HLTV-50k and Predicting CS:GO Success by Positional Information

Thomas Molinari
University of Virginia
tgm4br@virginia.edu

Abstract

This paper explores data collection and computer vision with Counter-Strike: Global Offensive as the domain. A data collection system, called CSGOCameraMan, is provided to capture data from running replays of CS:GO matches. It introduces a dataset, called HLTV-50k, which consists of over 50,000 images of live gameplay between top CS:GO teams played on de_cache. Finally a model is trained on this collected data and achieves an overall raw accuracy of 60.07%, generating some interesting results in regards to assessing sequenced images and classifying difficult rounds correctly.

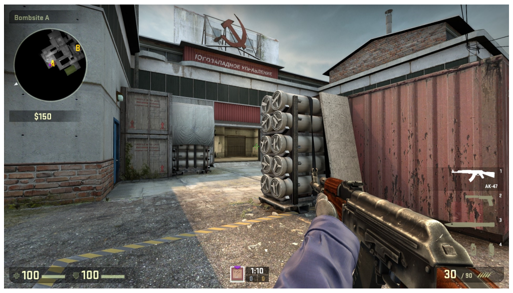

Figure 1. The typical perspective of playing CounterStrike: Global Offensive. Note the size and orientation of the mini-map in the upper left.

1. Introduction

Counter-Strike: Global Offensive (or simply CS) is a first person shooter that is played at a high competitive level on a daily basis. In 2016 alone over 300 online and offline events have awarded nearly$15M[2] in prizes to a community of professional, salaried players. It is encountering explosive growth, with TBS having recently added an entire league (known as ELEAGUE) to the mix to broadcast competitive CS directly into the American home. The path to stardom is notoriously difficult and a chief difficulty is effectively conveying information from seasoned veterans to neophytes. Other than possessing good reflexes, a strong ability to process spatial information quickly and efficiently is essential to become a quality player. This is inherently difficult to teach concept, and often concludes with the advice to play as much as humanly possible to fully understand the implications of play. Professional teams, too, are striving to get ahead against the competition as there is currently the highest level of parity across top teams that CS has ever experienced. Being able to effectively review their play and the play of other teams quickly is an incredibly hard task to accomplish given the volume of games played is far higher than what is humanly possible to observe.

With those considerations in mind, computer vision and visual recognition offer a potential solution to both problems. Instead of being forced to describe a situation in game using only words and hypotheticals, a good CS dataset could offer concrete examples of gameplay scenarios playing out and a computer vision model can help discriminate between more valuable teaching instances. For professional teams, this model could offer a tremendous edge in rapidly iterating strategies and counter plays on maps that the team is traditionally weak on. Both could potentially be invaluable for the health of the competitive scene and for the quality of preparation teams can afford before playing for a million dollar prize pot. However, there has yet to be a dataset that can accommodate a visual system learning CS, let alone any models or paradigms developed to train models on CS spatial knowledge.

This paper sets out to rectify these issues. The final results of this work are found in the HLTV-50k dataset, a collection system called CSGOCameraMan, and a convolutional neural network. This dataset consists of 51,941 images from high level, competitive CS played on one map, `de_cache`. These images come from around 41 high level matches played from March 2016 through November 2016, composed of 1,072 rounds played among the teams. CSGOCameraMan offers a potential framework for extracting game state information from CS replays in an organized manner. The visual recognition model was developed as an attempt to classify the outcome of a round based on a single image from that round. This convolutional network attempts to leverage the high level spatial and color infor-

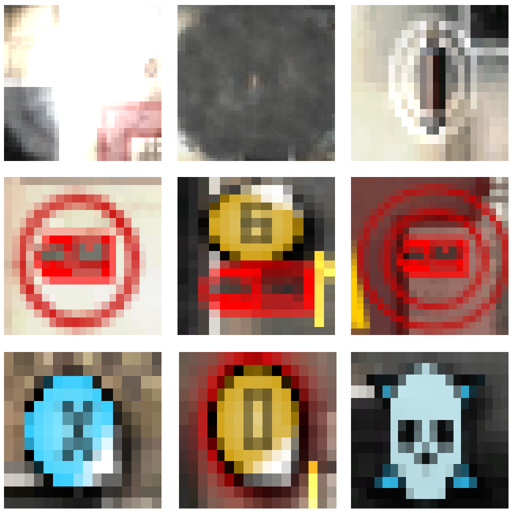
Figure 2. Game state information that is visible on the spectator map. From top left to bottom right: a flashbang exploding, a bloomed smoke grenade, an active decoy grenade, the bomb on the ground, the bomb being carried by a member of the T side, the bomb after being planted, a counter terrorist, an terrorist with less than 25 HP, a recently deceased counter terrorist

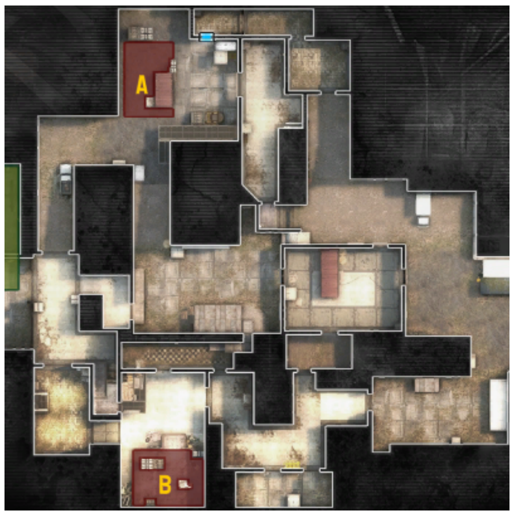

Figure 3. The spectator map overview of `de_cache`. Although obscured in this image, the CTs spawn on the left half of the map and the Ts on the far right. "A" and "B" refer to the areas the bomb can be planted (and detonated) by the Ts. Note that there can be a large discrepancy in the Z direction between points that is not captured when gamestate data is drawn to this map.

mation to attempt to predict which of the two teams will win the round based on only a single image of the round. The architecture and results will be discussed below.

2. History and Gameplay Overview of CSGO

Counter-Strike: Global Offensive was developed by Valve Software as a sequel to Counter-Strike: Source, and was formally released in 2012. A first person shooter based on team-based action gameplay[1], a match in CS consists of two teams of five players playing on a single map until one side wins 16 rounds, switching sides after fifteen rounds have been played. Players can purchase armor, weapons, and equipment at the start of the round based on how much money they possess, which deeply influences competitive play. Rounds can be won depending on which side of the map a given team is playing on. To win a round, the objectives are constrained to a small subset of potential win conditions for each side. For the terrorist(T) side the team can eliminate the opposing counter terrorist (CT) side (by reducing each of their opponents health to zero), or successfully plant and detonate a bomb at one of two possible locations on the map. The counter terrorist side can win by eliminating the terrorist side, defusing the bomb after the terrorists plant it, or surviving until the round timer expires (without the bomb being planted). There are monetary implications for both sides eliminating players, as money is awarded (to purchase equipment) for accomplishing certain tasks such as eliminating an opposing player or winning the round.

CS:GO has a rich strategic and tactical dimensions that competitive players need to consider to perform at the highest possible level. Strategically, the game is centered on how a team can best develop control over space on the map in broader terms. This could mean certain teams are predisposed to not playing expected positions, or utilizing parts of the map in ways that are distinct from their typical use. A very coarse example would be determining which of the two locations the T side of a team is likely to hit can inform how the CT side of their opponent is executed to best counter or undermine this predisposition. Tactically, the game can be observed through several lenses. Predominantly, CS tactics are understood through control of space (called map control), designed plays to attack a part of the map (setups when done by CTs, strats or executes when performed by Ts), money management, and equipment decisions. Examples of game state information can be found in Figure 2.

3. Data Collection, Dataset, and Model

3.1. Data Collection: CSGOCameraMan

For the sake of simplicity, we constrained our process to reviewing only one map from CS:GOs pool of maps: `de_cache`. This map is noteworthy for its balanced gameplay and current position in the professional metagame: both sides generally win rounds on the map at similar rates and it oftentimes is a map that gives underdogs a better chance of defeating more well-established teams. The uncertain outcome of any given round is valuable for any model that is developed as it will become easier to assess efficacy given the near identical win rates for both sides on the map.

The data collection process was helpfully informed by a prior project by Peder Langdal that was interested in creating an application that would display the seconds left until a planted bomb would detonate, information that players normally have to learn based off of the sound of the bomb ticking.[3] The data collector works by first launching the CSGO game client with a valid configuration file and initializing playback on a replay file called a demo. The configuration file causes the game client to post gamestate information to a local address. The collector itself is a listener set to that local address and our CSGOCameraMan that processes and acts based on the data posted to the listener server.

In-game preparation requires the spectator camera to be positioned in a spot where it cannot be blinded, which occurs when a flashbang grenade explodes within the field of vision of the camera. After it is secured, the spectator map is pulled up such that the cameraman can capture screens of the game action. To ensure the cleanliness of our recorded images, the cameraman needs to avoid capturing images during warmup, round freeze-time, during tactical timeouts, and during halftime. During the round, the processor captures frames of action at intervals of every 1.5 seconds. This timeframe allows the round to develop in a nontrivial way without compromising the opportunity to capture meaningful changes in round state, such as player death and bomb plants. Upon round completion, the cameraman cleans up by labeling the sequenced screen captures of the round with the appropriate side winning. This loop iterates over the entirety of the match, where one of the teams ultimately wins 16 rounds or wins overtime.

3.2. Dataset: HLTV-50k

Our candidate data sources are 41 matches played on `de_cache`. These demo files were gathered off of a prominent website dedicated to CS coverage called HLTV.org and the matches were played within the past eight months. This timeframe allows for a number of variables to be held constant. The round timer would be set to 1 minute 55 seconds across all of the matches and the overall metagame of the map would be comparatively similar. The matches were played between two professional teams, or a top amateur team and a professional team, to ensure the highest level of individual and organized play. This can safeguard the dataset from exposure to higher rates of success from high-variance, low probability plays made by either team or their players.

Which results in our HLTV-50k dataset. 3.2 These images are sequenced for each round, where the image is 400x400 and identically centered over the area of play. Rounds can go on for upwards of 80 images, with the same label applied to all images of a round. For our purposes, we ignored the first images few of every round as they were nearly identical.

Partition	**Rounds**	**CT Images**	**T Images**
Train	957	21,820	24,739
Test	109	2,589	2,793
Total	1,066	24,409	27,532

Table 1. The HLTV-50k dataset. The overall round balance between CT and T on the map is 537 to 529 respectively, with a 482 to 475 split in the train and 55 to 54 split in the test.

Of note about the dataset is that there are fewer rounds won by the T side, but a larger amount of images captured and labeled with a T success. This is due to the fact that planting the bomb effectively adds an additional 35 seconds to the overall round timer. Since the bomb can be planted before the round ends if it starts planting at the four second mark, these rounds can run far longer and as bomb detonations are a win condition for the Ts we see a higher rate of image collection for the T side. This is managed by balancing the test set in such a way that we are closer to the overall distribution of round success, which should be a better indicator of the maps overall balance.

A few further notes about the dataset. A pair of matches had their second half data corrupted, and three rounds are missing singular images within their round sequences. Neither of these pose an issue for our particular approach, however, they should be noted if anyone elects to pursue modeling any sequenced relationships using this dataset.

3.3. Model: CSGO-Analyst

The model used to act as our analyst is a convolutional neural network that has a final softmax layer that has two outputs corresponding to the two sides. There are four convolutional layers, followed by batch normalization layers, that proceed from 16 kernels to 128. Kernel sizes initialize at 12x12 with a stride of 0, and conclude with a kernel size of 2x2. Rectified linear units operate as the activation function for our layers, and there is a spatial max pooling layer of 2x2 between the last batch normalization layer and the activation layer. Two fully connected layers connect the model to the aforementioned softmax output. The model was trained using negative log likelihood as the loss function, with initialized class imbalance passed to the model to compensate for the larger number of training instances for the T side. Training parameters for the model included a learning rate of 1e-3, momentum of .5, and weight decay of 1e-3.

The model was trained for three epochs over the training set. The input image is reduced to 200x200, and two preprocessing convolutions are performed on the image. The first takes a patch referencing a T player and applies it across the

Label	Ground Truth T	Ground Truth CT
Predicted T	1208	564
Predicted CT	1585	2025

Table 2. The confusion matrix for the CSGO Analyst model.

entirety of the image, the second performs the same but with a CT as the reference. These two new 3x200x200 images are squashed to grayscale and added as additional channels to the original RGB image. The expectation is that this will accentuate the positions of the respective sides and make it easier to discern which side possesses stronger map control and thus a higher chance to win the round. The model architecture can be viewed in the appendix. 5

4. Results and Analysis

The model was trained on 37 of the 41 overall matches, which consisted of about 45,000 of the total images in the dataset. The matches in the training set had results as lopsided as sixteen to four and as close as a nineteen to seventeen overtime victory. The model was tested against four matches, totalling 109 rounds. 42 of those rounds came from one epic overtime match between two teams called Space Soldiers and Godsent respectively. To clarify our approach, the assumption is that the images are independent of each other and are not considered in sequence. The model only considers whether a given image results in the counter terrorists winning or the terrorists winning. However, given the low latency between consecutive images in a round, there is a high correlation between subsequent images in a round. This informed the decision to have the testing set contain matches that were completely unseen by the model in the training set.

The overall accuracy of the model was 60.07%. This is impressive given the overall win rate of the map is roughly 50% for either side, and due to the heavy skew towards T images. The confusion matrix is provided above to break down the classification rates. 4 As you can see, the model tends to predict that a round will wind up a CT victory, to the point that the model only accurately guesses T victory at a rate hovering near 43% overall. This could be due to an over-aggressive weighting in the class imbalance for the models objective function or potentially due to some unforeseen skewing of round type within the training set. This meaning there could be a higher rate of close rounds that the terrorist side manages to win in spite of large odds (known colloquially as clutch rounds).

The results images show a selection of predictions by the model, categorized by the degree of difficulty or quirk in the rating. Of particular interest is how predictions can flip between sequential images in a given round. Although we abstract sequential image information from the model, it is still of note to see sequential images receive opposite labels. This particular instance 4 has some interesting implications. The first image in the sequence is correctly labeled, and noteworthy in it is the bloomed smokes at the top and bottom of the map. Both of those are generally indicative of CT control over those portions of the map. The position of the terrorists as they begin to entry into the B bombsite also is an indicator that an attack is imminent but not of overall success. The subsequent image is reasonably correlated to T success in a round, as the T side has positioned itself in areas that generally indicate an attack is successfully underway (note the position of the gold 7, autimatic_ps) and the bloomed smoke at the bottom left of the map. Given the current positions of the teams, it is entirely reasonable to predict a T success in this instance.

Another point of fact about the images within the test set is that a good proportion of them are from timings that are generally not indicative of one team succeeding vs the other. These images usually come from the beginning of a round, or near the middle, when little meaningful gamestate information has changed. 5 Predicting who wins these rounds is inherently difficult, as discerning the minutiae of current player positioning in common spaces and the current state of grenades are the only indicators of overall round success. These rounds are inherently interesting examples for these reasons as well, for if a strong classifier can successfully predict round outcome from the limited information these provide, it could prove helpful in indicating strong or novel setups and executes for further research by teams and players.

5. Conclusions and Future Work

This paper demonstrates that images can be effectively captured from a running CSGO client and utilized for computer vision tasks. Future work would hopefully explore how learning one map, like `de_cache` can translate to other maps in the map pool. Additionally, exploring how the sequenced images can be indicative of round (or player level) success would prove interesting and meaningful. Incorporating additional gamestate metadata such as health and equipment would prove valuable to assess granular setups and strats. Creating convolutional layers that also reference gamestate data drawn to the map could also prove interesting for further research.

References

[1] Counter-strike: Global offensive, 2012. http://www.valvesoftware.com/games/csgo.html.

[2] E-sports earnings, 2012. A community run site dedicated to tracking results and earnings of eSports events http://www.esportsearnings.com/games/245-counter-strike-global-offensive/eventsl.

[3] P. Langdal. Gotimer, 2015. https://github.com/LangdalP/GoTimer.

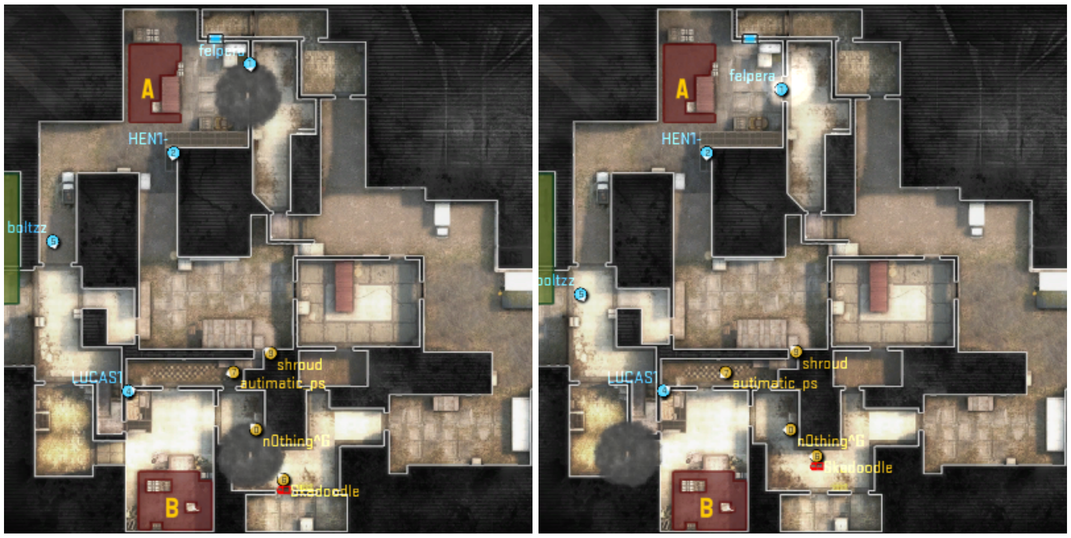

Figure 4. Example of sequence flipping. The image on the right is correctly predicted as a CT round, the image on the left was taken immediately afterwards and is predicted as T.

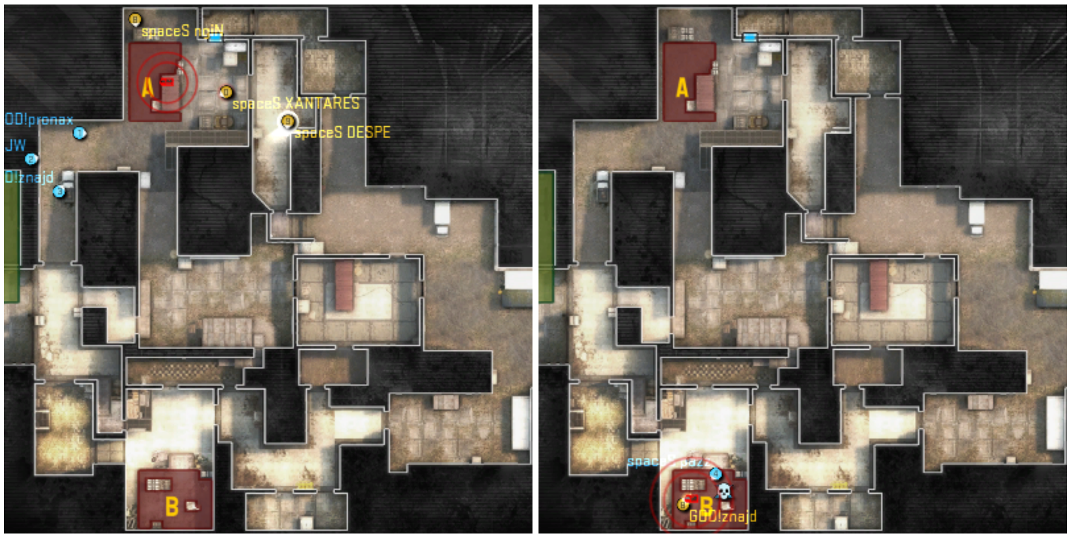

Figure 5. Two difficult but accurately labeled images. The one of the left is appropriately labeled as T, although positioning is not in their favor. The image on the right is correctly predicted CT, even though the terrorist player has recently defeated one of the two remaining CTs.

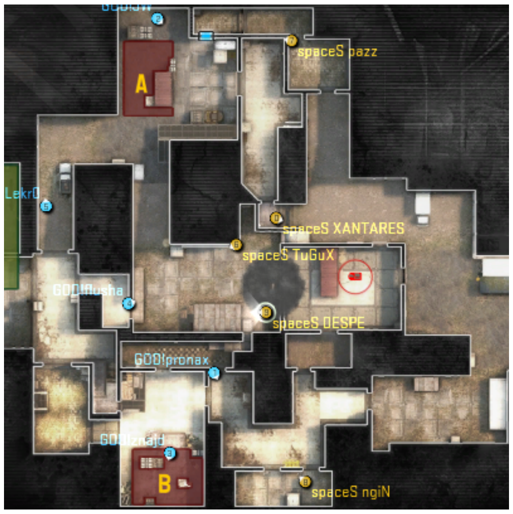

Figure 6. This is an example of a non-obvious round. The difficulty in assessing who wins this round stems from the lack of true positional advantage gained by the T side. The predicted label is CT but the ground truth is T.

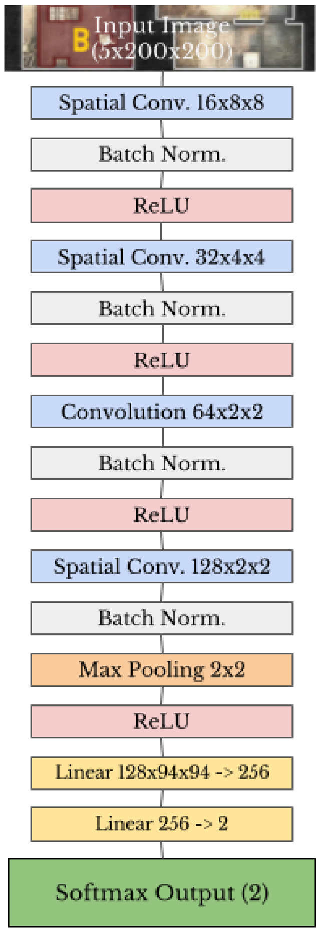

Figure 7. The CSGO Analyst model architecture. The two additional channels are convolutions with patches representing a T and CT player.