to explore CSGO matches, annotate interesting events and analyze tactics. ggViz is built upon a win probability model trained on over 70 million game frames from over 3,000 professional CSGO matches, along with associated event and trajectory data from those matches. We evaluate ggViz using a series of case studies that highlight ggViz's differing views. We also present an evaluation of the underlying win probability model to show that it is well calibrated and produces intuitive results.

We see many future avenues of work for ggViz. On the model side, we believe it will be useful for further development of features to add to a game frame. For example, information about the remaining player's weapons, or the distances between them, may provide better information to predict the outcome of a round. Additionally, we would like to consider more neural-inspired approaches, such as that which won the NFL's most recent Big Data Bowl [2]. There may also be an opportunity for clustering to be used for assessing similar events. For example, we could generate features from the users annotations to cluster similar events. Likewise, we could cluster player trajectories from those identified and labeled events to power a model to cluster tactics. Regarding the system itself, we seek to improve the way that we visualize trajectories on the map. By the middle of most rounds, players have moved around enough that their complex and overlapping trajectories introduce visual clutter on the map view. While we tested some ideas like only showing most recent trajectories or coloring the trajectory with a gradient to show recency, we believe these design decisions are best left to a formal user study. Additionally, we would like to overhaul the design of the system, to make the win probability graphs a focal point of round selection.

References

[1] Most watched games on twitch: Esports content and total. URL: `https://newzoo.com/insights/rankings/top-games-twitch/`.

[2] Nfl big data bowl. URL: `https://operations.nfl.com/the-game/big-data-bowl/`.

[3] Top games awarding prize money. URL: `https://www.esportsearnings.com/games`.

[4] David Bednárek, Martin Kruliš, Jakub Yaghob, and Filip Zavoral. Player performance evaluation in team-based first-person shooter esport. In *International Conference on Data Management Technologies and Applications*, pages 154–175. Springer, 2017.

[5] Alina Bialkowski, Patrick Lucey, Peter Carr, Yisong Yue, Sridha Sridharan, and Iain Matthews. Identifying team style in soccer using formations learned from spatiotemporal tracking data. In *2014 IEEE international conference on data mining workshop*, pages 9–14. IEEE, 2014.

[6] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D^3 data-driven documents. *IEEE transactions on visualization and computer graphics*, 17(12):2301–2309, 2011.

[7] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 785–794, 2016.

[8] Victoria Hodge, Sam Devlin, Nick Sephton, Florian Block, Peter Cowling, and Anders Drachen. Win prediction in multi-player esports: Live professional match prediction. *IEEE Transactions on Games*, 2019.

[9] Margaret C Keiper, R Douglas Manning, Seth Jenny, Tracy Olrich, and Chris Croft. No reason to lol at lol: the addition of esports to intercollegiate athletic departments. *Journal for the Study of Sports and Athletes in Education*, 11(2):143–160, 2017.

[10] Quan Li, Ziming Wu, Peng Xu, Huamin Qu, and Xiaojuan Ma. A multi-phased co-design of an interactive analytics system for moba game occurrences. In *Proceedings of the 2018 Designing Interactive Systems Conference*, pages 1321–1332, 2018.

[11] Quan Li, Peng Xu, Yeuk Yin Chan, Yun Wang, Zhipeng Wang, Huamin Qu, and Xiaojuan Ma. A visual analytics approach for understanding reasons behind snowballing and comeback in moba games. *IEEE transactions on visualization and computer graphics*, 23(1):211–220, 2016.

[12] Ilya Makarov, Dmitry Savostyanov, Boris Litvyakov, and Dmitry I Ignatov. Predicting winning team and probabilistic ratings in "dota 2" and "counter-strike: Global offensive" video games. In *International Conference on Analysis of Images, Social Networks and Texts*, pages 183–196. Springer, 2017.

[13] Alexandru Niculescu-Mizil and Rich Caruana. Obtaining calibrated probabilities from boosting. In *UAI*, page 413, 2005.

[14] Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 625–632, 2005.

[15] Charles Perin, Romain Vuillemot, Charles D Stolper, John T Stasko, Jo Wood, and Sheelagh Carpendale. State of the art of sports data visualization. In *Computer Graphics Forum*, volume 37, pages 663–686. Wiley Online Library, 2018.

[16] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems*, pages 6638–6648, 2018.

[17] Manuel Stein, Halldór Janetzko, Andreas Lamprecht, Daniel Seebacher, Tobias Schreck, Daniel Keim, and Michael Grossniklaus. From game events to team tactics: Visual analysis of dangerous situations in multi-match data. In *2016 1st International Conference on Technology and Innovation in Sports, Health and Wellbeing (TISHW)*, pages 1–9. IEEE, 2016.

[18] Vladimir Vovk. The fundamental nature of the log loss function. In *Fields of Logic and Computation II*, pages 307–318. Springer, 2015.

[19] Jiachen Wang, Kejian Zhao, Dazhen Deng, Anqi Cao, Xiao Xie, Zheng Zhou, Hui Zhang, and Yingcai Wu. Tacsimur: Tactic-based simulative visual analytics of table tennis. *IEEE transactions on visualization and computer graphics*, 26(1):407–417, 2019.

[20] Peter Xenopoulos, Harish Doraiswamy, and Claudio Silva. Valuing player actions in counter-strike: Global offensive. 2020.

[21] Yifan Yang, Tian Qin, and Yu-Heng Lei. Real-time esports match result prediction. *arXiv preprint arXiv:1701.03162*, 2016.