Video-Based Car Localisation in a Known Environment

Mark Newell
dept. of Computer Science
University of Exeter
Exeter, UK
mn367@exeter.ac.uk

***Abstract*—Formula One (F1) teams need a method to accurately track an F1 car around a circuit based on onboard footage alone. This would be the first step into creating a solution to optimize the information being passed to the drivers between qualifying sessions. This paper applies two classical approaches to localization to test how well they adapt to the challenge of having no Inertial Measurement Unit data. The results show that the two algorithms are inaccurate on the straight portions of track but can be very accurate in the corners. Whilst these results suggest classical approaches wouldn't currently work for commercial use, they do show potential areas for further research.**

***Index Terms*—localization, feature matching, particle filters, convolutional neural networks**

I. Introduction

In F1 the smallest improvement for both car and driver can make a huge difference. In the last qualifying session of 2020, there was a 0.025 second difference between first and second place [1]. Hence, each millisecond counts. Ignoring car performance, the main reason for losing time and thus places is driver error. If this was to be minimised, a team would have a greater chance of performing well over a race weekend.

If a driver qualifies in a higher position, this maximises the opportunity to gain points for both the driver and the team in Sunday's race. At the end of the season these points have a significant bearing on both a driver's career and the financial success of a team.

A race weekend starts in earnest with qualifying; this sets the starting grid for Sunday's race. Qualifying is made up of three sessions, the first lasting 18-minutes, the second 15-minutes and the last 12-minutes, where the drivers attempt to perfect their lap times. Between these qualifying sessions, drivers have a maximum of four minutes to analyse the data prepared for them by their teams. This is too short a time to digest all the data collected and hence the information given to the drivers needs to be condensed to only include the key information.

One way that is currently used to help a driver optimize their lap is to compare their best lap to the current best of their rivals. This comparison is often made via a video reel showing the sections of the lap where the driver loses the most time. Which sections are shown is decided by comparing the sector times made freely available by the F1 governing body.

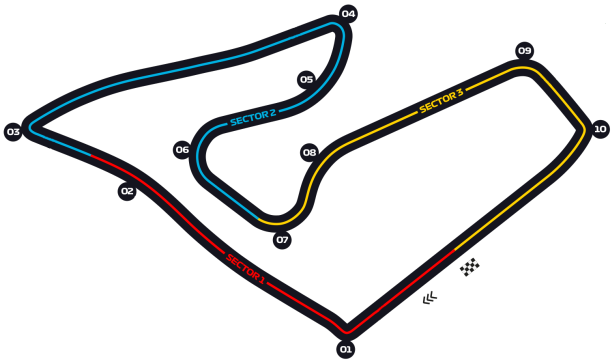


Fig. 1: The Austrian Grand Prix circuit showing the three sectors and nine corners. Adjusted from [2].

Figure 1 shows Austria's F1 track in Styria; this example shows the elements that constitute a lap. It consists of three sectors and nine corners. As there are always more corners than sectors on a track a more detailed comparison could be achieved if corner times were compared rather than sector times.

Apart from sector times, the only other data made publicly available is camera footage taken from onboard the cars. This footage could be used to estimate the location of the car, allowing for a more in-depth lap comparison. If a car could be localized accurately then timing it between points to find out where it lost time

would be trivial.

Some F1 teams have attempted this localization using the same onboard footage. However, these solutions, and their underlying algorithms have not been published so it is unknown how successful they have been.

Localization using camera footage has been applied to many different scenarios through its development and hence there are many ways to perform it. However, in most cases Inertial Measurement Unit (IMU) data is available for fine tuning the localization. Traditionally techniques such as Monte Carlo Localization (MCL) use this IMU data to be able to accurately track the camera and produce precise results. As stated previously, the only data available to teams from other cars is the onboard camera footage. Whilst each team holds GPS and IMU data on their own car it is not accessible to other teams. Therefore, as this paper is comparing cars from various teams, IMU data cannot be used. This brings a new level of complexity to the normal localization problem.

This paper focuses on two techniques: image retrieval combined with Monte Carlo Localization and image retrieval combined with Structure from Motion (SfM).

Image retrieval is a very simple classification technique which attempts to find the closest match to a query image from the train dataset. Once found it can assign the location of this train image to the query image. Image retrieval is a rough localization technique and hence it is not accurate enough on its own for this application, as timing measurements need to be accurate to a thousandth of a second. Therefore, this rough localization needs to be combined with a fine localization technique such as MCL or SfM.

These fine localization techniques are explained in detail in Section IV but a brief overview of how they will be used is given below:

For the MCL system the image retrieval technique is used as a way of weighting each particle with the likelihood of it being the location of the car. At the start of a video the particles are initialized randomly over the search space and iteratively converge to the most probable location of the car. Once initially localized the algorithm iterates over the frames in the video, updating the possible locations to keep track of all possibilities of where the car could be.

Unlike MCL, the SfM method does not track the car and keep context of the wider environment. It uses the image retrieval system to pick the k closest matches in the train dataset for each frame. These k images are then used to triangulate a 3D point cloud of the environment using an SfM technique. Finally, the model is used to localize the query image in relation to these k images. This brings much more accuracy than just assigning the location of the closest image.

This paper focuses on the use and adaptation of these two classical approaches to image localization. In order to facilitate the testing of these two algorithms an image dataset has been created from F1 2018 video game footage and an automatic localization system was prototyped to extract ground truth locations for these images.

The remainder of the paper will be structured as follows: Section II will provide a background on related work. Section III will outline the aims and objectives of the paper. Section IV will give a more in-depth explanation of the two methodologies. The experiments will be described, and analysis of the results will be given in Section V. Section VI will start a discussion on what the results mean in a wider context and areas for further work. Finally, Section VII concludes the paper.

II. Background

There has been no public research into vehicle localisation on a racing track as F1 is a very secretive sport. This section is therefore dedicated to the background knowledge on some of the techniques that were used in this paper, including vision-based localisation, Structure from Motion (SfM) and Monte Carlo Localization (MCL).

A. Features

In 2004, David Lowe presented the Scale-Invariant Feature Transform (SIFT) method to extract features from an image [3]. These features are robust to scale and rotation which makes them excellent for matching in many challenging scenarios. Since being presented by Lowe these features have been used in almost all computer vision fields of study including face authentication [4], object recognition [5] and image matching [6]. The SIFT algorithm has been iteratively improved upon, allowing for dimensionality reduction [7] and the inclusion of spatial information [8].

The introduction of the Convolutional Neural Network (CNN) AlexNet [9] was a huge step forward in computer vision and has been used in many fields including feature descriptors. [10] showed that the features described by the 3rd convolutional layer were invariant to different weather and lighting conditions. This research was extended by [11] who applied dimensionality reduction to improve matching efficiency and performance.

SIFT features in theory perform worse than CNN based descriptors in many scenarios such as small

changes in lighting, zoom and perspective, or the inclusion of blur [12]. However, in practice this often depends on the use case and implementation.

B. Image Retrieval (BOVW)

Vision-based localization is usually cast as an image retrieval problem which finds the closest image in the geo-tagged database to the query image via feature matching. One such example was proposed by Robertson & Cipolla [13]; they used a small dataset of 200 images consisting of building facades taken at different times of the day. Schindler, Brown & Szeliksi came up with a similar approach but used SIFT descriptors to improve matching over a much larger dataset [14]. Due to these ever-increasing dataset sizes, algorithms such as a vocabulary tree and Bag-Of-Visual-Words (BOVW) have been used to improve storage and searching efficiency [14] [15]. More recently CNNs have been used to improve the performance [16] [17] but ultimately, have also increased the complexity.

Image retrieval can only be as accurate as the ground truth coordinates assigned to the images. For many applications, such as in this paper, this is not sufficiently accurate. The remainder of this section looks at two ways to overcome this inaccuracy: SfM and MCL.

C. Structure from Motion

The first way to overcome this inherent inaccuracy of an image retrieval system was proposed by Se, Lowe & Little [18]. They created a 3D map of the environment from triangulating detected SIFT features matched over multiple viewpoints. This technique is known as Structure-from-Motion as the environmental structure can be found from the different camera viewpoints. As the size of the mapped environment increases so does the complexity for searching within it. Sattler, Leibe & Kobbelt used a vocabulary-based search to increase the searching speed for larger 3D maps [19]. This ability to deal with larger areas represented as a map has been steadily improving in the last decade with [20] able to run effectively on more than 74,000 images. SfM has also been applied to localization. The most common overarching method [21] [22] uses an image retrieval system to return N closest matching images in the training dataset. Using these images, the algorithm creates a 3D point cloud model. With this point cloud, the query image can then be accurately localized.

D. Monte Carlo Localization

The second way to improve an image retrieval system that is focused on in this paper is Monte Carlo Localization (MCL). MCL was first proposed in 1999 by Dellaert et al [23] as a probabilistic way of providing a comprehensive solution to the robot localization problem. It makes use of the random sampling Monte Carlo methods that were invented decades before by [24] and builds on the previous work of Markov Localization. Since its inception it has been applied to various different scenarios; [25] attempts to use Bluetooth low energy devices and [26] [27] combine it with an image retrieval system. Traditionally, the MCL method needed to add extra particles or use a method such as [28] to deal with the robot 'kidnapping' problem. This is not required in this paper as the algorithms are designed to run as an offline process so the footage will have been checked for any cut-outs. MCL normally uses odometry data from an IMU which is not present in this instance and therefore presents an additional challenge.

III. Aims and Objectives

Fig. 2: An example solution overlaying a ghost comparison car to video footage, from [29].

Figure 2 shows an example of how a final solution might look. It depicts a ghost car added to the video footage of the fastest lap to compare the two best lap times. This was created by the F1 governing body as they can compare the GPS locations of each car. If the solution proposed in this paper localized to a high accuracy, creating such a sequence would be relatively simple.

The problem, as detailed in Section I, that this paper seeks to address can be decomposed into three main stages:

- Car localization and tracking.
- Time comparisons.
- Condensing into a highlight reel.

Hence, the first step to solving this problem was to create an accurate and robust solution to the car localisation problem. Localisation from video footage

alone, without the help of IMU data, is very complex and therefore was the focus of this project.

The research question proposed was: *Is it possible to accurately track an F1 car in a known environment based on onboard footage alone?*

The objectives of this project were to:

- Compare the two ways outlined to localize a car on a track.
- Produce a working algorithm to track a car on a portion of an F1 track.
- Produce a public dataset of geo-tagged images to enable subsequent research.

IV. Methodology

The two methods proposed in this paper combine classical approaches to the problem of localization and present changes to these methods which vastly improve the overall performance. As described previously, these two methods are MCL and SfM both using a BOVW image retrieval system.

A. Dataset

At the time of conception there were no image datasets that covered an F1 track which were available for public use. Therefore, compiling a dataset was essential to evaluate any model created.

The Austrian Grand Prix track was chosen to be used for testing purposes because in the first sector of the track both long straights and sharp corners were able to be captured. This enabled the solutions proposed to be tested on a variety of track features whilst keeping the amount of data needed to be captured to a minimum.

The F1 governing body makes some onboard camera footage available to the public after each session. However, this footage would need to be tagged by hand with a GPS location using google earth. Using this method would have brought with it a large amount of inaccuracy because google earth can only show satellite images to a certain degree of detail and human error would also be introduced. Tagging the required number of images would also have taken a large proportion of the project time.

Therefore, a dataset was created using onboard footage captured from the F1 2018 video game. Using this video game, a large amount of onboard footage was able to be captured in a small amount of time. The game supports four different times of day: morning, midday, afternoon and evening. Each of these varies the lighting conditions for those laps. Alongside this, the weather conditions can be chosen to be either wet or dry; this further altered the lighting conditions, introduced rain and added blur to the camera lens. Several runs of the chosen section of track were captured with different combinations of conditions to build more variety into the dataset.

(a)

(b)

Fig. 3: Showing the same stretch of track from (a) The F1 2018 video game and (b) Styria, Austria, 2020, from [30].

Once all this footage had been captured it had to be localized with ground truth values. This was done by including the minimap supplied by the game in the images. Looking at the minimap, the location of the small locator arrow, seen in Figure 4, was able to be extracted. This was performed by searching for a colour within the minimap whilst allowing for some variance to enable as many frames to be located as possible.

Whilst this process did produce a location for almost all the images, some were not correct as other areas on the minimap shared too similar a colour to the locator arrow. These incorrect locations had to be removed by hand to uphold the accuracy of the dataset. This process was much quicker than tagging images by hand. However, there was some inbuilt inaccuracy to this process. The locator arrow was 10 pixels wide and covered the entire width of the track no matter where the car was actually located. Alongside this inaccuracy, if the car

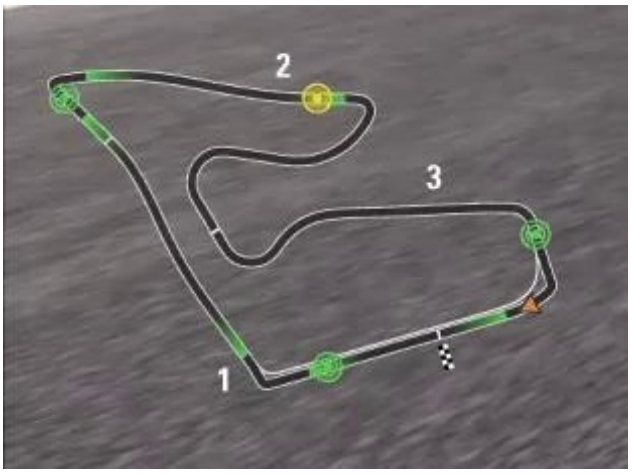


Fig. 4: An example of the minimap extracted from each image. The orange arrow shows the location of the car.

left the track the locator arrow stayed at its last known position on the track.

To conclude, a dataset was created which contained images, in various stages of lighting, of the first sector of the Austrian Grand Prix circuit. It also contains automatically retrieved locations to use as ground truths which are accurate to roughly a 10-pixel radius.

B. Image Retrieval

The Bag of Visual Words (BOVW) image retrieval system used in this paper is a very simple one. A BOVW model has two parts, an offline vocabulary creation and an online classification system.

In the offline phase of the model, SIFT features are extracted from the training dataset and passed through a K-Means clustering algorithm to group them into k different clusters. This k value is known as the vocabulary size. The cluster centres are then saved in a vocabulary file for later use. For all the results shown this k value was set to 500 as this seemed the best trade-off between performance and computational efficiency.

The online phase of the model is where the method used in this paper deviates from traditional approaches. In a traditional BOVW model the online phase starts by detecting the features of all the test and train images. For each image, all the descriptors found are compared to the vocabulary histograms prepared in the offline phase to find the closest match. In this case matching is performed by comparing the cosine distance between each feature histogram and all the vocabulary histograms. The image descriptor histogram is used to keep track of how many matches each vocabulary word receives. Once this matching has been performed for all the features detected, the image descriptor histogram, which counted the matches, is normalized and stored in association with its respective image until they've all been processed. When this is completed and all images have a histogram associated with them, the test and train histograms are compared to find the closest matching train histogram to each test histogram. The metric used to compute this similarity is the Bray-Curtis dissimilarity, shown in (1). In this equation, u_i or v_i is the ith component of the input histogram u or v.

$$BC = \frac{\sum u_i - v_i}{\sum u_i + v_i} \tag{1}$$

Whilst the method proposed within this paper follows a similar vein, it indirectly encodes spatial information into these image description histograms using the method in [8]. This technique works by subdividing each image x times, so the number of image subsections is equal to 4^x. Each of these subsections is then passed through the BOVW model and hence has its own associated BOVW histogram. All these histograms are subsequently concatenated into one large histogram before being fed into the closest neighbour matching algorithm as in the traditional method. As an example, when x is equal to 2 and k is equal to 500 the size of the final histogram is 8,000.

Algorithm 1: Offline BOVW

```
Input: The vocabulary size (k)
Result: BOVW vocabulary
for each train image do
    Extract features;
end
K-Means cluster extracted features;
Save cluster centers as vocabulary;
```

C. Monte Carlo Localization

Monte Carlo Localization (MCL) is a particle filter algorithm designed to localize a robot, in this case a car, in a known environment. The algorithm represents possible states for the car as particles which each have a location and orientation.

The particles start as a random distribution over the known environment and iteratively improve on their positional estimates. In this implementation the algorithm is initialized with 200 particles, this then drops down to 50 after the first iteration gives a better idea as to the initial starting location.

Algorithm 2: Online BOVW with Spatial Information

```
Input: The number of subdivisions (n)
Result: Closest train image to each test image
Load vocabulary;
for each train and test image do
    Split image up into 4^n equal parts;
    for each subsection in the image do
        Extract features;
        Create histogram the same size as the
         vocabulary, full of 0s;
        for each extracted feature do
            Find closest word in vocabulary;
            Add 1 to the histogram bin for that
             word's index;
        end
        Normalize histogram;
    end
    Concatenate histograms of all parts of the
     image;
    Store histogram;
end
for each test image do
    Find closest matching histogram from the set
     of train image histograms;
end
```

There are three phases to each iteration of the MCL algorithm: a movement phase, a sensing phase and a resampling phase.

The movement phase traditionally uses Inertial Measurement Unit (IMU) data which specifies x, y and z movement along with orientation changes. Each particle is then moved by this amount and combined with gaussian noise to account for IMU data inaccuracies. However, as IMU data is not available, a method must be made to approximate this movement.

The method used to approximate the movement was to create a 'Movement Map'. This was a dictionary which mapped a location to an orientation and distance. The orientation and distance were precomputed to estimate how far the car should move in the next 0.2 seconds, if the car was traveling at qualifying speed. 0.2 seconds was the interval chosen as this enables the algorithm to run over each frame of a 5 fps video. For each particle, the closest location in the dictionary's keys is found and the particle is moved the associated distance along the associated bearing. Both the distance and orientation measures introduce gaussian noise of differing sizes to account for the large inaccuracies in this method. The largest gaussian noise is introduced to the distance measure as although the distance in the movement map is how far the car should move, there are often reasons for a car to not move as far as predicted. This could happen on warm up laps or if a crash occurs and the car stops.

The sensing phase of MCL for a computer vision application often uses an image retrieval system and this implementation is no different. Before any iterations the video footage is reduced to run at 5 fps so each iteration has a frame in the footage associated with it. This image is passed through the BOVW model resulting in a histogram which describes the image. Then the Bray–Curtis dissimilarity, shown in (1), between this histogram and those of the train images are normalized and used as probability scores. Each particle averages the weights in a radius around itself to calculate its probability weighting (W_n).

Before each resampling phase the average location is calculated, using (2) and (3), to find the predicted location at that time step. In these equations X_n or Y_n is the x or y component of the nth particle's location.

$$\bar{X} = \sum_n W_n X_n \tag{2}$$

$$\bar{Y} = \sum_n W_n Y_n \tag{3}$$

Finally, the resampling phase is when the current iteration of particles gets discarded, and a new iteration gets created. However, unlike the initial particle creation where each position in the environment has equal chance of being chosen, the new distribution of particles uses the old particle weights to influence where they are initialized. Each new particle is created at the same location as an old particle with the probability of the old particle's weight. This enables strong possible locations to live on whilst improbable particle locations die off.

D. Structure from Motion

Structure from Motion (SfM) is an algorithm which creates a 3D model of the environment by triangulating features found in multiple image viewpoints. It is often used to create 3D models from large datasets of images with no other input. However, as it creates an accurate description of the environment it can also be used to localize one viewpoint in relation to the others making up the model.

The algorithm proposed by [20] and used in this paper works in seven stages:

Algorithm 3: Monte Carlo Localization

```
Input: A 5 fps input video, The weighting radius
       (R)
Result: A set of locations for the car
Initialize particles in random locations;
for each frame in video do
    Update particle's location using the
     Movement Map;
    Get probabilities from feeding frame into
     BOVW model;
    for each particle do
        Calculate which train images are in radius
         R;
        Average these probabilities to assign as
         particle weight;
    end
    Calculate weighted average location;
    Resample particles using previous particle's
     weights;
end
```

- Feature Extraction - SIFT features are extracted from the images fed into the model.
- Feature Matching - SIFT features are matched between the different images.
- Geometric Verification - a way of verifying the feature matches by estimating a transformation from one image to another that is within the tolerance specified. A scene graph is created where images are nodes and verified matches are edges.
- Initialization - an initial pair of images is chosen to begin the model creation; this choice is critical as the algorithm may never recover from poor initialization.
- Registration - the rest of the images in the scene graph are registered to the model by solving the PnP problem [31].
- Triangulation - newly registered images are used to verify matches and increase scene coverage by adding new feature points.
- Bundle Adjustment - after each new image is added to the model all transformations are readjusted to minimise the error.

This algorithm is used in conjunction with the BOVW image retrieval system by using the k closest matches to a query image as the initial images for a model. If the model creation succeeds within the bounds of the error tolerance specified, the query image is added. The location of the query image in relation to the other k images is then extracted and converted back to the original coordinate system. If the algorithm is not able to make an accurate enough model, then the system falls back on assigning the location of the closest matching image in the training dataset to the query image.

The minimum k value to work in this algorithm is 3 as this is the smallest number of viewpoints needed by [20] to create a model. When testing this technique k values range from 3 to 10.

Algorithm 4: Structure from Motion

```
Input: A 5 fps input video, The number of
       images to model (K)
Result: A set of locations for the car
for each frame in video do
    Feed frame into BOVW model;
    Get K closest matching images;
    Create SfM model with these images;
    if model created successfully then
        Add query image to model;
        Extract locations of all images;
        Calculate relative position of query
         image;
        Convert relative position to global
         position;
    end
    else
        Get closest matching image's location;
    end
end
```

V. Results

A. Feature Descriptors

One of the most important decisions to make was the type of feature descriptor to use for the Bag of Visual Words model.

As an F1 car is often traveling in excess of 200 miles per hour it is common for there to be large amounts of motion blur in the footage. Coupled with this, the tracks are outdoors so the weather and lighting conditions can quickly change between each qualifying session.

SIFT is a classical approach to creating a feature descriptor and whilst it is still cutting edge it does have its limitations. One of which is dealing with blurred environments [12].

ConvNet is a more modern approach that utilizes advances made by Convolutional Neural Networks in the last decade to create feature descriptors capable of accurate matching. Each descriptor is produced using

the output of the 3rd convolutional layer of the AlexNet CNN. Where these feature descriptors excel is in changing weather conditions and blurred environments [11]. Therefore, in theory, ConvNet features should be a good fit for this problem.

Both these feature descriptors were used to create five BOVW vocabularies, each with 500 words, using a random train/test split from the image dataset. The train images are processed again at runtime and given a histogram describing them. The test set is also processed, and the resulting histograms are compared to find the closest match. Finally, the location tag of the closest match is assigned to that of the test image. Figure 5 shows a curve depicting the average classification accuracy across these five test/train dataset splits as the accepting radius around the true point increases.

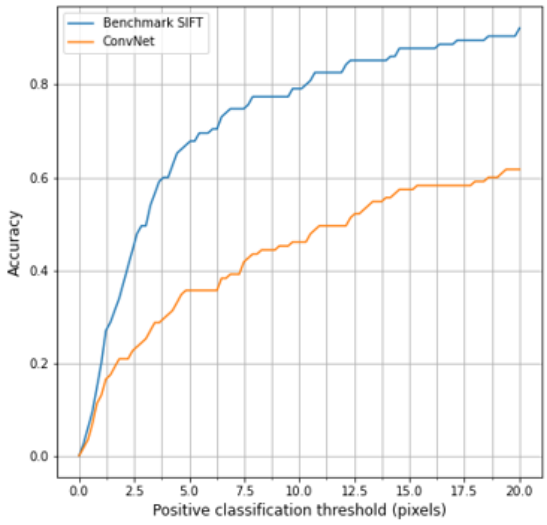


Fig. 5: The two feature descriptor localization accuracy results.

From Figure 5, it is clear that the ConvNet features do not work as well in practice as theory suggests they should. This could be because the ConvNet features are so large (64,296 values) very few of them can be extracted from each image. A maximum limit to the number of ConvNet feature descriptors had to be set at 50, whereas with the SIFT features the maximum could be set to 1000. Clearly with so many more potential features, SIFT was going to have a greater chance of matching similar images.

The sixth layer in the AlexNet network is fully connected which would reduce the dimensionality of the ConvNet descriptors to 4096. It would be interesting to see if a larger number of features were extracted whether this would result in a significant performance increase.

B. Spatial Information

Feature descriptors can often be very similar as only the local area surrounding the feature is described. However, it is rare for the surroundings outside of this local area to also look alike. Hence, encoding this wider information in the form of spatial information into the matching algorithm can be very useful in reducing the number of false positives.

The solutions do not need to be able to detect the same features from multiple different angles as the car rarely deviates far from the track. Therefore, the inclusion of spatial information should work especially well in this instance because many features might look the same from a single angle.

The ways of including spatial information in a matching algorithm can be split into two types, direct and indirect encodings. Three different methods were compared, 2 direct encodings and 1 indirect encoding for the spatial information against a benchmark of classical SIFT feature matching.

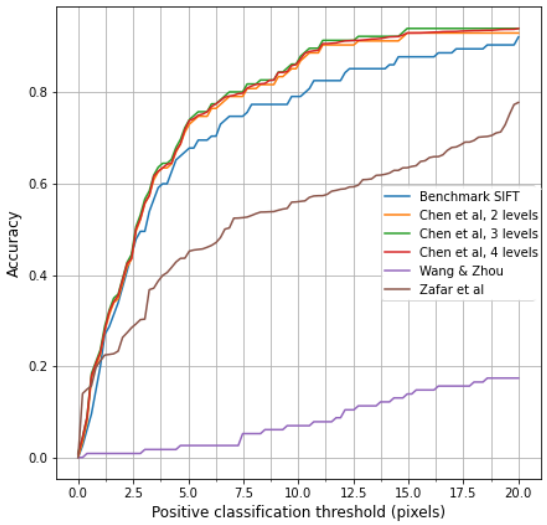


Fig. 6: Spatial algorithm accuracies on the dataset, Chen et al [8], Wang & Zhou [32], Zafar et al [33].

The algorithm presented by Chen et al [8] was the most accurate at classifying the images and was the only spatial information algorithm to outperform the benchmark. The other algorithms didn't perform well in this instance for two reasons: Firstly, they have multiple parameters that have a wide available range which makes optimization computationally costly. As the initial results were not promising enough the limited time spent on this paper had to be prioritized in other areas. Secondly, the papers [32] [33] give little detail for practical implementation.

Once again, all these experiments were performed using a 500 word BOVW model and used random train/test splits to generalize performance across the dataset.

C. Final Results

One of the main aims of this paper was to compare the fine localization techniques SfM and MCL. Both algorithms are described in detail in Section IV.

To give an overview of accuracy in the context of the initial problem and as Monte Carlo Localization requires sequential images, this experiment was conducted using 12 videos divided into 4 different scenarios. These were: crashes, varying speed, wet lap qualifying and dry lap qualifying. Testing all four different scenarios allowed for complete results of the final solution to be gathered.

TABLE I: The mean average location error values (pixels) for the two algorithms and a BOVW benchmark

Video Sequence	SfM	MCL	BOVW
Crash	38.198	9.681	51.245
Varying Speed	52.089	23.53	60.86
Wet Race	15.693	9.104	49.122
Dry Race	41.668	7.446	52.973

The results shown in Table I show that the Monte Carlo Localization performed better in all scenarios that were tested. Not only did MCL have a higher accuracy it also had a smaller runtime per iteration at 330ms compared to 1070ms for the SfM algorithm. On the surface this higher accuracy and low runtime looks very promising. However, as a large amount of random gaussian noise is added into the algorithm at various stages the results are not consistent between runs. This means that whilst a single iteration may only take 330ms the whole algorithm would have to be run numerous times to get the best results.

The localization error for the ground truth gives roughly a +- 10-pixel error to the values shown in Table I. This is a significant amount. It was however, the minimum possible with the resources available at the time of writing. Due to this uncertainty, only general trends will be discussed, and no concrete conclusions can be made.

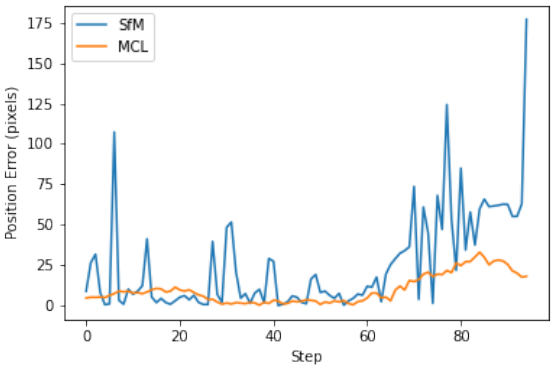


Fig. 7: The position error of the two algorithms at each timestep.

Figure 7 shows the evaluation of localization accuracy in the first crash video and is representative of all the other videos. MCL performed much better on average than the SfM across the 100 frames of this video. It is also interesting to note that both algorithms had significantly worse accuracy on the straight parts of the track compared to the corners. As almost all the time is won or lost in the corners, the ability of the MCL algorithm to localize accurately in the corners shows that it could still be successful in calculating where time is lost.

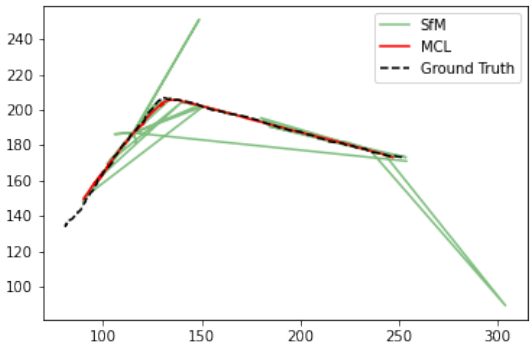


Fig. 8: The estimated movement of the car using the two algorithms.

Figure 8 plots the path taken according to the two methods against the ground truth which is extracted from

the video footage. At certain points the SfM algorithm plots the locations very accurately in relation to the ground truth. However, it is not consistent enough and as it doesn't make use of a tracking algorithm the path appears almost random. On the other hand, the MCL algorithm follows the track well due to the tracking system.

VI. Discussion

At the time of writing there is no similar, publicly available work, that uses onboard cameras without IMU data for localization. Therefore, a direct comparison cannot be drawn to with any other localization algorithm. However, the techniques are compared against a benchmark BOVW image retrieval system to enable basic conclusions to be drawn on the algorithms' performance.

The main finding of the work performed in this paper is that whilst classical localization approaches perform better than a simple image retrieval system, they do not work as well as expected when devoid of accurate IMU data. Whilst the MCL algorithm outperformed SfM it introduced a large amount of random gaussian noise with the use of the Movement Map. This meant it had to be run multiple times to produce satisfactory results. Overall, this made both algorithms run times roughly equal. To combat these issues, a lack of accuracy and variation between runs, there are areas for further work including: alternative localization approaches which rely less heavily on IMU data, or a more accurate way of imitating IMU data to work with existing solutions.

This project was not however without success. The MCL based algorithm was able to localize to a high degree of accuracy around corners, as there were more features to detect and compare. In order to meet the original hypothesis a track could conceivably be split up into its constituent corners. Car tracking would take effect as it approached each corner and stop once the exit has been reached. This would be sufficient to compare lap times as the majority of time is lost in the corners of a track and is where driver improvement would have the greatest effect.

The creation of the geo-tagged image dataset was also successful. However, the location tagging did introduce a large degree of inaccuracy. This could be improved if GPS data for several laps was made available to create the dataset. F1 teams wanting to create a solution to the problem posed in this paper would have access to this GPS data and hence in a commercial environment the accuracy of the final solution would be much greater.

VII. Conclusion

The problem presented in this paper aimed to give F1 teams a means to optimize their qualifying laps. This could be done by showing a driver where improvements could be made on their last lap by comparing it to the current best. Due to the secretive nature of F1 the only data available to a team on other drivers' laps is the onboard footage that the F1 governing body release. The first step to comparing lap times is to accurately localize the car. The approach tested in this paper was to cast the problem as a complex application of image localization. Hence, this paper asked the question: *Is it possible to accurately track an F1 car in a known environment based on onboard footage alone?*

Detailed in this paper were two classical approaches to robot localization. The first being Monte Carlo Localization and the second being a Structure from Motion algorithm, both combined with an image retrieval system.

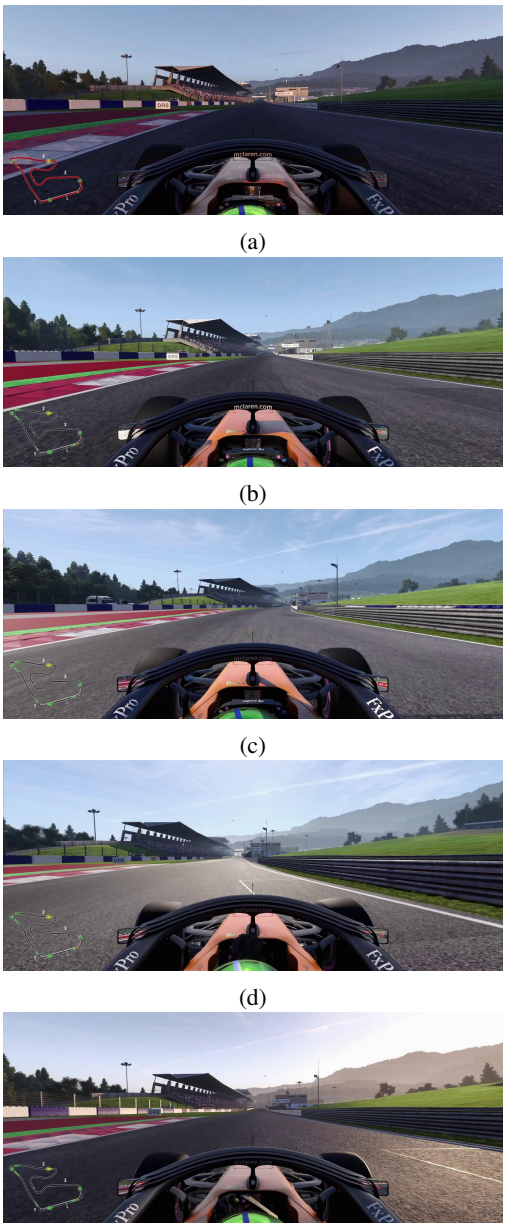
The findings showed that these classical approaches did not work well when devoid of accurate Inertial Measurement Unit data. Whilst the results showed it was possible to track a car in a known environment with onboard footage alone it was not as accurate as it would need to be for commercial use. However, the results showed the MCL algorithm was highly accurate around corners. This could be sufficient to compare lap times as the majority of time is lost in the corners of a track and is where driver improvement would have the greatest effect.

Whilst creating these solutions a geo-tagged image database was created using F1 2018 video game footage. This has been made publicly available to enable further research to be performed without spending time recreating such a dataset.

Declaration of Originality. I am aware of and understand the University of Exeter's policy on plagiarism and I certify that this assignment is my own work, except where indicated by referencing, and that I have followed the good academic practices.

Declaration of Ethical Concerns. This work does not raise any ethical issues. No human or animal subjects are involved neither has personal data of human subjects been processed. Also no security or safety critical activities have been carried out.

APPENDIX



(a) (b) (c) (d) (e)

Fig. 9: Showing the same stretch of track in different lighting conditions: (a) Sunrise, (b) Morning, (c) Midday, (d) Afternoon, (e) Evening.

REFERENCES

[1] (2020) FORMULA 1 ETIHAD AIRWAYS ABU DHABI GRAND PRIX 2020 - QUALIFYING. Formula One. [Online]. Available: https://www.formula1.com/en/results.html/2020/races/1061/abu-dhabi/qualifying.html

[2] (2021) Austrian GP race track layout. Formula One. [Online]. Available: https://www.formula1.com/en/racing/2021/Austria/Circuit.html

[3] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[4] M. Bicego, A. Lagorio, E. Grosso, and M. Tistarelli, "On the Use of SIFT Features for Face Authentication," in *2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)*, 2006, p. 35.

[5] H. Zhou, Y. Yuan, and C. Shi, "Object tracking using SIFT features and mean shift," *Computer Vision and Image Understanding*, vol. 113, no. 3, pp. 345–352, 2009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1077314208001331

[6] J. Joglekar and S. S. Gedam, "Image Matching With Sift Features – A Probabilistic Approach," *Iaprs*, vol. XXXVIII, no. September, pp. 7–12, 2010.

[7] Y. Ke and R. Sukthankar, "PCA-SIFT: a more distinctive representation for local image descriptors," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 2, 2004, pp. II–II.

[8] R. Chen, M. Hawes, L. Mihaylova, J. Xiao, and W. Liu, "Vehicle logo recognition by spatial-SIFT combined with logistic regression," in *2016 19th International Conference on Information Fusion (FUSION)*, 2016, pp. 1228–1235.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'12. Red Hook, NY, USA: Curran Associates Inc., 2012, pp. 1097–1105.

[10] N. Sünderhauf, S. Shirazi, F. Dayoub, B. Upcroft, and M. Milford, "On the performance of ConvNet features for place recognition," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 4297–4304.

[11] N. Sünderhauf, S. Shirazi, A. Jacobson, F. Dayoub, E. Pepperell, B. Upcroft, and M. Milford, "Place recognition with convnet landmarks: Viewpoint-robust, condition-robust, training-free," *Robotics: Science and Systems*, vol. 11, 2015.

[12] P. Fischer, A. Dosovitskiy, and T. Brox, "Descriptor Matching with Convolutional Neural Networks: a Comparison to SIFT," 2014.

[13] D. Robertson and R. Cipolla, "An Image-Based System for Urban Navigation," in *BMVC*, 2004.

[14] G. Schindler, M. Brown, and R. Szeliski, "City-scale location recognition," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007.

[15] J. Yang, Y. G. Jiang, A. G. Hauptmann, and C. W. Ngo, "Evaluating bag-of-visual-words representations in scene classification," *Proceedings of the ACM International Multimedia Conference and Exhibition*, pp. 197–206, 2007.

[16] F. Radenović, G. Tolias, and O. Chum, "Fine-Tuning CNN Image Retrieval with No Human Annotation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 7, pp. 1655–1668, 2019.

[17] M. Tzelepi and A. Tefas, "Deep convolutional learning for Content Based Image Retrieval," *Neurocomputing*, vol. 275, pp. 2467–2478, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231217317587

[18] S. Se, D. Lowe, and J. Little, "Global localization using distinctive visual features," *IEEE International Conference on Intelligent Robots and Systems*, vol. 1, no. November, pp. 226–231, 2002.

[19] T. Sattler, B. Leibe, and L. Kobbelt, "Fast image-based localization using direct 2D-to-3D matching," in *2011 International Conference on Computer Vision*, nov 2011, pp. 667–674.

[20] J. L. Schonberger and J. M. Frahm, "Structure-from-Motion Revisited," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 4104–4113, 2016.

[21] M. Salarian, N. Iliev, A. E. Çetin, and R. Ansari, "Improved Image-Based Localization Using SFM and Modified Coordinate System Transfer," *IEEE Transactions on Multimedia*, vol. 20, no. 12, pp. 3298–3310, 2018.

[22] M. Humenberger, Y. Cabon, N. Guerin, J. Morat, J. Revaud, P. Rerole, N. Pion, C. de Souza, V. Leroy, and G. Csurka, "Robust Image Retrieval-based Visual Localization using Kapture," 2020. [Online]. Available: http://arxiv.org/abs/2007.13867

[23] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte Carlo localization for mobile robots," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, vol. 2, may 1999, pp. 1322–1328 vol.2.

[24] J. E. Handschin, "Monte Carlo techniques for prediction and filtering of non-linear stochastic processes," *Automatica*, vol. 6, no. 4, pp. 555–563, 1970. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0005109870900105

[25] X. Hou and T. Arslan, "Monte Carlo localization algorithm for indoor positioning using Bluetooth low energy devices," in *2017 International Conference on Localization and GNSS (ICL-GNSS)*, 2017, pp. 1–6.

[26] J. Wolf, W. Burgard, and H. Burkhardt, "Robust vision-based localization by combining an image-retrieval system with monte carlo localization," *IEEE Transactions on Robotics*, vol. 21, no. 2, pp. 208–216, 2005.

[27] S. Xu, W. Chou, and H. Dong, "A robust indoor localization system integrating visual localization aided by CNN-based image retrieval with Monte Carlo localization," *Sensors (Switzerland)*, vol. 19, no. 2, 2019.

[28] D. Fox, "KLD-sampling: Adaptive particle filters," *Advances in Neural Information Processing Systems*, 2002.

[29] G. F1. Ghost qualifying comparison @max33verstappen vs @landonorris — 2021 austrian grand prix. [Online]. Available: https://mobile.twitter.com/GhostF111/status/1412545064551714822

[30] (2020) Valtteri Bottas' Pirelli pole position lap in Austria. Formula One. [Online]. Available: https://www.formula1.com/en/video/2020/7/ONBOARD__Valtteri_Bottas%27_Pirelli_pole_position_lap_in_Austria.html

[31] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, p. 381–395, Jun. 1981. [Online]. Available: https://doi.org/10.1145/358669.358692

[32] Y. Wang and Z. Zhou, "Spatial descriptor embedding for near-duplicate image retrieval," *International Journal of Embedded Systems*, vol. 10, no. 3, pp. 241–247, 2018.

[33] B. Zafar, R. Ashraf, N. Ali, M. Ahmed, S. Jabbar, and S. A. Chatzichristofis, "Image classification by addition of spatial information based on histograms of orthogonal vectors," *PLOS ONE*, vol. 13, no. 6, pp. 1–26, 06 2018. [Online]. Available: https://doi.org/10.1371/journal.pone.0198175