Universiteit Leiden

Master Computer Science

Video-based tracking of a speed skater

in the corner

Name:	Michael de Rooij
Student ID:	s2713454
Date:	24/06/2024
Specialisation:	Artificial Intelligence
1st supervisor:	Arno Knobbe
2nd supervisor:	Arie-Willem de Leeuw

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

This research examines how a top-down view camera system, installed in the ceiling of an ice skating stadium, can be used to review and analyze the performance of a speed skater through the corner of the track. For this research, videos were recorded using the camera system. Using these videos as training data, a CNN model is trained to automatically locate the ice skate blades in a video frame. To use these locations for calculation of performance measures, the pixel locations are translated to real-life locations using trilateration and a homography. From this, the trajectory and the speed of a speed skater through the camera frame are calculated. We found that the CNN model is able to train and learn to locate the ice skate blades in a given frame with an prediction error around 7.5 pixels. After the application of trilateration and the homography, we were able to calculate and visualize the trajectory and speed of a speed skater during a video. The used camera system combined with additional software shows to be a promising tool to assist speed skaters and their teams analyzing speed skating performance.

Contents

1 Introduction

Speed skating is a highly dynamic sport in which every (milli-)second counts to beat an opponent. This makes analysing the technique of the speed skater and other data resulting from a lap around the ice rink very valuable to keep improving for better lap times. Because of this, there are various studies available already related to sports that involve ice skating, which analyse various performance metrics using different types of cameras [PKK+18][LTC+09]. However, none of these studies record videos using cameras with a top-down view on the ice rink. This raises the question in what ways data from recordings showing a top-down view can be used to analyse the performance of speed skaters.

In order to be able to analyse speed skating performance, a wider project is in progress to have an Apex camera system installed in the ceiling of Thialf, the biggest speed skating stadium in the Netherlands. This system can be used to record videos of speed skaters, enabling speed skaters and their teams to review, analyse and assess the laps done by a speed skater afterwards. This camera system consists of six cameras installed in the ceiling with one camera installed at the start of one of the corners of the track, one installed at the end of the corner and the other four cameras installed evenly spread in between the other two cameras. The goal of this research is to add another building block towards a more standardized software package, which can be used to analyse the performance of a speed skater on the basis of a camera system. Having a standardized system and software package that can be used in different applications allows coaches and athletes to analyse and gain insights into differences between laps and how these differences affect particular performance measures.

Many aspects of the performance of a speed skater can be analysed, but this research focuses on using recordings captured by the Apex camera system to gain insight into the speed and the trajectory of a speed skater in the corner. This results in the following research question:

How can the use of the Apex camera system aid in analyzing a speed skater's performance in terms of speed and trajectory?

Videos have been recorded using the camera system to automatically determine speed skater position in each frame using a CNN (Convolutional neural network) model. The distance travelled between frames is calculated in order to analyse performance measures such as the speed of the speed skater through the corner.

The remainder of this thesis is organised as follows: Section 2 discusses related work, which contains topics that form a basis for this research. Section 3 gives background on some of the techniques used in this study. Section 4 discusses how this research is conducted. Section 5 describes the results and Section 6 concludes.

Finally I want to thank Arno Knobbe for his continuous support and supervision during this project. I have learned a lot during this period and it was really helpful for me that there was always time to ask questions or brainstorm about solutions to obstacles during the project. I also want to thank Arie-Willem de Leeuw for his supervision, especially in the last stage of the project. A special thanks goes to Renz Roos for the nice collaboration we had along the way.

2 Related Work

Section 2.1 discusses general literature on speed skating. Section 2.2 discusses the current state of object detection in images using neural networks.

2.1 Speed skating analysis

Even though there is not an abundance of literature available on the use of data analysis in speed skating, there already is a reasonable amount of literature covering different areas of interest in the sport. One of these areas is that of training schedules, in which the literature contains studies towards training load and training intensity [KOH+17][OHdKF14], but also towards the type of training performed [OHMK20].

Another important part of speed skating is the technique used by a speed skater to maximize the velocity around an ice rink. Already in 1982, a study was conducted to measure the air friction a skater endures during speed skating [vIS82]. In more recent literature, various studies discuss the effects of elements such as the lean angle of the skate [VdKSVDHV16] or the push-off angle [NFHDK13], calculated from multiple angles, on the performance of the skater. Unlike the discussed literature, this study will focus more on analysing the speed in the corner while observing the line taken through the corner.

2.2 Object detection using deep learning

In the past few years, deep learning models have become the state-of-the-art for object detection in video frames and images [RHGS15][MLAD17]. The main reason for this increase of performance is the rise of Convolutional Neural Networks (CNNs) [LHB04][KSH12]. CNNs work well for computer vision problems, because these models are designed to process arrays containing pixel intensities as input. Through the use of convolutional layers and pooling layers in the first stages of the model and the use of fully-connected layers in the later stages, updating the parameters of the CNN model is similar to how a regular deep neural network is updated [LBH15].

There is plenty of literature that discusses the use of CNNs for object detection in various ways [ZZXW19]. One study introduces the use of weakly-supervised learning with CNNs, which results in a heatmap showing where the labeled object is in the image [OBLS15]. This method works well especially in situations where the number of richly annotated images is slim. Another deep learning technique that is being used more frequently due to its remarkable performance is a Region-based Convolutional Neural Network (R-CNN) [GDDM15][RHGS15][MLAD17]. This system proposes many regions of interest per image and then uses a CNN model to compute features for each region, which can be used to classify the regions.

While the literature discussed so far shows promising results for addressing the problem of object detection, there are also formulations of object detection present in the literature that represent the problem as a regression problem. The studies by Szegedy et al. [STE13] and Erhan et al. [ESTA14] introduce methods to apply a Deep Neural Network (DNN)-based regression model to object masks and extract object bounding boxes afterwards. While the performance of these models is solid, the training process requires a significant amount of training data, which is

infeasible for this study.

Object detection usually consists of two parts, being object localization and object classification. Because this study only focuses on localizing one type of object, ice skates, only the object localization step is examined. Bounding-box regression is a deep learning technique which is frequently used to localize objects in images and video frames. The technique is used to regress from an image or video frame to a bounding box surrounding the object, which is usually used by the object classification step to classify the object in the bounding box [LKC19]. In this study, this technique will be applied to video frames showing speed skaters on track, in order to localize the skates of the athletes.

3 Background

In this section, we give some background information on the topics of RANSAC, trilateration and homography, which are used in this study.

3.1 RANSAC

Not all predictions that are produced by the object detection model will be perfect. If a model is to be fitted to the data points, these outliers influence this model negatively. The fitting of this model will be corrected using the iterative method Random Sample Consensus (RANSAC) [FB81], a technique to robustly estimate a model using only the data points that are close to the fitted model. In this algorithm, random samples of the data are taken for multiple iterations. The goal of the algorithm is to find a random sample of data points that contains the most inlier points and the least outlier points, aiming to find a model that fits best to the inlier subset of the full dataset. In this way, the outliers in the dataset are excluded from the fitted model. In each iteration, a model is fitted to the random sample of data points. Using some error threshold, the algorithm determines which data points from the full dataset would be a good fit to this fitted model and therefore could be part of the inlier set. Regarding the error threshold, this can be defined in various ways, also depending on the use case. The original authors on the RANSAC algorithm state that setting the error tolerance one or two standard deviations away from the average error could be a useful estimate for defining the error threshold [FB81]. The Scikit-learn package, which will be used in this study to execute the RANSAC algorithm, defines the error threshold as the median absolute deviation.

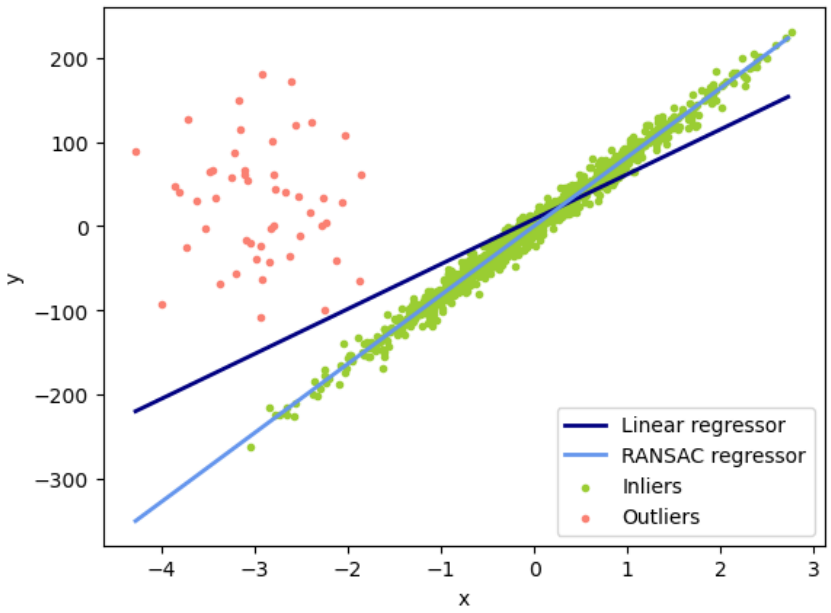


Figure 1: Example image of how RANSAC works. The RANSAC algorithm fits a model to the data, excluding the outliers in the data.

The stopping criterion of the RANSAC algorithm for finding a good fitting model to the inliers of the dataset can be defined in several ways. One can define a threshold for the number of inliers that need to be found for the model to be deemed a good fit to the inlier data of the dataset. Another criterion that can be used is to predefine a maximum number of attempts to

find a good enough model fitting to the data. [FB81] suggests to use the following formula to determine the number of attempts:

$$E(k) = w^{-n} \tag{1}$$

Here, w is the probability that any selected data point is within the defined error threshold and n is the size of a subset containing good (inlying) data points. In addition to this formula, they suggest to exceed $E(k)$ by one or two standard deviations to obtain a good model. When the used stopping criterion is met, the algorithm returns the fitted model that has the best fit to the data. Figure 1 visualizes the result of using the RANSAC algorithm using a linear model. For our study, a regular linear model will not suffice to model the data. Because of this, the linear input data will be transformed to a polynomial feature matrix with the desired degree.

RANSAC has shown good performance in various research areas, such as robotics [MORMMS22], object detection using 3D point clouds [JSK21] and even in combination with deep learning for moving object tracking [BRB19].

3.2 Trilateration

Trilateration is a widely used localization and positioning technique [DSO08][TR05], which uses known distances between points and the known location of some points to determine the position of a point of interest of which the location is unknown. Figure 2 visualizes the method. The application of trilateration results in a local coordinate system based on two reference points, point A and point B located on the x-axis in the figure. Let us define points A and B as the two reference points of which the position is known. Point P is then defined as the point of interest of which we do not know the exact location. In order to determine this location in the coordinate system, first of all the distance between both reference points, $d(A, B)$, is measured as this information is vital for locating other points in the world. In addition to this, we need to measure the distance from both reference points to the point of interest, so $d(A, P)$ and $d(B, P)$. With this information, the x-coordinate of the point of interest (X_P) can be calculated using the following formula:

$$X_P = \frac{d(A,B)^2 + d(A,P)^2 - d(B,P)^2}{2d(A,B)} \tag{2}$$

The y-coordinate of the point of interest follows from the x-coordinate using the following formula:

$$Y_P = \sqrt{d(A,P)^2 - X_P^2} \tag{3}$$

Following the steps described above this will result in an estimation of the location of point P. This process can be repeated to estimate other points of interest within the local coordinate system.

Trilateration has many applications and is therefore discussed in plenty of literature. For instance, one study discusses the use of trilateration to locate a robot [TR05], while another study uses a trilateration-based localization algorithm for the positioning of a wireless sensor network [OAEO13]. While these studies use trilateration in a situation where only physical objects are a factor, we will eventually locate objects in a camera frame after applying trilateration. Various

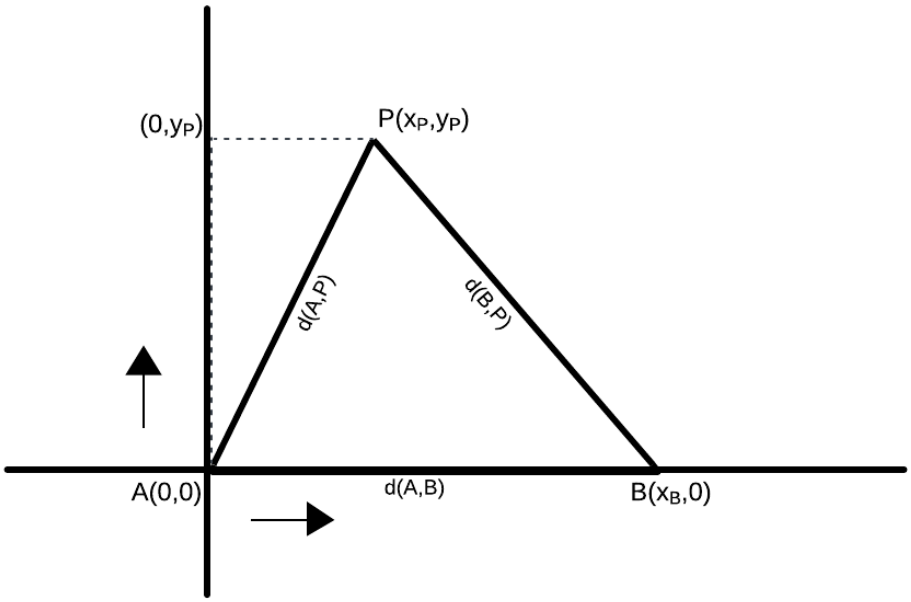


Figure 2: Trilateration example. Points A and B are known, point P is unknown.

studies are present in the literature discussing the use of trilateration with camera settings directly. One study discusses the use of trilateration for object detection and localization in known environments [Pac21], while another study uses the same technology to apply self-localization for the camera's position [ZLLZ19]. Two other studies combine a camera with a LiDAR system [MW22] and an infrared detector set [MGGGP+19] to optimize object localization.

3.3 Homography

In the previous section we discussed how to locate several markers in a local coordinate system. We want to combine this data on their world location with data on their location in the frame of a camera, which is in image coordinates. By doing this, we have data on locations in both coordinate systems, which allows determining the location of a speed skater going through the camera frame in world coordinates. We will use a homography to connect these coordinate systems, which is a transformation technique between two projective planes. Initially, all available point pairs (image coordinate and world coordinate of each marker) will be used to estimate a perspective transform (homography matrix) on the data. This can be noted in the following way:

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \sim H \begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix} \tag{4}$$

Here, x_i and y_i are the coordinates of marker i in one plane and x_i' and y_i' are the coordinates of marker i in the other plane. H is the homography matrix. If there are point pairs that do not fit well to this perspective transform estimate, several techniques such as RANSAC can be applied here to improve the estimate. The resulting homography matrix allows translating any point in image coordinates to world coordinates or vice versa, depending on the use case.

4 Methodology

4.1 Data preprocessing

The dataset used to train the neural network consists of different parts. All the data is collected using the Apex camera system installed in Thialf, located in Heerenveen. With this camera system, many videos have been recorded of professional speed skaters using a high frame rate, of which the majority is recorded in the first part of a corner on the track. From each frame in each video, a bounding box is extracted with a downscaled size of 75x100, which contains the speed skater. The script that we use to extract the bounding boxes is created by Renz Roos, who is a colleague on the same bigger project on the use of the Apex camera system for speed skating analysis. Figure 3 shows multiple example images that are part of the dataset.

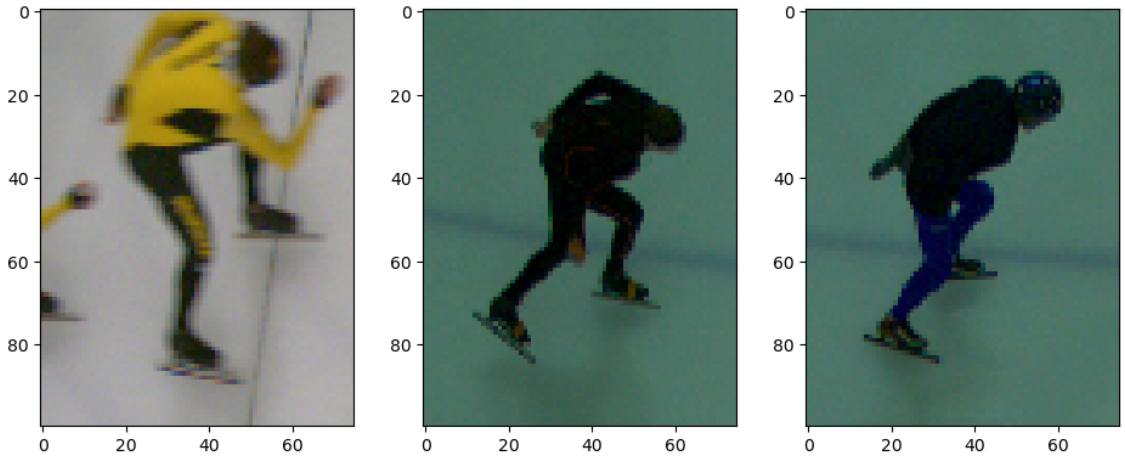


Figure 3: Example images from the dataset.

The first part of the data is recorded a couple of years ago and includes videos of professionals whom are part of the biggest commercial speed skating teams in the Netherlands. This leads to frames of speed skaters wearing suits with various colors, such as yellow, green and red. This data consists of 1457 images (video frames) of which each entry contains an ID number, a video frame number, the x- and y-coordinates of the location of the bounding box in the original video frame and finally eight values for the x- and y-coordinates of the front and rear endpoint of both ice skates. These images are collected from 34 different sequences of frames, extracted from videos. Each sequence of frames contains frames from only one speed skater. This part of the dataset is owned and supplied by the supervisor of this study.

The second part of the data is recorded recently and includes videos of a professional junior speed skater. These speed skater wears a special motion capture suit, which is fully black. This data consists of 502 images of which each entry contains an ID number, the x- and y-coordinates of the location of the bounding box in the original video frame, the video frame number and finally eight values for the x- and y-coordinates of the front and rear endpoint of both ice skates. These images are collected from 6 different videos and all show the same speed skater.

The third part of the data includes videos of amateur skaters that were present at the recording day and these videos were also recorded recently. The skaters recorded here wear mostly black

and blue clothes, with some orange and white present. This data consists of 681 images of which each entry again contains an ID number, the x- and y-coordinates of the location of the bounding box in the original video frame, the video frame number and the eight values for the x- and y-coordinates of the front and rear endpoint of both ice skates. These images are collected from 3 different videos and show five different speed skaters. In total, this gives 2640 images to train and test the CNN model.

The older data has been annotated before and therefore needed no additional processing. However, the recently recorded data still needed annotation. This is done using a Python script which pops up all the frames one by one and allows the user to click on the front and rear endpoints of both skates. The script then draws two lines between these points for the user to verify that the annotation is as desired. If this is the case, the eight coordinate values are stored in a CSV file and the next frame pops up. This location data is combined afterwards with the data on the bounding box location in the original full frame of the video.

4.2 Applying the CNN model

After the preprocessing phase, the data is almost ready to be processed by the CNN model. Before the images are fed to the model, depending on the model used, we convert all images from the RGB color model to grayscale, reducing the size of the images from 75x100x3 to 75x100x1. Then we divide the full dataset into a training set and a validation/test set, where the training set contains 90% of the images and the validation/test set contains 10%. For the experiments, the computed loss values will be averaged over three runs. For each of these runs, the dataset will be randomly split again into a 90% training set and a 10% validation/test set.

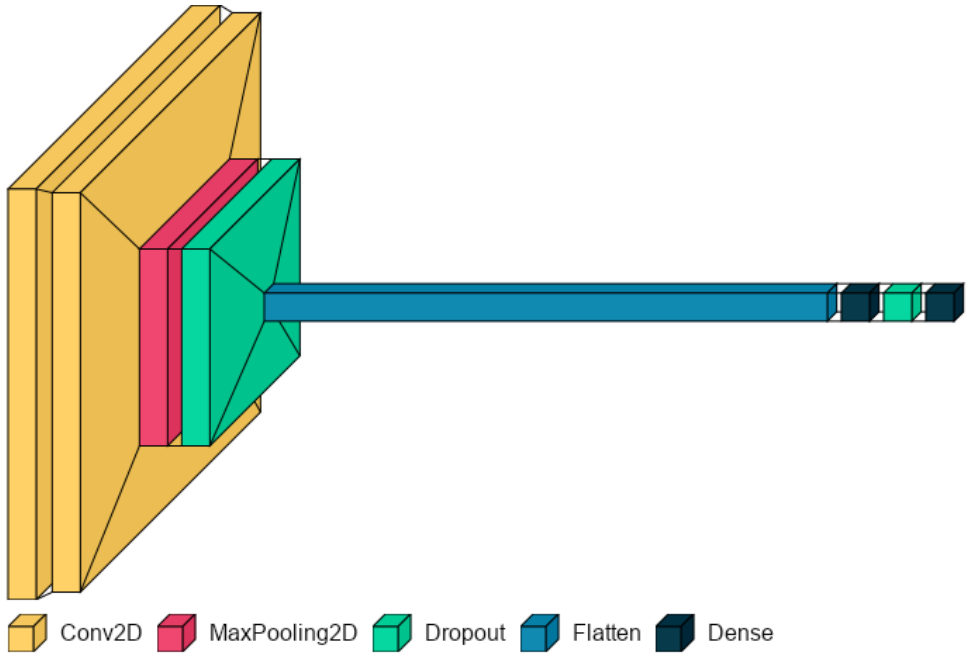


Figure 4: Graphical visualization of the architecture of the used CNN model. Visualization made with a Python script by [Gav20].

Figure 4 shows the architecture of the CNN model used in this study. In the first step the

training images are fed to the model in batches of 64 images. The first part of the CNN model consists of two convolutional layers, each containing 64 filters and a kernel of size 3x3. Both layers use the ReLU activation function and only apply padding when to prevent the kernel from ending unevenly around the edges. After these layers, a max pooling layer with a pool size of 2x2 is added, which reduces the spatial dimensions of the data, while preserving the most important data. Then a dropout layer is applied, to randomly set some nodes to zero, in this case with 0.25 probability. In the final part of the model, the data is flattened and fed to a fully connected layer of size 128, using the ReLU activation function, after which another dropout layer of 0.25 is added before the data arrives in the final output layer of size 8, where the linear activation function is applied. This results in eight outputs which predict the x- and y-coordinates of the location of both ice skate blades.

4.3 Prediction post-processing

After obtaining predictions for the ice skate blades locations, we want to use these predictions for further calculations. However, we first process the results to ensure that there are no clear outlying results present. To do this, all predictions of the ice skate blade locations, which are based on the bounding box, will be translated to their location in the full camera frame using the position and original size of the bounding box in the frame. Then we apply the RANSAC algorithm, which fits a model to the results excluding the outliers in the data. The resulting fitted model will be used to substitute the original predictions with new predictions based on this model, aiming to remove the majority of prediction error caused by the CNN model.

4.4 Trilateration

Following the steps discussed before, we end up with full frame predictions for the location of the ice skate blades in pixels. In order to use these results for further analysis, we need to translate these pixel locations to real-life locations, enabling the possibility to calculate various metrics such as the speed of the speed skater at some point in the corner. Before we can do this translation, we need to create a local coordinate system in world coordinates, so that we know where a point is in the world. This system is created using trilateration. We have two reference points A and B of which we know the location. We put nine markers down on the ice of which we do not know the exact location. Then we determine the distances between each of these markers and both reference points using a distance measuring device. Using the formulas discussed in Section 3.2, we can estimate the location of the nine markers. The local coordinate system resulting from this procedure is shown in Figure 5. Unfortunately, something has gone wrong with the distance measurements for marker 2, resulting in marker 2 not being close to where the marker was on the ice. Therefore this marker will be excluded from further calculations.

4.5 Homography

Now that the world coordinates of the eight markers are established, we have data on the marker locations in world coordinates. Since the markers are put on the ice in the frame of one of the cameras, we also have data on the marker locations in image coordinates. This will allow to translate any point of interest in pixels to world coordinates using a homography. As discussed before, all available point pairs will be used to estimate a perspective transform,

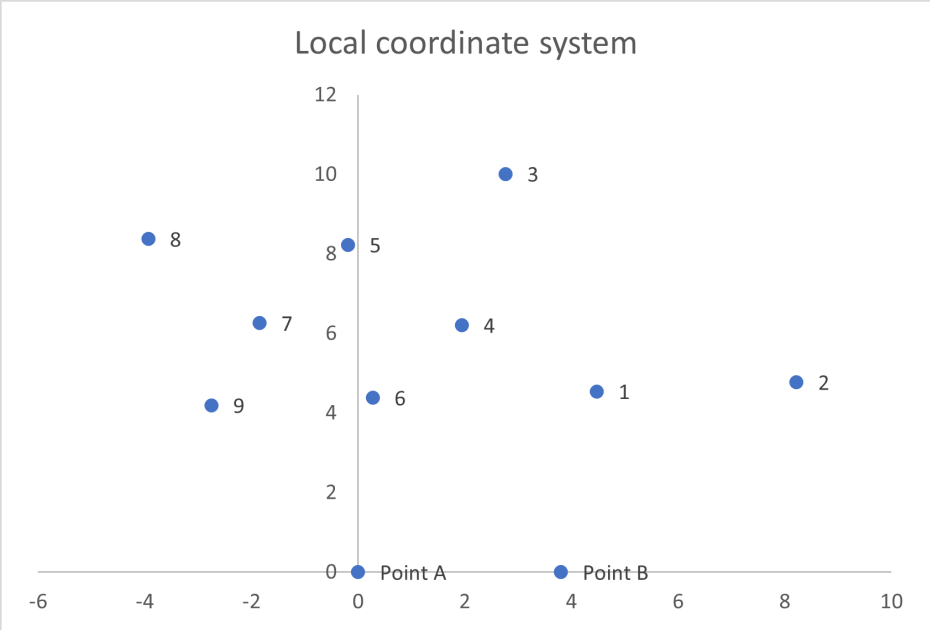


Figure 5: Visualization of the local coordinate system. The two reference points located on the side of the ice are placed on the x-axis and the nine markers located on the ice are marked and numbered in the system. Marker 2 will be excluded from future calculations.

which can then be used to translate any coordinates from plane to plane. Figure 6 shows the nine markers (including the erroneous marker 2) placed on the ice in the stadium in the camera frame. Using the obtained homography matrix and the obtained predictions for the location of the ice skate blades in a video frame, we can estimate the ice skate blade locations in world coordinates. This allows calculating various metrics such as the speed of the speed skater at some point in the camera frame. Having the location of the speed skater in each frame of a video will also allow to draw the trajectory of the speed skater through the camera frame, which can give insight into potential differences between laps. For this study, only this local system is used for just one camera, since it was infeasible to use local systems for all six cameras. Ultimately, for the bigger project this study is part of, the goal is to have a Thialf-wide coordinate system that can be used for all six cameras.

Figure 6: Visualization of the nine markers placed on the ice in the camera frame. The two reference points discussed in the trilateration process are located above the visible frame.

4.6 Travel distance and speed calculation

Now that we can determine the real-world locations of the ice skate blades, we can use this to calculate the distance that the speed skater travels between each frame. This distance is used in turn to calculate the speed that the speed skater travels between each frame. This will result in a dataset containing the speed of the speed skater at each point in a video. The distance travelled between two frames will be established by calculating the Euclidean distance between the real-life locations of the skates from two consecutive frames. The Euclidean distance formula is as follows:

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2} \tag{5}$$

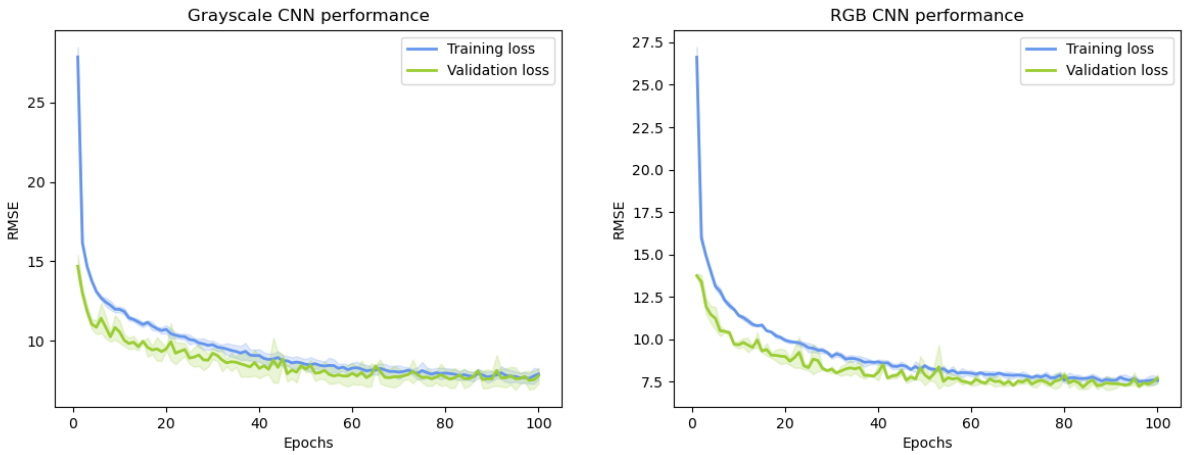
Here, p and q are two points with coordinates (p_1, p_2) and (q_1, q_2) respectively. The distance is calculated separately for the left leg and the right leg, because both legs move at different speeds during ice skating strokes. The points that will be compared for the distance calculation are the center points of the ice skate blades, which are determined by taking the average of the front and end point of the skate. After this procedure, the calculated distance will be multiplied by the frame rate the camera uses to record a video, which is 97 for this study, to obtain the speed in metres per second (m/s) and will again be multiplied by 3.6 to get the speed in kilometers per hour (km/h). Doing this for every pair of consecutive frames in a video will show the speed of the speed skater through the video. In order to reduce the noise in the graphs, we apply Gaussian smoothing with a value of 1.5 for the standard deviation of the Gaussian kernel.

5 Results

The results of this research will be discussed divided over the different steps in the process. Section 5.1 discusses the performance of the CNN models and Section 5.2 shows the results of applying the RANSAC algorithm. Section 5.3 discusses the results of applying the homography in terms of trajectory of the speed skater and Section 5.4 analyses the results of calculating the travelled distance and resulting speed of the speed skater. Section 5.5 discusses the results in more detail and finally Section 5.6 discusses the limitations of this study. From Section 5.2 onwards, the results are obtained using the second part of the dataset, which is recorded recently and includes 6 videos of a professional junior speed skater.

5.1 CNN model performance

The first part of this study is to predict where the blade of the ice skates of a speed skater are located in a frame of a video. For this we will use the CNN model explained before. During the training of the model, the Adam optimizer is used and the loss function is measured using the root mean squared error. The model uses 100 epochs for training and a learning rate of 0.001. Figure 7a and Figure 7b show the training loss and the validation/test loss over the training epochs for the grayscale model and RGB model respectively. All values are averaged over three runs and the colored area in the figure represents the standard deviation of the mean.



(a) (b)

Figure 7: Visualization of the performance of the grayscale (7a) and the RGB (7b) CNN model. Both the training loss and the test loss are visualized, along with the standard deviation. Training and test loss end near each other after 100 epochs with an RMSE around 7.5. Both models perform roughly equally.

The results clearly show that there is not much difference between feeding the images as grayscale images or in their original RGB format to the model, despite the difference in number of inputs. The fact that the RGB model performs roughly the same as the grayscale model also shows that the difference in background color in the data images, as is visible in Figure 3, is fortunately not a big issue for the neural network and arguably even makes the model more robust. Looking at the RMSE values, we see that both models reach an RMSE at least close to 7.5, meaning that the average error of the model on predicting a coordinate is around

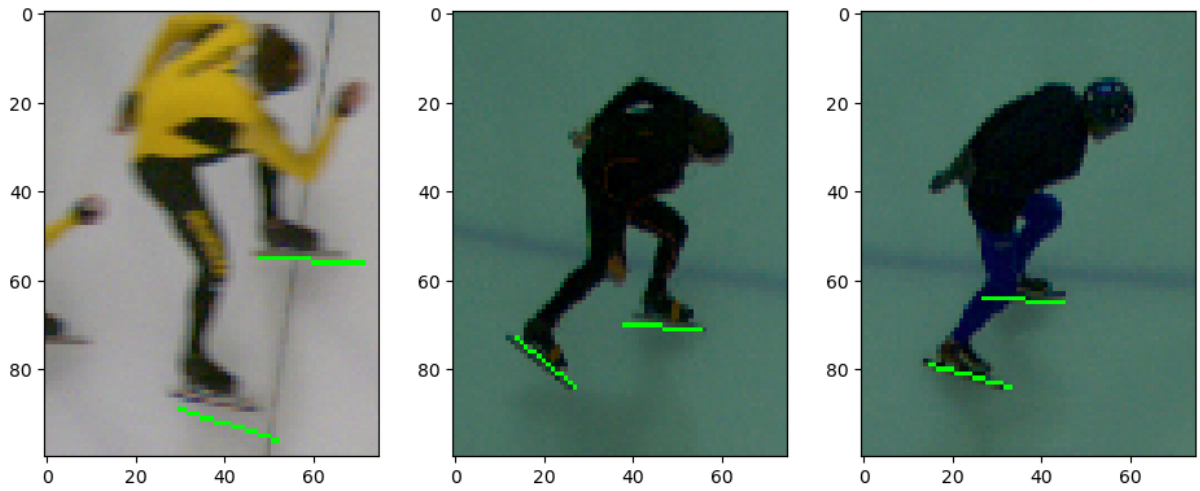


Figure 8: Three example images from the dataset with the predictions from the CNN model. Low loss after training is reflected here, as all predictions are fairly close to the ground truth.

7.5 pixels, which is good enough to use for the next calculations. The area of the standard deviation is also consistently small during the training process.

Figure 8 shows three images from the dataset as examples for the performance of the CNN model. The predictions are shown here by the green lines and are produced using the grayscale model. These images are good examples of the low loss values that we observed at the end of training. Especially in image two and three, the predictions are very close to the ground truth, while there is some error in image one for the right skate prediction.

5.2 RANSAC application

Now that we have a solid CNN model for predictions, we will apply the RANSAC algorithm to the predictions to fit a model that omits the outliers. As said before, this model will be used to substitute the original predictions with new predictions based on this model, to remove the majority of leftover prediction error caused by the CNN model. Figure 9 and Figure 10 show the predictions made by the CNN model for each of the eight ice skate blade location variables and the RANSAC fitted model. The graphs depicting a skate coordinate with a 1 in the name concern the rear endpoint of a skate and the graphs with a 2 in the name concern the front endpoint of a skate. The results shown here are generated using video 1 and video 5, while the results for the other videos can be found in Appendix A. Each iteration of RANSAC takes a random sample of size 20 and the polynomial degree of the fitted model is 8 for all videos.

The results in Figure 9 show that the predictions for the x-coordinates of the ice skate blade locations are very consistent and the graphs depict a very gradual progression in the data. In all four x-coordinate graphs, all points seem to at least touch the fitted RANSAC model and all values are shown in green, which shows that there are no outliers present in these predictions. Looking at the right skate y-coordinate predictions, the results show a bit more spread of data points around the RANSAC model line. Still, these predictions are very consistent and the RANSAC model did not mark any outliers here either. The most fluctuation in predictions can

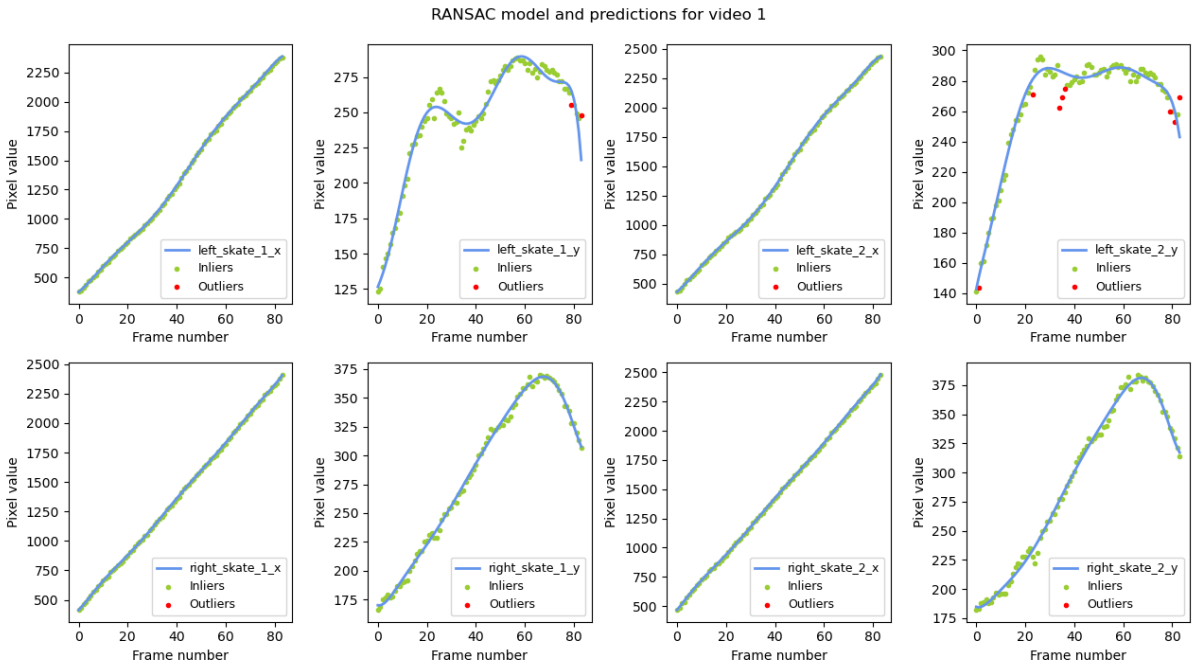


Figure 9: Visualization of the fitted RANSAC model to the predictions of the CNN model for video 1. Each graph shows the predictions for one of the eight predicted variables for the location of the ice skate blades. The predictions for the x-coordinates seem very good and no outliers are present. Especially for the left skate y-coordinates, the RANSAC algorithm detects some outliers, but not too many.

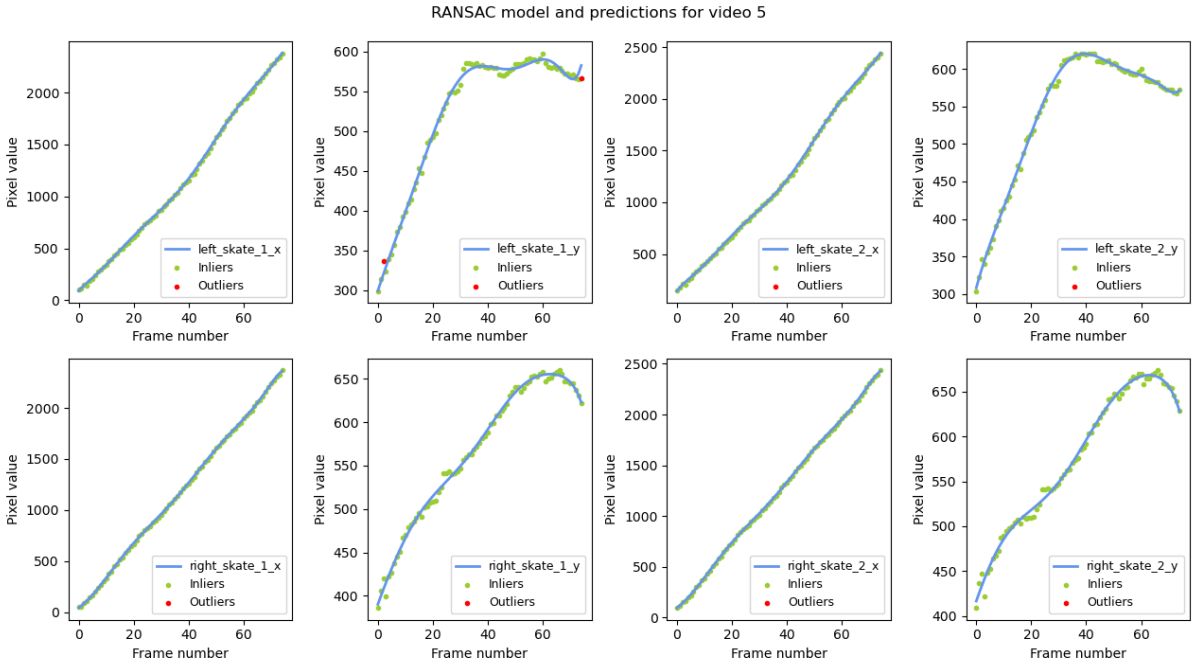


Figure 10: Visualization of the fitted RANSAC model to the predictions of the CNN model for video 5. Each graph shows the predictions for one of the eight predicted variables for the location of the ice skate blades. Some small differences in y-coordinate graphs are visible compared to video 1, but hardly any outliers are present.

be seen in the y-coordinate predictions of the left skate. Here the data shows that it is more difficult to realise a fitted model without any outliers. Looking at Figure 10, we observe similar results in the x-coordinate graphs and the right skate y-coordinate graphs of video 5 compared to the results in video 1. However, there are differences between the videos present in the left skate y-coordinate graphs. We observe that these graphs from video 5 follow a different path after the initial part of the video than the graphs from video 1. This is likely due to a difference in ice skating motion captured in the videos. Finally, we also observe fewer outliers for the predictions in video 5 than the predictions in video 1.

RANSAC results other videos

All the results that we will discuss in this section are related to the application of RANSAC to the other videos that are available and these figures can be found in Appendix A. One thing that can be observed consistently in all figures is that the predictions for the x-coordinates are almost always perfectly fitted by the RANSAC model. Therefore we do not observe any outliers in these graphs. When there are outliers present in a graph, it is usually one of the graphs showing the y-coordinate of the left skate, suggesting more error during prediction. An interesting observation is that the data from video 2, depicted in Figure 13 in Appendix A, shows that the shape of the y-coordinate graphs for the left skate and the right skate seem to be swapped, compared to the shape of these graphs in the other videos.

5.3 Homography and video trajectory

The predictions that result from applying the RANSAC algorithm predict the pixel locations of the ice skate blades. However, in order to use these predictions for other calculations, such as speed calculations, we need to translate the pixel locations to real-life locations. As said before, we will use a homography to do this translation. Doing this for each prediction gives us the trajectory of the speed skater through the camera frame. Figure 11 shows for each of the six videos the predicted trajectory of the speed skater through this section of the corner.

Plotting the location of both skates in this system gives a good idea of the line taken by the speed skater through the corner. For instance, it is clear that in video five and six the speed skater took a wider line through the corner than in the other videos, which is confirmed by looking at the videos. Furthermore, we also see the ice skating stroke exercised by the speed skater through this part of the corner and with that, we see differences in this stroke over the laps. As an example, placing the right skate over the left skate happens earlier in video 4 than in video 2. These results can, when combined for all six cameras, arguably already give some insight into differences between laps and how that might affect lap times.

5.4 Speed estimation

One of the things that we can measure with the collected data is the estimated speed of the speed skater on a certain moment in the corner. Figure 12 shows the speed of both ice skates separately for the six videos. As discussed before, the speed is calculated using the real-life locations from consecutive frames and determining the distance travelled in between.

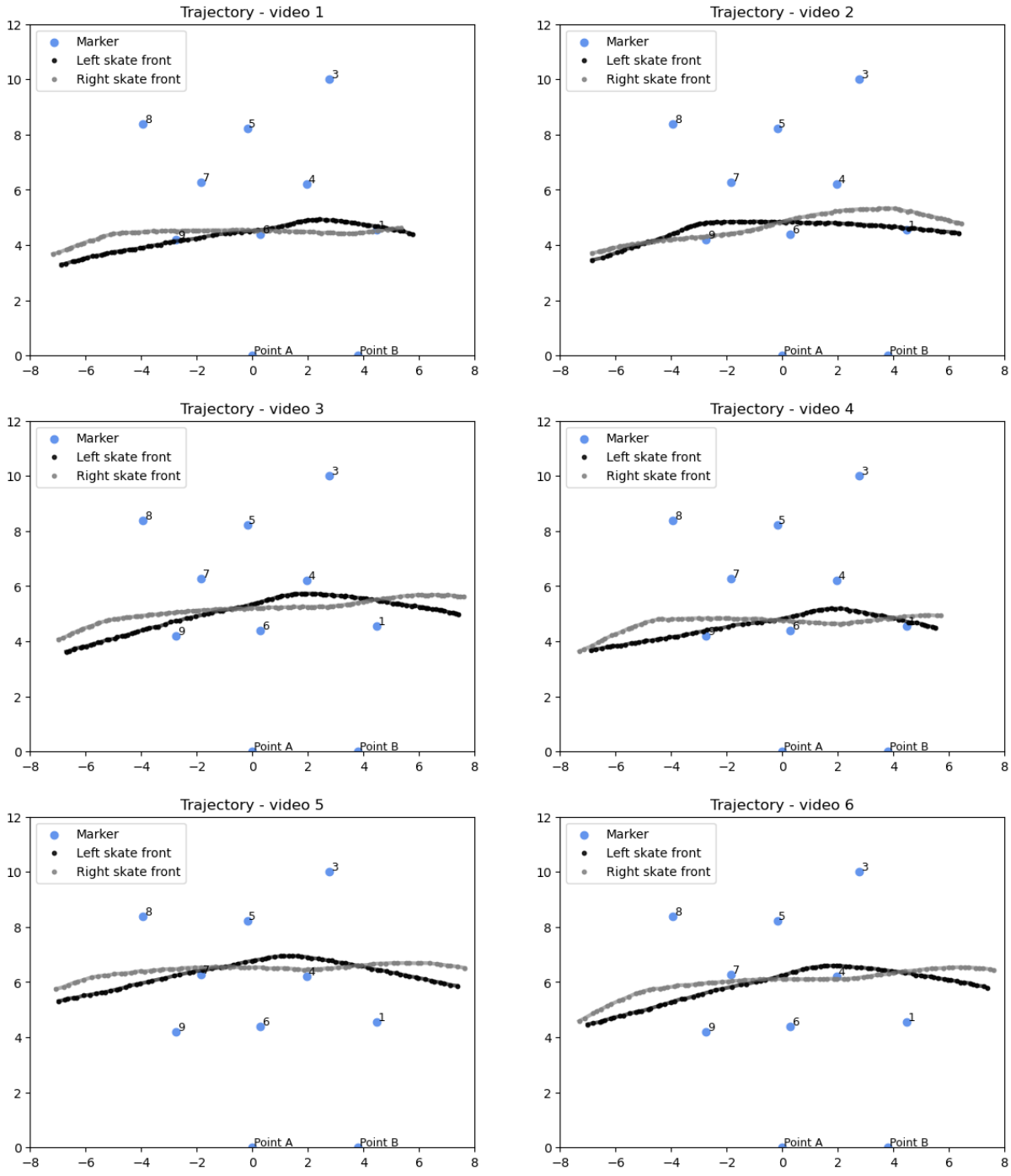


Figure 11: Visualization of the trajectory of the speed skater through this corner section for videos 1-6. The trajectories of both legs are shown separately. This visualization shows differences in line taken through the corner and different motions in the same part of the corner on different laps.

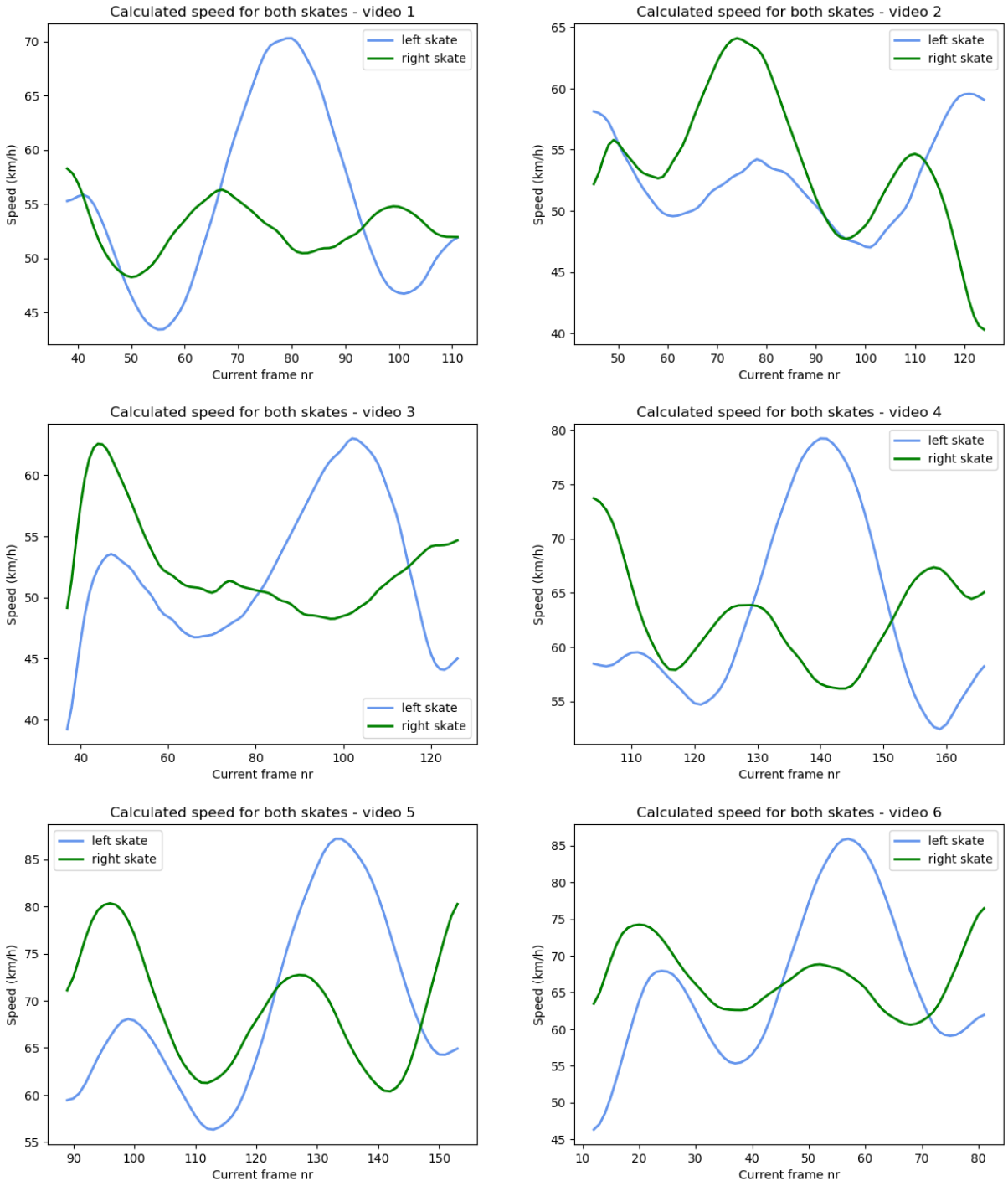


Figure 12: Visualization of the speed of both skates, calculated using the distance travelled between frames. The speed graph of the left skate looks very similar for most videos. There are big differences present in the speed interval between the videos.

Something that stands out in these graphs is that the left skate graphs for all videos except video 2 roughly follow the same pattern. Looking at the right skate graphs, we see that most graphs follow the same pattern, except for the one in video 3. Another thing that differs between the videos is the interval of the speed in which both skates are measured. Especially for videos 5 and 6, the maximum speeds are significantly higher than in video 3 for example. Finally, it is worth noting that in most videos speeds are recorded that are beyond the speed that is physically possible for the entire body to reach on ice. This is possible, because we track the distance travelled by the skates. For instance, if one of the skates is positioned behind the skater and the skater lifts their foot off the ice and drags it towards a position in front of the body, that foot "overtakes" the body and therefore registers a higher top speed than the full body will reach.

5.5 Discussion

CNN model performance

Both the neural network taking grayscale images as input and the neural network taking RGB colored images as input used in this study show good results and the RMSE of both models is around 7.5 pixels. Translating this to world coordinates, this gives an RMSE of around 4.5 centimeters. Looking at the figures depicting the loss during training, the downward trend seems to stabilise towards the end of training, suggesting that not much improvement can be expected after 100 epochs with the current dataset. Increasing the size of the current dataset might open up new possibilities for improvements of the model.

RANSAC

The results show that the x-coordinates of the locations of the ice skate blades follow an almost linear path. This is partially explained by the fact that a speed skater always skates through the camera view from left to right, meaning that the x-coordinates will automatically increase as the video goes on. Despite the fact that a speed skater moves its legs back and forth while skating, the flow of the graph keeps its consistency. This is probably due to the gliding effect, meaning that a speed skater will still move forward despite moving one of the legs backwards, because the other skate is still gliding over the ice. Based on how close all predictions are to the RANSAC fitted model, it seems like predicting the x-coordinate of an ice skate is fairly straightforward for the neural network.

The y-coordinate predictions of the right skate are also very consistent and the RANSAC model hardly marks any outliers for these skate coordinates. This is likely due to the fact that the right skate is almost always fully visible from the camera's position, as Figure 3 shows. This is probably the main reason though that the predictions for the y-coordinates of the left skate show more variety and more outliers. There are many video frames where only part of the left ice skate and blade are visible and on some frames at least the blade is fully hidden behind the legs. This is simply more difficult for the neural network to learn and predict and therefore there will be more prediction error here.

Another thing that stood out in the data of the videos is that the shape of the y-coordinate graphs for the left skate and the right skate of video 2 seem to be swapped (see Figure 13), compared to the shape of these graphs in the other videos. This can probably be at least

partially explained by the difference in segment of the ice skating motion that was captured in these videos, which is supported by the trajectories in Figure 11.

Speed measurements

One thing that follows from the graphs in Figure 12 is that the left skate graphs roughly follow the same pattern for most videos. This effect can be explained by the fact that most videos recorded the same section of the skating stroke on the different laps. This means that the speed skater usually exercised the same section of his stroke while skating through this camera frame. These observations confirm why the results, both the earlier discussed RANSAC results and the speed results, show a different picture for video 2 than for the other videos. Figure 11 shows that in video 2 the speed skater executed a different section of the skating stroke in the camera frame compared to the other videos. Here, the speed skater only started placing his right skate over the left skate in the last part of the camera frame instead of earlier. It is unsurprising that this difference in graphs is explicitly present, since we currently use videos from just one camera. When it is possible in the future to make graphs based on all six cameras, we will have a better overview on the exercised ice skating stroke through the corner by the speed skater.

The graphs also show a difference in speed intervals recorded from the videos. Videos four, five and six show a significantly higher maximum speed reached than the other three videos. The visualizations in Figure 11 give the main reason for this. For videos five and six, Figure 11 shows that the speed skater took a wider line through the frame of the camera here. This results in the speed skater entering the frame more horizontally and slightly lower in the frame, meaning that the angle between the speed skater and the camera is smaller, which influences the covered distance estimation. A part of the difference in entering the frame discussed here is likely due to the camera setup not being perfect, which will be discussed in more detail in Section 5.6. The reason that video four has a higher maximum recorded speed, is probably due to a higher stroke frequency on this lap in this part of the corner, based on observing the recorded video. The observations discussed above show that when this system is used for comparison over different laps, the user will need to make sure that some parameters are equal for the videos, in order to be able to make an accurate comparison.

5.6 Limitations

There were some limitations during this research. First of all, due to time constraints at the recording day in Thialf, the region of interest of the cameras and the lighting are not ideal for the newer part of the dataset. This results for instance in a difference in background color between the newer part and older part of the dataset, which can be noticed in Figure 3. This limitation will be improved for future studies in this project, but this is too late for this study. Despite the difference in background color, the results show no decline in the performance of the CNN model. The aforementioned time constraints are also the main reason that for this study only one local coordinate system is used for just one camera. This means that we only have the ability currently to measure elements such as the speed in the frames of one camera, while an overview of the entire corner is preferred.

Another limitation is the method currently used to create the bounding boxes around the speed skater. Right now, the speed skater is located in a camera frame and the program automatically

draws a bounding box around the skater with some margin. However, this leads to a variable bounding box size, which is eventually downscaled to 75x100 for the neural network models. This will be improved for future studies by having a method that pads all drawn bounding boxes to a certain size, resulting in bounding boxes that are all of equal size.

6 Conclusions and Further Research

This research was conducted to find answers to the question how the Apex camera system can aid in analyzing speed skating performance. Two CNN models have successfully been trained to locate the ice skates in a camera frame automatically. The prediction error is around 7.5 RMSE for both models after training for 100 epochs, which is good enough to work with in further calculations. The application of the RANSAC algorithm afterwards also helps to minimize the prediction loss as much as possible. The predicted location coordinates in terms of pixels were successfully translated to real-life location coordinates using trilateration and then a homography. With this, the trajectory of the speed skater through the corner could be established. This also enabled us to calculate the distance travelled between frames and in turn allowed for calculation of the speed of the speed skater through the camera frame. This research shows that the use of the Apex camera system and the development of additional software can aid in analyzing a speed skater's performance in several ways, such as analysis on the speed and the trajectory of a speed skater through the corner.

There is still plenty of room for future research to build upon this research. As mentioned before, this study is part of a wider project that aims to have a functional tracking system in Thialf, which can be used by speed skaters and their teams to analyse their laps around the track. Further research within this project can be conducted on using all six cameras to generate a complete view of how a speed skater traverses through the corner. Currently, only videos from one camera are used for the measurements and the local coordinate system used is only based on one camera frame. Ideally in the future this coordinate system is a Thialf-wide coordinate system that can be used by all six cameras installed. This will enable users to use videos from all six cameras and measure elements such as the speed over the entire corner, which will give a more complete view on the speed skater through the corner of the track.

The current size of the dataset is good enough to obtain the results discussed in this study. However, we believe that there is still profit to be gained in terms of loss on the predictions made by the neural network models. By increasing the dataset with frames from more cameras and eventually conducting an analysis on which types of frames result in the most prediction error, the models should improve even further. This will also enable new measures to be used for analysis of the skating performance. In this study, we have introduced the calculation of the speed of both skates based on the distance travelled between frames. While this is an informative measure, there are plenty of other measures available, which might help speed skaters and their teams in analysing the corner traversal of the speed skater and compare different laps, aiming to find new details that will allow the speed skater to set faster lap times during races.

References

[BRB19] Masoud S Bahraini, Ahmad B Rad, and Mohammad Bozorg. Slam in dynamic environments: A deep learning approach for moving object tracking using ml-ransac algorithm. *Sensors*, 19(17):36–99, 2019.

[DSO08] Evgueni Doukhnitch, Muhammed Salamah, and Emre Ozen. An efficient approach for trilateration in 3d positioning. *Computer communications*, 31(17):4124–4129, 2008.

[ESTA14] Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable object detection using deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2147–2154, 2014.

[FB81] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[Gav20] Paul Gavrikov. visualkeras. https://github.com/paulgavrikov/visualkeras, 2020.

[GDDM15] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Region-based convolutional networks for accurate object detection and segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 38(1):142–158, 2015.

[JSK21] Ahmad Jalal, M Zeeshan Sarwar, and Kibum Kim. Rgb-d images for objects recognition using 3d point clouds and ransac plane fitting. In *2021 International Bhurban Conference on Applied Sciences and Technologies (IBCAST)*, pages 518–523. IEEE, 2021.

[KOH+17] Arno Knobbe, Jac Orie, Nico Hofman, Benjamin Van der Burgh, and Ricardo Cachucho. Sports analytics for professional speed skating. *Data Mining and Knowledge Discovery*, 31(6):1872–1902, 2017.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[LHB04] Yann LeCun, Fu Jie Huang, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages II–104. IEEE, 2004.

[LKC19] Seungkwan Lee, Suha Kwak, and Minsu Cho. Universal bounding box regression and its applications. In *Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part VI 14*, pages 373–387. Springer, 2019.

[LTC+09] GuoJun Liu, XiangLong Tang, Heng-Da Cheng, JianHua Huang, and JiaFeng Liu. A novel approach for tracking high speed skaters in sports using a panning camera. *Pattern recognition*, 42(11):2922–2935, 2009.

[MGGGP+19] Ernesto Martín-Gorostiza, Miguel A García-Garrido, Daniel Pizarro, Patricia Torres, Manuel Ocaña Miguel, and David Salido-Monzú. Infrared and camera fusion sensor for indoor positioning. In *2019 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–8. IEEE, 2019.

[MLAD17] Daniel Mas Montserrat, Qian Lin, Jan Allebach, and Edward J Delp. Training object detection and recognition cnn models using data augmentation. *Electronic Imaging*, 2017(10):27–36, 2017.

[MORMMS22] José María Martínez-Otzeta, Itsaso Rodríguez-Moreno, Iñigo Mendialdua, and Basilio Sierra. Ransac for robotic applications: A survey. *Sensors*, 23(1):327, 2022.

[MW22] Travis W Moleski and Jay P Wilhelm. Trilateration positioning using hybrid camera–lidar system with spherical landmark surface fitting. *Journal of Guidance, Control, and Dynamics*, 45(7):1213–1228, 2022.

[NFHDK13] Dionne A Noordhof, Carl Foster, Marco JM Hoozemans, and Jos J De Koning. Changes in speed skating velocity in relation to push-off effectiveness. *International journal of sports physiology and performance*, 8(2):188–194, 2013.

[OAEO13] OS Oguejiofor, AN Aniedu, HC Ejiofor, and AU Okolibe. Trilateration based localization algorithm for wireless sensor network. *International Journal of Science and Modern Engineering (IJISME)*, 1(10):2319–6386, 2013.

[OBLS15] Maxime Oquab, Léon Bottou, Ivan Laptev, and Josef Sivic. Is object localization for free?-weakly-supervised learning with convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 685–694, 2015.

[OHdKF14] Jac Orie, Nico Hofman, Jos J de Koning, and Carl Foster. Thirty-eight years of training distribution in olympic speed skaters. *International journal of sports physiology and performance*, 9(1):93–99, 2014.

[OHMK20] Jac Orie, Nico Hofman, Laurentius A Meerhoff, and Arno Knobbe. Training distribution in 1500-m speed skating: a case study of an olympic gold medalist. *International journal of sports physiology and performance*, 16(1):149–153, 2020.

[Pac21] Valeria M Salas Pacheco. *Trilateration-Based Localization in Known Environments with Object Detection*. PhD thesis, University of South Florida, 2021.

[PKK+18] Andrew Post, David Koncan, Marshall Kendall, Janie Cournoyer, J Michio Clark, Gabrielle Kosziwka, Wesley Chen, Santiago de Grau Amezcua, and T Blaine Hoshizaki. Analysis of speed accuracy using video analysis software. *Sports Engineering*, 21:235–241, 2018.

[RHGS15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[STE13] Christian Szegedy, Alexander Toshev, and Dumitru Erhan. Deep neural networks for object detection. *Advances in neural information processing systems*, 26, 2013.

[TR05] Federico Thomas and Lluís Ros. Revisiting trilateration for robot localization. *IEEE Transactions on robotics*, 21(1):93–101, 2005.

[VdKSVDHV16] E Van der Kruk, AL Schwab, FCT Van Der Helm, and HEJ Veeger. Getting the angles straight in speed skating: a validation study on an imu filter design to measure the lean angle of the skate on the straights. *Procedia engineering*, 147:590–595, 2016.

[vIS82] Gerrit Jan van Ingen Schenau. The influence of air friction in speed skating. *Journal of Biomechanics*, 15(6):449–458, 1982.

[ZLLZ19] Yu Zhou, Wenfei Liu, Xionghui Lu, and Xu Zhong. Single-camera trilateration. *Applied Sciences*, 9(24):5374, 2019.

[ZZXW19] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11):3212–3232, 2019.

A RANSAC results other videos

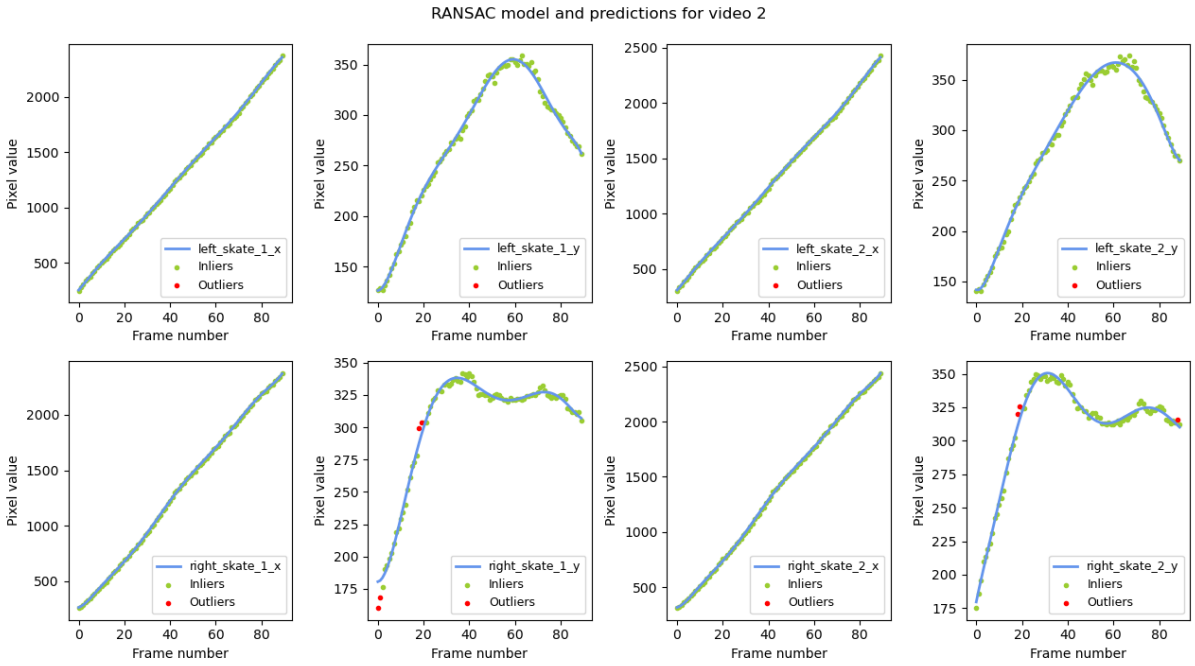


Figure 13: Visualization of the fitted RANSAC model to the predictions of the CNN model for video 2. Each graph shows the predictions for one of the eight predicted variables for the location of the ice skate blades.

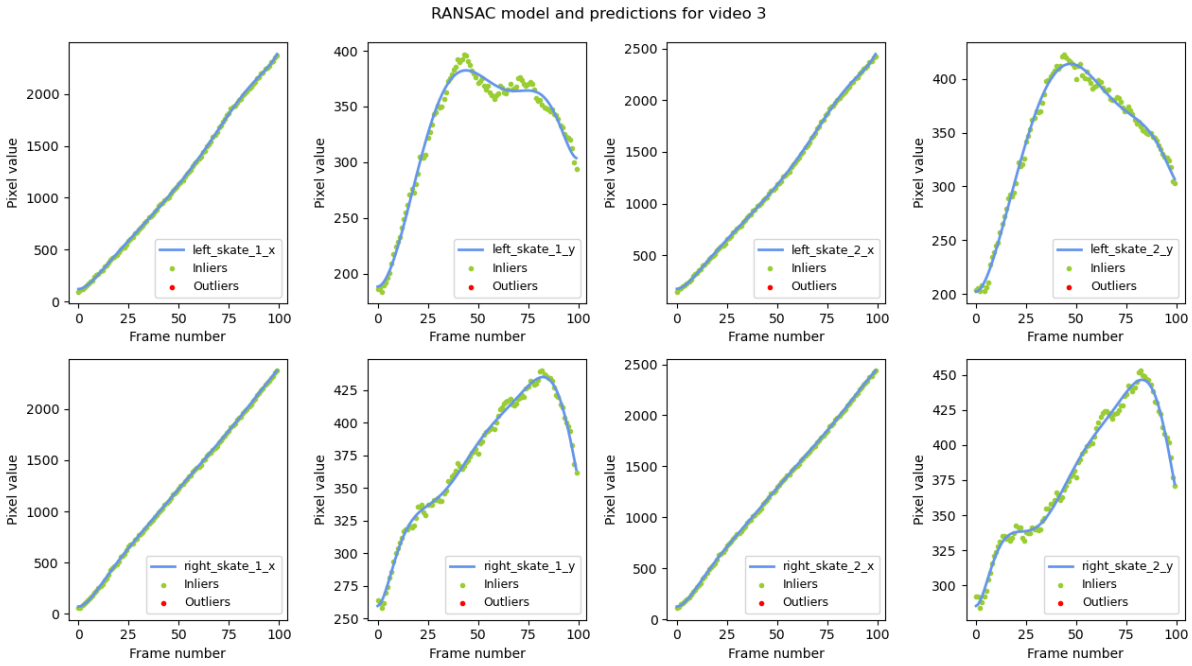


Figure 14: Visualization of the fitted RANSAC model to the predictions of the CNN model for video 3. Each graph shows the predictions for one of the eight predicted variables for the location of the ice skate blades.

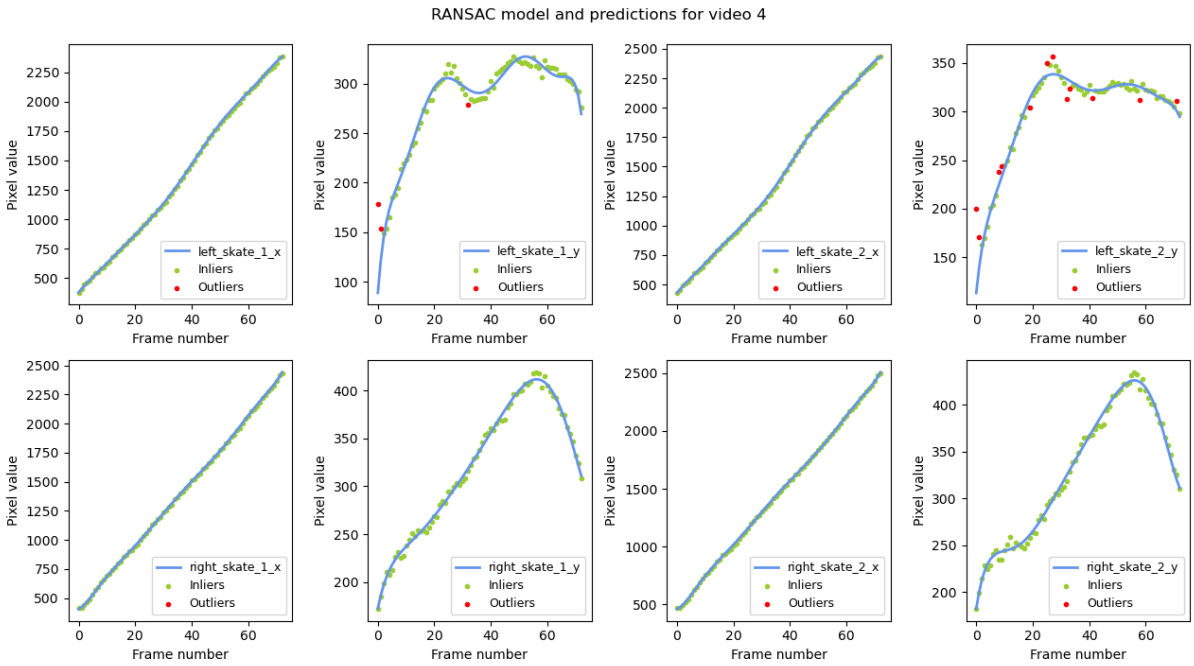


Figure 15: Visualization of the fitted RANSAC model to the predictions of the CNN model for video 4. Each graph shows the predictions for one of the eight predicted variables for the location of the ice skate blades.

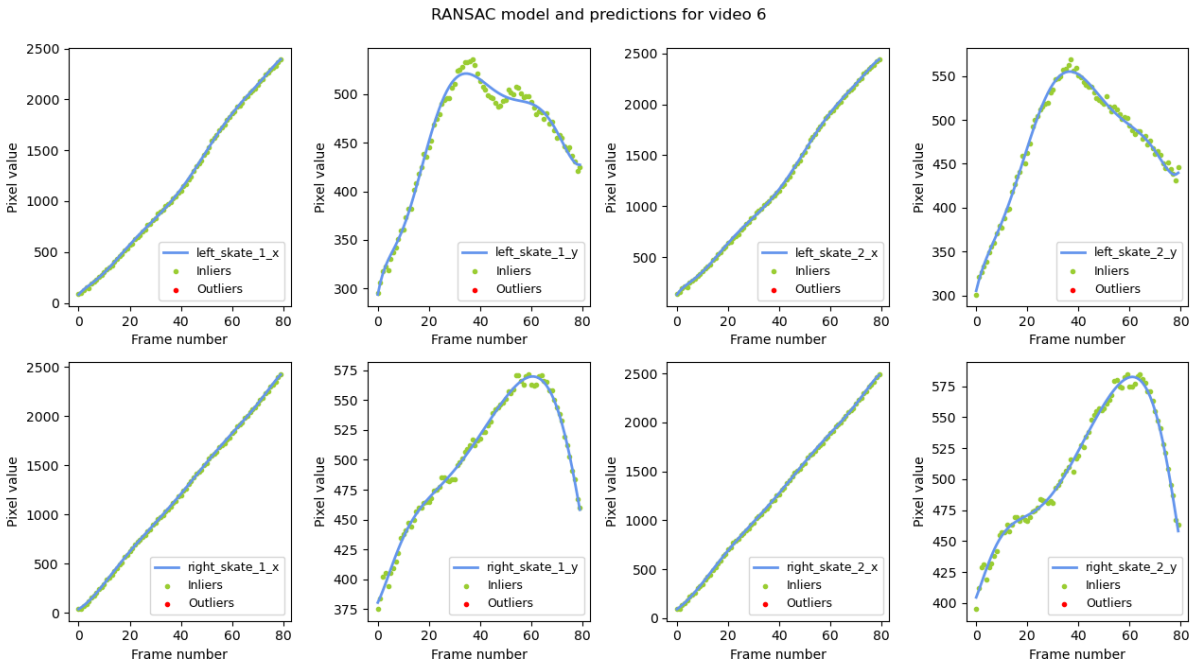


Figure 16: Visualization of the fitted RANSAC model to the predictions of the CNN model for video 6. Each graph shows the predictions for one of the eight predicted variables for the location of the ice skate blades.