Tracking Character Movements Around The Map in League Of Legends

Andrew Dybka - 101041087

COMP 4102

Carleton University

CODE: https://github.com/adybka/COMP4102-LeagueCharacterTracking

1 Introduction

In this project, developing a program that can track character movements on the map in the video game League of Legends was done. This topic was chosen to collect data of characters as they move throughout the map to apply analysis to. In the game there are 10 characters, all being controlled by individual players. Within there the game there is a mini map, which shows the entire map on a smaller scale. Within the mini map, portraits of 10 characters are used to show the locations of each character on the map. One challenge for tracking these characters is as characters move towards one another, their portraits on the mini map will overlap making some portraits partially or completely obstructed from view which can make tracking them difficult. Another challenge, is in the game a character can use an ability to "teleport" else where on the map causing their portrait on the mini map to change places in a drastically and instantly. These challenges outlined where the the main factors in why this project was suitable for this course.

2 Background

As mentioned before this project is focusing on tracking changes of League of Legends characters on the mini map. The reason why this was chosen is because League of Legends is a huge esport. An esport is competitive video games, and currently it is becoming one of the larges forms of entertainment world wide and a billion dollar industry [1]. As it grows, so do the stakes tournament and the investments into players and training staff. One thing that has been has been lacking in esport team's support staff are data analysts.

Over the last few years basketball has exploded with statistics and analysis thanks you computer vision and machine learning where they track player and ball movements [2]. Applying computer vision to track movements of characters in league of legends opens a new area of analysis for a highly analytical competition. However there have been other similar projects but the code is often private and the data available from it is also private or expensive to gain access to [3]. This project would give me access to all the data I would need to apply analysis to League of Legends esports and allow me to adjust the data collected

and modify it was needed.

OpenCV was the main python library used for this project for the computer vision aspect [4]. Numpy was another package that was also used for array and matrix like manipulation [5].

3 Approach

The first step in this project was to detect a character portrait on the mini map. Template matching was used for this as the character portraits would be known prior to tracking and the size of these portraits are always consistent. OpenCV's template matching function was used. There are multiple methods to choose from that performs the template matching. Three images were tested where the portrait being matched were in various places around the mini map. All three images were ran through the template matching function with all 6 possible methods. The methods TM_SQDIFF_NORMED, TM_CCOEFF and TM_CCOEFF_NORMED performed the best having all three matching each image correctly. From here TM_CCOEFF was used because it had the most consistent match value and the best value relative to the others but the other were also kept note of for alternate solutions.

Figure 1: Template matching tests with method TM_SQDIFF that failed

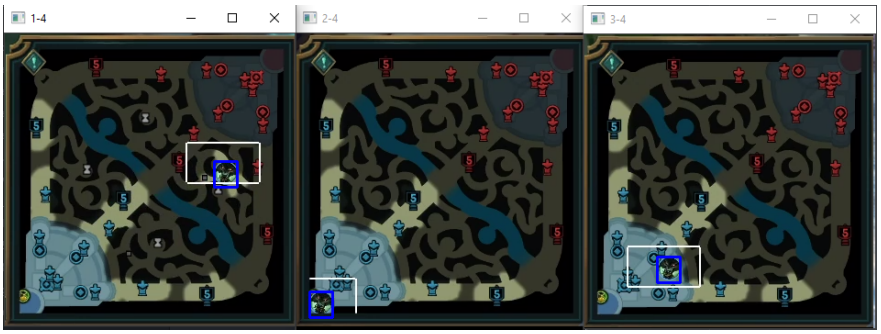

Figure 2: Template matching tests with method TM_CCOEFF that succeeded

The next step was to track the images moving in a frame. To make the detection easier, the frame containing the whole screen is cropped to just the mini map to make template matching more effective.

Figure 3: Full screen before being cropped

The first solution applied was to simply apply template matching to each frame in the video. At this point as well, 10 characters are now visible on the map and are moving so the challenges mentioned in the introduction are becoming relevant. However, the first test was done on the character that happened to have priority over all other characters so it's portrait is always visible and in this case of testing it never "teleports". The solution of applying template matching to each frame worked fairly well but there were frames where the template matching would get a match in the wrong location.

To combat this issues, trackers were used. According to a blog post on pyimagesearch the CSRT tracker is typically the best [6]. From my tests the CSRT was the best tracker out of trackers tested: GOTURN, medianflow, KCF, MIL and CSRT. The tracker is initialized by template matching finding the portrait to track and drawing an box around it to be tracked by the tracker. This worked very well with the portrait mentioned that is always visible and never drastically moves.

The next step was to test on characters that a portrait that "teleports". For this case none of the trackers provided a sufficient failure detection when the portrait teleported. To solve this issue the code reran a template match until a threshold is hit to re-initialize the tracker on the first frame or if the tracker returns a failure. Since the tracker did not report failure well enough, template matching was used to re-initialize the tracker again every 200 frames. This was quite good but there were still some issues with the re-initialization. The re-initialization was tested again with the different methods but the TM_CCOEFF method still performed the best. The re-initiation sometimes passed but on the wrong location and would have to wait 200 frames until the next re-initiation meaning the tracker was tracking the wrong thing. If the tracker did fail or the periodic re-initialization failed it would attempt to re-initialize on every frame until a success.

The last thing that was attempted to optimize the program was adding a mask to the template matching. Since the template is a square but portraits are round, a mask was used to ignore the corners of the template. Applying the mask made the template matching more sensitive and would fail in cases when it would pass and did not provide any clear benefits.

Finally for characters that moved out of view, the tracker again would not register it lost sight of the tracked object. No solution was found that could have been implemented in the limited time apart from the periodic re-initialization. One solution that may be implemented is the introduction of deep learning to identify the portraits and track them more intelligently with better results from drastic movements or when portraits become obstructed. For example, if a portrait become obscured it would then track the portrait the covered it until the tracked portrait became visible again. This solution was implemented in another instance of a similar project [3].

4 Results

The result of this project was a mix of success and failures. The successes were the tracking for easily trackable characters were done successfully. Characters that were slightly obscured were still tracked properly. Characters that would drastically change positions instantly were able to be re-found and characters that were obscured and then visible again were also able to be re-found. The failures were the re-initialization would sometime re-initialize the wrong location when the tracker lost the portrait or on the periodic check. The last failure was that a solution to track obscured images could not be found.

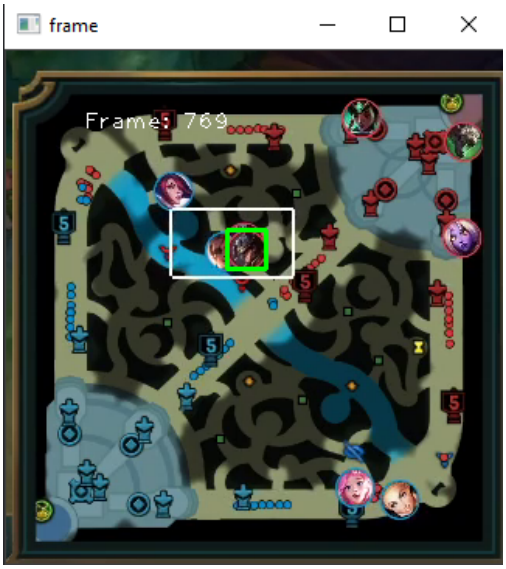

Figure 4: Tracker success on the easiest trackable character

Figure 5: Tracker successfully re-found a portrait that was temporarily obscured

Figure 6: Tracker successfully re-found a portrait that changed positions drastically and instantly

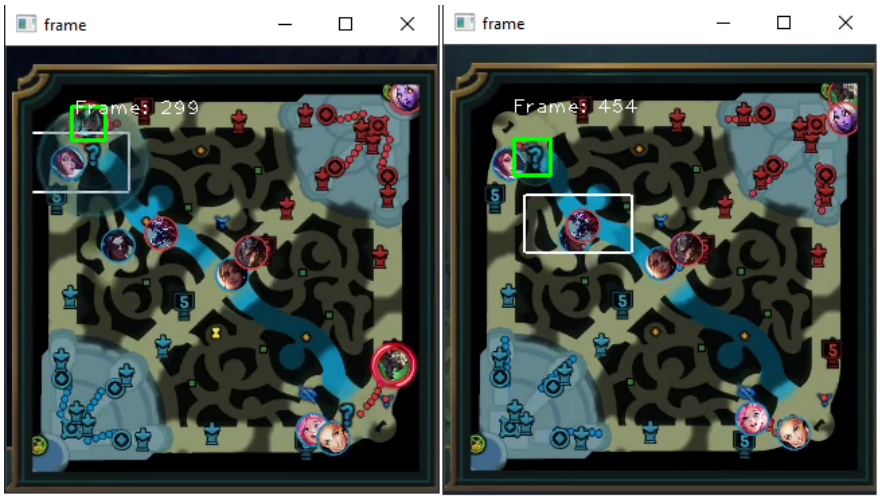

Figure 7: A false positive on a re-initialization of the tracker

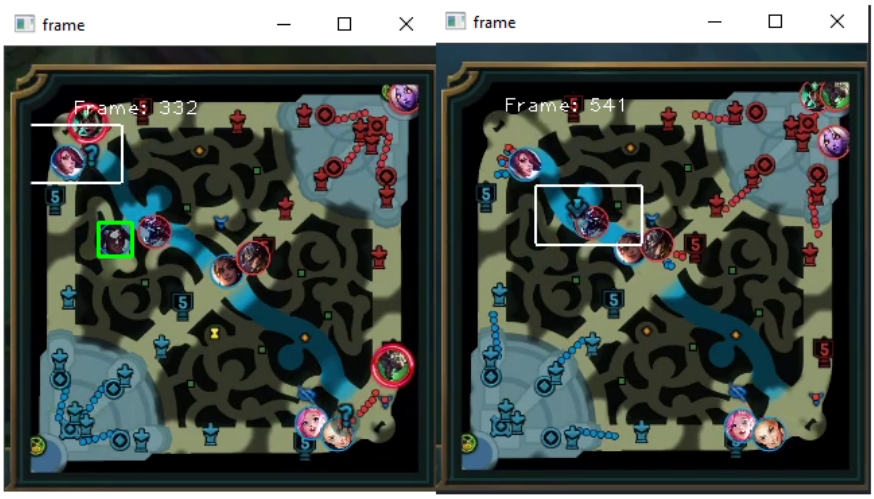

Figure 8: Tracker losing portrait behind another and could not be tracked

References

[1] J. T. Abid Ahmed, "Esports: The next billion dollar industry." [Online]. Available: https://www.millerthomson.com/wp-content/uploads/2019/09/Esports-The-Next-Billion-Dollar-Industry.pdf

[2] "Stats perform." [Online]. Available: https://www.statsperform.com/team-performance/basketball/optical-tracking/

[3] liam schoneveld, "Getting champion coordinates from the lol minimap using deep learning." [Online]. Available: https://nlml.github.io/neural-networks/getting-champion-coordinates-from-the-lol-minimap/

[4] "Opencv." [Online]. Available: https://opencv.org/

[5] "Numpy." [Online]. Available: https://numpy.org/

[6] A. Rosebrock, "Opencv object tracking."