



Universiteit
Leiden
The Netherlands

Bachelor Computer Science

Reconstructing and smoothing speed skater trajectories with
gaps and overlap in multi-camera motion data

Lohrenz Amatkanan

Supervisors: dr. Arno Kobbe, dr. Jeanette de Graaf

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

15/01/2026

Abstract

In this thesis we will investigate the reconstruction and smoothing of athlete trajectories in speed skating, using multiple camera tracking data from the Apex camera system. This system is based in the competitive speed skating arena Thialf, in the Netherlands. The research consists of two major challenges. The first involves interpolating missing data caused by dropped frames due to a limitation of the Apex camera system, particularly in the outer lane where skaters temporarily fall outside the field of view during camera transitions. The other challenge is resolving noise and inconsistencies caused by overlapping data in the inner lane, where the same skater is captured simultaneously during a camera transition. This results in duplicated trajectory segments that must be fused into a single continuous path. Interpolation methods are applied to ensure the transition between cameras is continuous, smooth and realistic. For these challenges, we are going to use two interpolation methods: linear and cubic spline interpolation. Additionally we apply two smoothing methods: the Kalman filter and Gaussian filter to try reducing noise and improving the traject accuracy even more. For evaluation, we used Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) to compare the results to annotated ground-truth data. This thesis aims to emphasize the trade-offs between interpolation methods and assess the effectiveness of smoothing in improving trajectory accuracy.

Contents

1	Introduction	1
1.1	Problem Definition	1
1.2	Research question	2
1.3	Thesis overview	2
2	Background and Related Work	3
2.1	Tracking in Multiple Camera Systems	3
2.2	Interpolation in Motion Tracking	3
2.2.1	Linear Interpolation	4
2.2.2	Spline Interpolation	6
2.2.3	Cubic Spline Interpolation	6
2.3	Smoothing Methods	8
2.3.1	Kalman Filter	9
2.3.2	Gaussian Filter	10
3	Methodology	12
3.1	Pipeline Overview	12
3.1.1	Input data	14
3.1.2	Segmentation Analysis	14
3.1.3	Interpolation and Merging	14
3.1.4	Smoothing and Filtering	15
3.1.5	Evaluation	15
3.2	Interpolation Methods	15

3.2.1	Estimating the number of missing frames	16
3.2.2	Linear Interpolation	17
3.2.3	Cubic Spline Interpolation	18
3.3	Handling overlaps	18
3.3.1	Overlap Detection	19
3.3.2	Merging Overlapping Segments	20
3.4	Smoothing Methods	22
3.4.1	Kalman Filter	22
3.4.2	Gaussian Filter	23
3.4.3	implementation and evaluation	23
3.5	Evaluation	24
3.5.1	Metrics and Variants	24
3.5.2	Interpretation	25
4	Experiment Setup	26
4.1	Dataset	26
4.2	Preprocessing and Aligning	26
4.2.1	Segment Stitching	27
4.2.2	Overlap Handling and Average Merge	27
4.2.3	Length Alignment	27
4.3	Experimental Conditions	28
4.4	Execution and Outputs	28
5	Results	29
5.1	Overview of Results	29
5.2	Interpolation Results	29
5.3	Parameter Selection for Overlap	41
5.3.1	Spline Parameter Tuning	41
5.3.2	Gaussian Filter Parameter Tuning	41
5.3.3	Kalman Filter Parameter Tuning	41
5.4	Overlap Results	42
5.4.1	RMSE and MAE	43
5.4.2	Visual Results	45
6	Discussion and Conclusion	50
6.1	Limitations	50
6.2	Future Work	51
6.3	Conclusion	51
	References	52

1 Introduction

In the last decade, the use of data driven methods has become really important in sports analytics, especially for performance monitoring and motion tracking. In sports such as speed skating, where even little improvements in trajectory and velocity can have effects on performance, the collection of precise motion data is essential. To achieve this, tracking systems based on multiple cameras and sensors are used, such as the Apex camera system [KdRM+25] installed in the competitive speed skating arena Thialf. Aside from detecting and tracking objects, the Apex system generates a large amount of positional data, that can be used for analyzing athlete trajectories and optimizing training [KOH+17]. However, as in other multi camera tracking systems [DHPJV06], these systems are often inconsistent and suffer from data loss, mainly due to occlusions and overlapping detection zones. Such limitations and inconsistencies lead to missing or noisy positional data, which could jeopardize the reliability of motion analyses.

Accurate trajectory reconstruction is important for evaluating technical performance, race strategy and even training load. In recent research on the Thialf Apex camera system [dR24, KdRM+25], Michael de Rooij observed that skaters moving in between camera zones may temporarily disappear from the view, which creates gaps in data, while overlapping camera regions can produce duplicated trajectories. These limitations, specific to the setup in Thialf, complicate performance analysis and require data reconstruction methods to produce continuous and realistic motion trajectory paths.

This thesis investigates how interpolation and filtering techniques can address these shortcomings in the Apex camera system. The challenge lies in interpolating missing data points and smoothing noisy segments caused by overlapping observations, while maintaining realistic motion behavior. The goal is to support reliable analyses of athlete performance while highlighting the strengths and limitations of different trajectory reconstruction approaches.

1.1 Problem Definition

Interpolation and filtering techniques are widely used in trajectory reconstruction across various domains, including sports such as soccer and basketball, as well as other applications where data often contain missing or noisy segments [DHPJV06]. However, these methods have not yet been evaluated in speed skating, where motion patterns differ significantly from team sports [KOH+17].

Skaters are moving at high speed along curved lanes, and frequent camera transitions lead to gaps or duplicated positions in the Apex data [dR24]. Although earlier studies have recognized these issues, there has been no research comparing how different reconstruction methods perform for this specific dataset. This thesis evaluates four common approaches that are widely used, linear interpolation, cubic spline interpolation, Kalman filtering and Gaussian filtering, to determine which methods give the most accurate and visually consistent reconstruction of speed skating trajectories. The results and conclusions of this comparison aim to show whether further improvements in trajectory reconstruction and performance analysis are needed for the Apex system.

1.2 Research question

The main goal of this research is to evaluate how different reconstruction methods perform on missing or inconsistent tracking data from the Apex system at Thialf. This research focuses on four widely used techniques: linear interpolation, cubic spline interpolation, Kalman filtering and Gaussian filtering. The objective is to compare their accuracy and visual quality when filling in missing data or smoothing overlapping segments. The objective is to compare their accuracy and visual quality and identify the limitations of each approach.

The research question of this thesis is:

”How can interpolation and filtering methods reconstruct missing coordinates and correct overlapping data from the Apex camera system in Thialf?”

To eventually answer the research question, the following sub-questions are defined:

- **How do linear interpolation, cubic spline interpolation, Kalman filtering and Gaussian filtering perform when reconstructing missing or overlapping data in the Apex dataset?**
- **Which method produces the lowest reconstruction error, measured in RMSE and MAE?**
- **How do the reconstructed trajectories visually compare in terms of smoothness and realism?**

1.3 Thesis overview

This thesis consists of six chapters.

- Chapter 1: Introduces the background, motivation, problem and presents the research question.
- Chapter 2: Discusses the theoretical background and related work on multi camera-tracking, interpolation and filtering methods.
- Chapter 3: Describes the dataset, preprocessing and the implementation of the reconstruction pipeline.
- Chapter 4: Presents the setup of the experiments and evaluation procedures.
- Chapter 5: Presents the results using RMSE and MAE and visual comparison.
- Chapter 6: Discusses the finding, strengths and weaknesses of each method. Additionally, the conclusion and gives suggestions for future research on trajectory reconstruction within the Apex system.

2 Background and Related Work

2.1 Tracking in Multiple Camera Systems

Motion tracking systems for tracking athletes with multiple cameras allow motion capture across large areas such as in the speed skating arena Thialf. By combining the camera views, these systems can synchronously follow the athletes movement, even when they move across different parts of the track. However, using this multiple-camera setup introduces a few challenges, such as synchronization, calibration and managing the overlapping and missing frames [DHPJV06].

Earlier work [DHPJV06] showed that in team sports, multiple camera tracking often suffers from occlusion, loss of detection and inconsistency of object association between the views of the camera. When these athletes cross each other and overlap in the field of view, the same person may be detected twice, while during camera transition, short gaps may appear. These kinds of problems lead to missing or duplicated trajectory data.

In terms of computer vision, Fischler and Bolles worked on a model estimation algorithm and introduced the Random Sample Consensus (RANSAC) [FB81]. This is a method designed to detect and ignore incorrect data points such as outliers when fitting a model. This principle has influenced many computer vision and multiple camera tracking techniques, where data from different views must be fused into a smooth trajectory.

In the Apex camera system, similar challenges occur. The setup consists of six synchronized cameras placed around the corner of the skating oval, each covering a specific part of the track. Michael de Rooij reported [dR24] that during sequential camera switches, skaters temporarily disappear from view in the outer lane, during transitions between cameras. In the inner lane, camera fields of view overlap, which causes double detection of the same skater. These overlapping detections lead to duplicated data points that represent nearly identical but slightly shifted coordinates. Because of these small differences in the overlap, the duplication is not immediately visible when inspecting the tracked data. A detailed explanation of how these overlapping detection were identified and handled is given in Section 3.1. Such effects result in fragmented trajectories that must be reconstructed into a single continuous path. Understanding these limitations is important, as they define the nature of the data imperfection studied in this thesis. In the following subsections, the methods used to correct these issues are discussed.

2.2 Interpolation in Motion Tracking

Interpolation is a mathematical technique used to estimate missing data between known observed data points. In motion tracking, interpolation is used to reconstruct missing frames/data or to smooth unexpected movements caused by the Apex multiple camera system that either dropped data or has double detections between camera transitions. The goal is to reconstruct a realistic and continuous trajectory that can follow the athlete movement as close as possible [MSBB⁺15].

In this section, multiple interpolation methods are considered. Linear interpolation is introduced as a baseline method due to simplicity, spline based interpolation is discussed as a more ad-

vanced method to capture curved motion better. Based on the characteristics of speed skating trajectories, cubic spline interpolation is selected and used for trajectory reconstruction in this thesis.

Assume a gap occurs between two consecutive camera segments, resulting in one or more missing data points between the last observation of camera i and the first observation of camera $i + 1$. When a gap occurs, interpolation is applied to reconstruct the missing data point(s) of the gap. In Figures 1 and 2, the camera observations are shown in grey, while the boundary and interpolated data point(s) are shown in blue. The blue line represents the predicted path, with red crosses indicating the interpolated values.

2.2.1 Linear Interpolation

Linear interpolation assumes that the change between two data points is constant over time. This interpolation method connects two observed data points followed by each other with a straight line and estimates the missing point based on a proportional distance between them. Thus, given two neighboring observations (x_i, y_i) and (x_{i+1}, y_{i+1}) any intermediate position is estimated along the straight\linear line connecting these points. This interpolation method is simple, fast and has a lot of effect when the gaps between two data points are very small and where the motion does not change direction significantly [KK23].

The formula used for linear interpolation for a frame at time t , between two data points (x_0, y_0) and (x_1, y_1) is:

$$x(t) = x_i + \frac{(t - t_i)}{(t_{i+1} - t_i)}(x_{i+1} - x_i) \quad y(t) = y_i + \frac{(t - t_i)}{(t_{i+1} - t_i)}(y_{i+1} - y_i)$$

Where index i denotes the segment between two consecutive observed frames from the last coordinate from cam i and cam $i + 1$. The fraction:

$$\frac{t - t_i}{t_{i+1} - t_i}$$

determines the position of the interpolated point between the two known observations.

Figure 1 shows the linear interpolation process in a two dimensional setting. The data points $(x_0, y_0), (x_1, y_1), (x_2, y_2), (x_3, y_3)$ represent observations of camera i and camera $i + 1$. The blue line indicates the interpolated value within the gap, where the number of missing points is estimated based on the distance between the last coordinate of cam i and the first coordinate of cam $i + 1$. The red cross shows the interpolated path predicted by linear interpolation, which lies directly on the straight line between two surrounding observations.

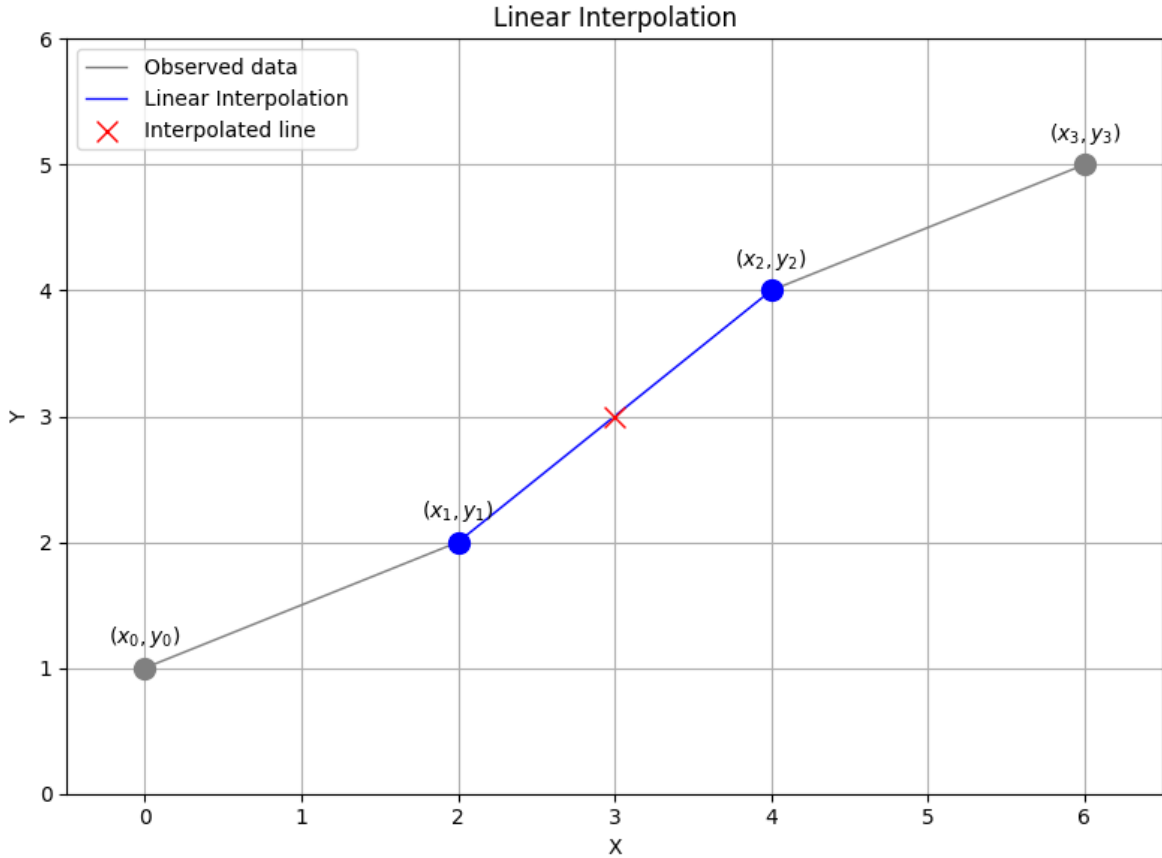


Figure 1: Linear Interpolation result of the outer lane trajectory.

To compute the missing data points shown in Figure 1, a constant velocity motion between the two observed data points (x_1, y_1) and (x_2, y_2) is assumed. Let t_1 be the frame of (x_1, y_1) and t_2 be the frame of (x_2, y_2) . For any missing frame at time t_k with $t_1 \leq t_k \leq t_2$, the interpolated coordinates are therefore defined as:

$$x(t_k) = x_0 + \frac{(t_k - t_1)}{(t_2 - t_1)}(x_2 - x_0) \quad y(t_k) = y_0 + \frac{(t_k - t_1)}{(t_2 - t_1)}(y_2 - y_0)$$

Using this method is computationally inexpensive and can quickly fill short gaps in our tracking data. However, it cannot capture curved motion or changes in direction. Thus, this method can lead to unrealistic trajectories when applying to longer missing segments or when the athlete's motion is non-linear. It is also sensitive to outliers [MSBB⁺15].

2.2.2 Spline Interpolation

The spline interpolation method estimates missing points by fitting piecewise polynomial functions between existing data points. Unlike linear interpolation, which connects two points with a straight line, spline interpolation makes sure that each segment connects smoothly to the following data point. The smoothness is achieved by enforcing continuity in both the position and the first derivative across all segments. The result of this spline interpolation is that the reconstructed trajectory becomes more realistic when the trajectory changes direction, like our case in speed skating [KK23, PHMS21].

In this thesis, trajectories are expressed as functions of time (frame index) and will be denoted by t . A segment in terms of splines refers to one piece of the piecewise spline defined between two consecutive time steps t_i and t_{i+1} . Each segment $S_i(t)$ on the interval $[t_i, t_{i+1}]$ between two consecutive observed data points (x_i, y_i) and (x_{i+1}, y_{i+1}) are represented by a polynomial of the form::

$$S_i(t) = a_i + b_i(t - t_i) + c_i(t - t_i)^2 + d_i(t - t_i)^3 + \dots, \quad t \in [t_i, t_{i+1}]$$

Where index i denotes the segment between the two consecutive observed data points (x_i, y_i) and (x_{i+1}, y_{i+1}) . The coefficients $a_i, b_i, c_i, d_i, \dots$ determine the shape of each segment. They are chosen such that the spline connects smoothly at the segment boundaries without any gaps (see Section 2.2.3 for more details). Spline interpolation avoids sudden changes in direction and produces a trajectory that follows the overall structure of the observed data points. This makes spline based interpolation suitable for motion tracking problems involving curved paths, such as speed skating [KK23, PHMS21]. The exact smoothness properties of the spline depend on the polynomial degree used. A widely used variant is the cubic spline, which is discussed in Section 2.2.3.

2.2.3 Cubic Spline Interpolation

A variant of the spline interpolation method is the cubic spline, where each local polynomial has degree three. The cubic spline is widely used in trajectory reconstruction, because of the balance the method gives in terms of computational efficiency and smoothness. They ensure continuity in both the first and second derivative, which creates naturally curved transitions without sharp corners [MSBB⁺15]. In contrast to linear interpolation, which assumes constant velocity between two observations, cubic splines interpolation can accurately model curved motion, which speed skating trajectories have. In addition to continuity in position and first derivative, cubic spline also enforce continuity of the second derivative. This results in smooth transition along the trajectory and prevents sharp corners or abrupt changes in direction.

The cubic spline consists of piecewise polynomial segments $S_i(t)$. Each segment is defined over a time interval $[t_i, t_{x+1}]$. Where t denotes the frame index and i indicates the spline segment between two consecutive observations. In this thesis, cubic spline interpolation is applied in a two-dimensional setting, by constructing separate spline functions for the spatial coordinates as functions over time. For each segment i , the spline functions are defined as follows:

- One for the x -coordinates:

$$S_{x,i}(t) = a_i^x + b_i^x(t - t_i) + (c_i^x(t - t_i)^2 + d_i^x(t - t_i)^3), \quad t \in [t_i, t_{i+1}]$$

- One for the y -coordinates:

$$S_{y,i}(t) = a_i^y + b_i^y(t - t_i) + (c_i^y(t - t_i)^2 + d_i^y(t - t_i)^3), \quad t \in [t_i, t_{i+1}]$$

The coefficients $a_i^x, b_i^x, c_i^x, d_i^x$ and $a_i^y, b_i^y, c_i^y, d_i^y$ determine the shape of the spline. These coefficients are chosen such that the segment connects smoothly at their boundaries, without visible gaps or sudden changes in direction. As a result, the reconstructed trajectory follows the overall shape of the data in a natural and realistic way. The coefficients $a_i^x, b_i^x, c_i^x, d_i^x$ and $a_i^y, b_i^y, c_i^y, d_i^y$ are chosen such that the spline satisfies the following constraints:

- Interpolation of data points:

$$\begin{aligned} S_{x,i}(t_i) &= x_i, & S_{x,i}(t_{i+1}) &= x_{i+1} \\ S_{y,i}(t_i) &= y_i, & S_{y,i}(t_{i+1}) &= y_{i+1} \end{aligned}$$

This condition ensures positional continuity, which means that the spline passes exactly through all observations.

- Continuity of the first derivative:

$$S'_{x,i}(t_{i+1}) = S'_{x,i+1}(t_{i+1}), \quad S'_{y,i}(t_{i+1}) = S'_{y,i+1}(t_{i+1})$$

The first derivative corresponds to the velocity of the trajectory. Enforcing continuity of the first derivative ensures that the estimated velocity does not change abruptly between segments.

- Continuity of the second derivative:

$$S''_{x,i}(t_{i+1}) = S''_{x,i+1}(t_{i+1}), \quad S''_{y,i}(t_{i+1}) = S''_{y,i+1}(t_{i+1})$$

The second derivative corresponds to the acceleration of the motion. This constraint ensures smooth curvature transitions and prevents sudden changes in acceleration. Resulting in a realistic trajectory.

These constraints ensure that the resulting trajectory is smooth in both direction and curvature across all spline segments. Because the second derivative rule are enforced globally, the coefficients of each segment are influenced not only by its direct neighbors, but also by the overall structure of the trajectory.

Figure 2 shows the cubic spline interpolation process in a two dimensional setting of this thesis. The data points $(x_0, y_0), (x_1, y_1), (x_4, y_4), (x_5, y_5)$ represent observations of camera i and camera $i + 1$ before and after a gap. The blue lines with the red crosses indicate the predicted trajectory within the gap, where the missing points, in this case (x_2, y_2) and (x_3, y_3) , is estimated by the euclidean distance between the last coordinate of camera i and the first coordinate of camera $i + 1$, in this case (x_1, y_1) and (x_4, y_4) . Note that the shape of splines are influenced not only by the surrounding observation of the apex camera system, but also by the overall shape and coordinates of the trajectory defined by neighboring data points.

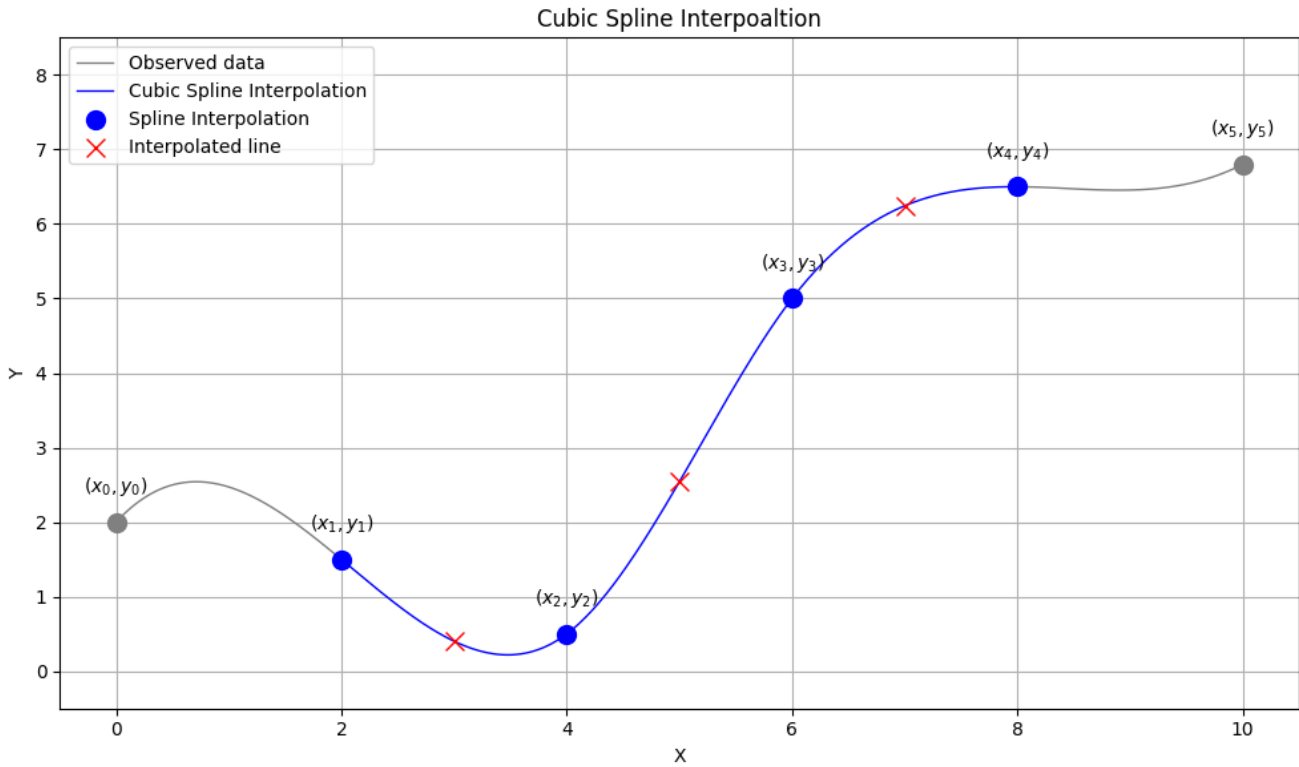


Figure 2: Spline Interpolation result of the outer lane trajectory.

Cubic spline interpolation has been shown to be very effective for trajectory based motion analysis in sports and vehicle tracking, as it accurately captures curved motion while maintaining stability [PHMS21]. For this reason, in addition to linear interpolation, cubic spline interpolation is also selected as an interpolation method for reconstructing missing trajectory segments in the outer lane data set in this thesis.

Both spline and linear interpolation are well-known methods for reconstructing missing data points in trajectory analysis. Spline interpolation offers smoother and more realistic results for curved motion, while linear interpolation remains computationally inexpensive and simple to implement. In this thesis, both methods are implemented to reconstruct missing trajectory segments in the dataset gathered from the Apex camera system.

2.3 Smoothing Methods

Smoothing methods are used to reduce random noise in measurement data and to produce a more stable and realistic trajectory. In motion tracking systems such as the Apex camera system, small variations can appear in the measured positions due to imperfections in the multiple camera setup or minor detection flaws. These small variations are often referred to as measurement noise, meaning that the measured position can differ from the skaters true position. Smoothing algorithms help to minimize noise by estimating the most likely true trajectory based on the observed data. This thesis discusses two smoothing approaches: the Kalman filter and the Gaussian filter. Both smoothing algorithms are well known and commonly used in tracking to correct noisy data and

improve reliability of reconstructed trajectories.

2.3.1 Kalman Filter

The Kalman smoothing filter is a mathematical algorithm used to estimate the true state of a system from a series of noisy measurements. It provides a recursive solution to the problem of estimating unknown variables that evolve over time. This approach was first introduced by R.E. Kalman in 1960 [Kal60]. While the original paper of Kalman was a little inaccessible, a more recent and accessible description of the Kalman filter equation is given by Qiang Li [LLJD15], who also discusses the use in areas such as target tracking and navigation. The Kalman filter combines two sources of information:

- The predicted state: This is derived from a mathematical model that describes how the system changes over time, for example assuming constant velocity.
- The measured state: This is obtained from sensor data that may contain measurement noise.

The Kalman filter combines the prediction from the motion model with the measured observation obtained from the Apex camera. Based on the estimated reliability of both sources, the filter determines how much influence each should have on the final estimate. This results in a new state estimate that is more reliable than using only the prediction or only the observation measurement. This characteristic makes the Kalman filter a good method for tracking object when observations are noisy or incomplete.

The Kalman filter operates by alternating between two main steps: a prediction step and an update step [Kal60, LLJD15]:

Prediction Step

In the prediction step. The Kalman filter estimates the future position of the skater based on the previously estimate state. This step assumes that the skater continues moving smoothly according to a motion model, which in this thesis is based on approximately constant velocity over short time intervals. At this stage, no new camera observation is used. The predicted position therefore represents where the skater is expected to be if the motion continues without interruption. Because this estimate relies only on the motion model and past observation, the uncertainty of the estimate increases over time during this step.

This prediction is important when measurements are temporarily missing or unreliable, such as in our case, during overlapping camera regions in the Apex system. It allows the trajectory estimate to remain continuous even when direct observations are not perfect.

Update Step

In the update step, the prediction state is corrected, using a new observation from the Apex camera system. The measured position of the skater is compared to the predicted position and the estimate is adjusted accordingly. The Kalman filter determines how much influence the measurement should have relative to the prediction depending on their estimated reliability. When measurements are noisy, the influence is reduced, while more reliable observations are followed more closely.

After the update step, the uncertainty of the estimate is reduced and the refined state is passed to the next prediction step. By repeating this process for each frame, the Kalman filter produces a smooth and stable trajectory.

The Kalman filter has already been used widely in fields such as robotics, aerospace navigation and object tracking. In sports analytics, similar filtering approaches have been explored to reduce noise and reconstruct trajectories, although the Kalman filter itself has not yet been applied to the Apex speed skating data. In this thesis, the Kalman filter is implemented and evaluated, as a potential smoothing method for reconstructing a continuous and realistic trajectory for speed skaters. Compared with interpolation, which is used for reconstructing missing points, the Kalman filter also filters existing measurements, reducing the effect of measurement noise while maintaining a consistent estimate of motion tracking.

2.3.2 Gaussian Filter

Gaussian smoothing is a simple kernel-based method for reducing local noise in a time series data. Instead of using a motion model, the method smooths the signal directly by taking a weighted average of neighboring data points over time. Data points close to the current frame will receive a higher weight, while data points further away receive less weight. These weights follow a Gaussian curve, controlled by the standard deviation σ , which determines how strong the smoothing is.

In this thesis, Gaussian smoothing is applied to the reconstructed trajectories of speed skaters captured by the Apex camera system. The trajectory consist of world coordinates, represented as time series of spatial positions. The Gaussian filter is applied independently on the x-coordinates and y-coordinates of the reconstructed trajectory. Let x_t denote the x -coordinates and y_t the y -coordinates of the speed skater at time index t (frame). The smoothed x-coordinate at index/frame t is defined as:

$$x_t^{smoothed} = \sum_{j=-k}^k w_j x_{t+j}, \quad \text{with} \quad \sum_{j=-k}^k w_j = 1$$

This also applies to the y -coordinate:

$$y_t^{smoothed} = \sum_{j=-k}^k w_j y_{t+j}, \quad \text{with} \quad \sum_{j=-k}^k w_j = 1$$

Here, j represents the offset relative to the current frame t . k defines the half width of the smoothing window. The Gaussian weights w_j are defined by:

$$w_j = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{j}{\sigma}\right)^2\right)$$

Here, σ determines the width of the Gaussian kernel and thus the degree of smoothing. Higher σ values smooth the signal more strongly, whereas lower values keep the trajectory closer to the original observations of the Apex camera system.

A similar Gaussian smoothing approach is used in structural health monitoring, where acceleration

signals of bridges are smoothed to remove high frequency noise and extract a clean structural response. Earlier research [KP25] shows that applying a Gaussian Kernel Smoothing Filter (GSKF) helps stabilize noisy vibration data and improves signal interpretation in dynamic environments, see [KP25]. Their method also relies on the weighted moving average form, which makes it almost similar to the approach used in this thesis. In this thesis, the same principle is applied to the two dimensional world coordinates captured by the Apex system. The coordinates $x_{(t)}$ and $y_{(t)}$ are smoothed independently using a one dimensional kernel: `scipy.ndimage.gaussian_filter1d`. The amount of smoothing is controlled by the parameters σ and `truncate`, which together determine the window length:

$$L = 2 \times \sigma \times \text{truncate} + 1$$

The Gaussian filter has already been widely used in many different fields such as in navigation and state estimation, where data are inconsistent or missing. In this thesis, it is applied as an additional method, alongside the interpolation and Kalman filtering methods. While interpolation fills missing points and both filters operate on the coordinate time series, Kalman filter use a motion model and state estimation, whereas the Gaussian filter applies direct temporal smoothing without a motion model. This allows us to evaluate whether Gaussian smoothing approach can improve the visual continuity and stability of reconstructed trajectory in the Apex data set.

3 Methodology

In this chapter, we describe the methodology used to reconstruct and evaluate athlete trajectories from the Apex multiple camera data set. This data set contains world coordinates of the speed skater’s trajectory, recorded by six synchronized cameras at the speed skating arena at Thialf. Two sets of data were used:

- Outer lane in the curved section with missing data between cameras at almost each transition.
- Inner lane in the curved section where the data overlap at almost each transition.

These sets represent the main reconstruction challenges: interpolation and overlap corrections, which we are going to describe in this chapter. A detailed description of the data set and its coordinate mapping process, where the cameras transform object detections into world coordinates, can be found in Chapter 4. The following sections outline how the processing pipeline, interpolation and filtering techniques and the evaluation has been implemented.

3.1 Pipeline Overview

The processing pipeline converts the raw segmented trajectories from the Apex camera system into smooth and continuous paths that can be evaluated. Each step of the workflow corresponds to a specific operation implemented in Python methods developed during the research. The two data sets, representing the outer and inner curved lanes of the speed skating arena, are described in detail in Chapter 4. An overview of both sets can be seen in detail in Figures 5 and 6, which show the raw trajectories from all six cameras captured by the Apex camera system, and show where gaps and overlaps occur between segments.

Figures 3 and 4 show the two main challenges that the pipeline faces. The first case (3) shows a clear spatial gap between the last frames/observations of one camera and the first frames/observations of the following camera. The second case (4) has two camera segments that overlap, which results in duplicate observations that require merging.

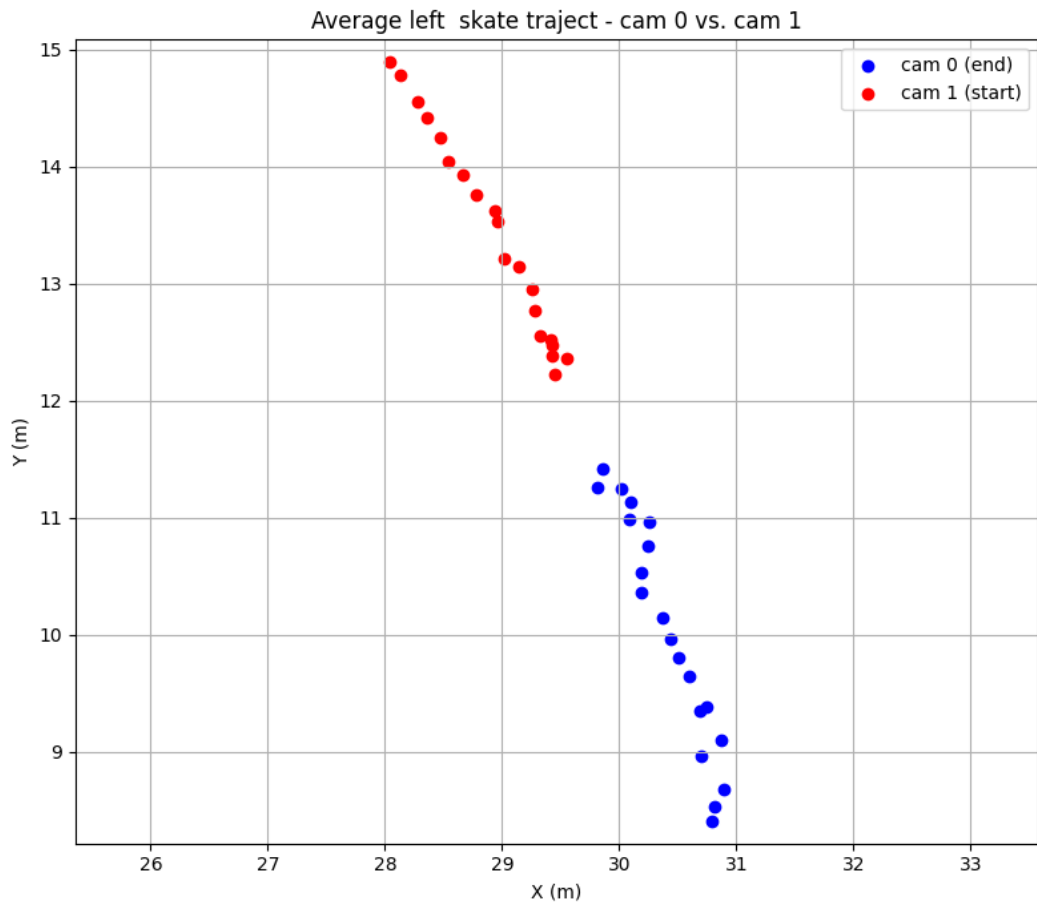


Figure 3: Missing observations: segment from camera 0 to camera 1

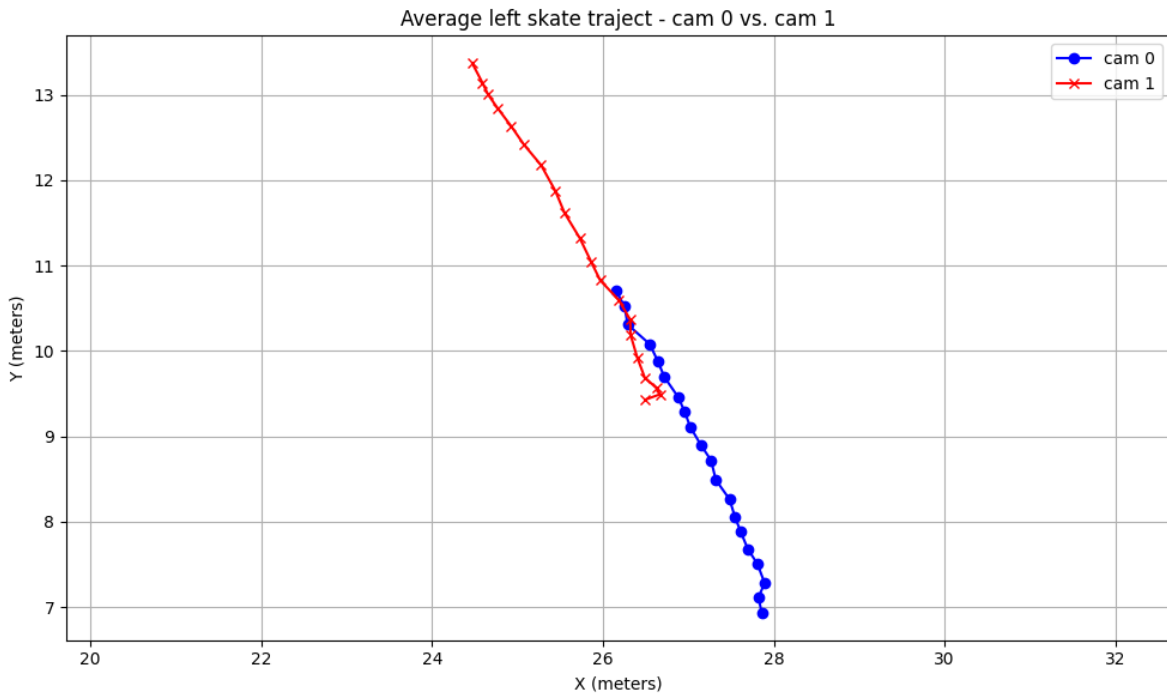


Figure 4: Overlapping observations: segment from camera 0 to camera 1

The overall reconstruction process is divided into several sections (see Sections 3.1.1 - 3.1.5), as described below.

3.1.1 Input data

The pipeline begins with loading the world coordinate .csv files for each of the six cameras in either the outer or inner lane data set. Each file contains frame numbers, therefore, each observation is converted to x and y positions for the front and rear of both the left and right skates. These measurements form the segmented trajectories that will later be merged or interpolated

3.1.2 Segmentation Analysis

Before reconstructing, consecutive camera segments are analyzed by plotting the data to determine their spatial relation at the transition:

- Gap: the skater disappears between cameras, producing missing frames. This occurs only in the outer lane.
- Overlap: the same skater is visible in both cameras simultaneously, which occurs only in the inner lane.

This relationship is determined by comparing the end frames/observations of one camera with the start of the following camera, using the Euclidean distance between coordinates.

This segmentation analysis directly determines which reconstruction strategy is applied. When a gap is detected between the end of one camera and the start of the next camera, indicating frames are missing, interpolation is applied to reconstruct the trajectory for continuity. When overlapping observations are detected, the trajectory is already continuous but affected by duplicated observations, hence measurement noise. Therefore, smoothing methods are applied instead.

In practice, this distinction is not always strict. As shown in Figure 6, the inner lane data set contains a trajectory composed of six camera segments with overlapping regions, while one transition still results in a small gap. In such cases, interpolation is applied as a fallback to bridge the missing frames. Then after no gaps occur, smoothing can be applied when there is overlapping data. Thus, segmentation analysis forms the decision step between interpolation and smoothing (see Sections 3.2 - 3.4 for details).

3.1.3 Interpolation and Merging

Depending on the result of the analysis, one of the two operations is applied:

- Interpolation: when a gap is detected, missing data points are filled using either linear interpolation or spline interpolation.
- Merging: when an overlap is detected, the duplicated observations are merged by averaging the matched pairs, optionally followed by smoothing.

3.1.4 Smoothing and Filtering

After reconstruction, optional smoothing is applied to reduce measurement noise from the series data points. Two filters are used: the Kalman filter, which assumes linear motion and a Gaussian filter, which smooths the signal without a motion model (see Chapter 2). Both are used as post-processing steps to improve or stabilize noise of the reconstructed trajectories.

3.1.5 Evaluation

The reconstructed trajectories are compared to manually annotated ground truth data in CSV format. To assess the real impact of the reconstructed trajectories, two evaluations are performed: one between the reconstructed trajectories and the ground truth, and second the original Apex measurements and the ground truth. Both evaluations use the RMSE and the MAE to measure how closely the coordinates match. This evaluation therefore helps to determine whether interpolation and filtering can contribute to the accuracy of the Apex data set.

3.2 Interpolation Methods

Interpolation is applied when a gap occurs between two camera transitions in the outer lane, as you can see in Figure 5. In this situation, the skater temporarily disappears between the consecutive cameras, resulting in missing coordinates. The goal of interpolation is to estimate these missing positions to produce a continuous and realistic trajectory. In this thesis, two interpolation techniques are used: Linear Interpolation and Spline Interpolation.

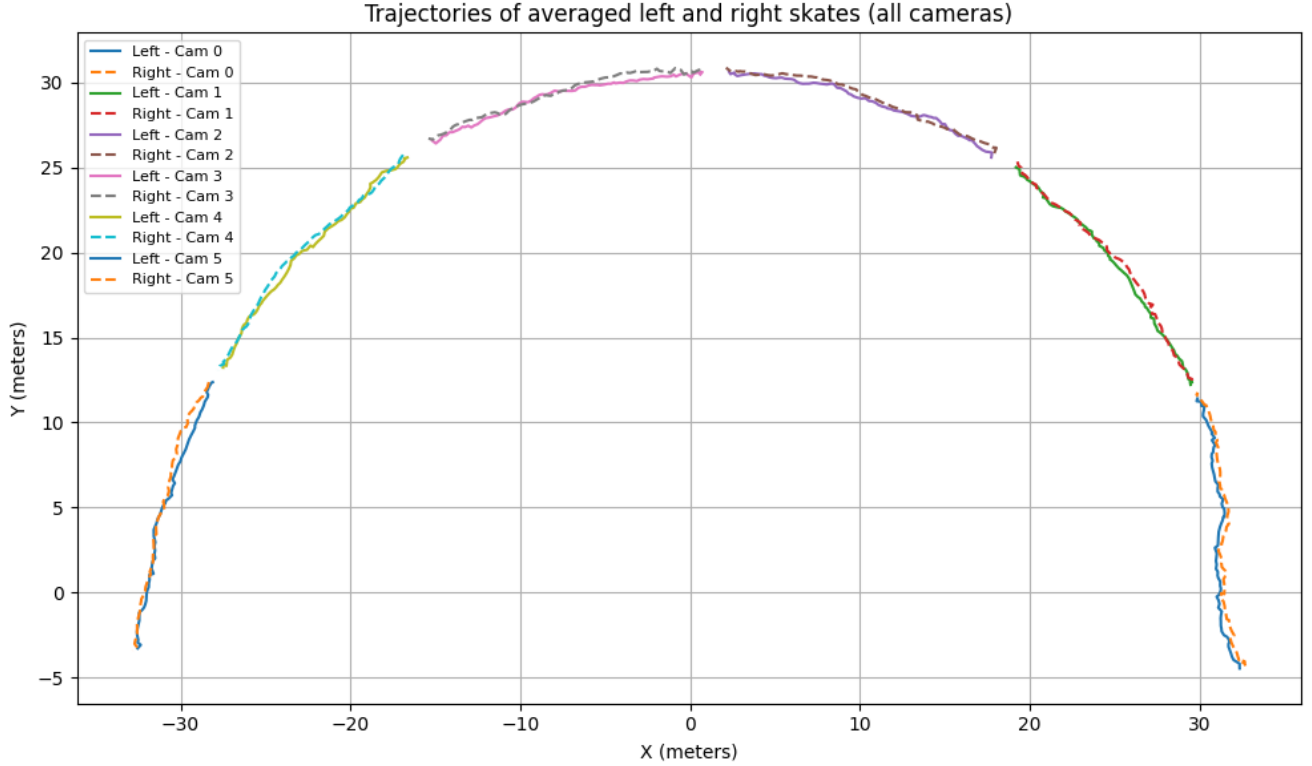


Figure 5: Outer lane of the speed skating arena in Thialf captured by the Apex system.

3.2.1 Estimating the number of missing frames

Before interpolation can be applied, the number of missing frames of each gap is estimated between two consecutive cameras. This is done by measuring how far apart the last point of camera i and the first point of camera $i + 1$ are in world coordinates. This distance is calculated using the Euclidean distance:

$$gap = \sqrt{(x_{i+1(\text{first})} - x_{i(\text{last})})^2 + (y_{i+1(\text{first})} - y_{i(\text{last})})^2}$$

After this is done, the average step size per frame is computed for both camera segments by taking the mean of all consecutive frame to frame euclidean distances of each camera:

$$d_i = \frac{1}{N_i - 1} \sum_{k=1}^{N_i-1} \sqrt{(x_{i,k+1} - x_{i,k})^2 + (y_{i,k+1} - y_{i,k})^2}$$

$$d_{i+1} = \frac{1}{N_{i+1} - 1} \sum_{k=1}^{N_{i+1}-1} \sqrt{(x_{i+1,k+1} - x_{i+1,k})^2 + (y_{i+1,k+1} - y_{i+1,k})^2}$$

Here, N_i denotes the number of observations captured by camera i and $(x_{i,k}, y_{i,k})$ represents the k -th world-coordinate measurement of camera i . The index k therefore iterates over consecutive frames within a camera segment. This also applies for camera $i + 1$. The two averages of camera i and camera $i + 1$ are combined to obtain the overall average step size:

$$d_{avg} = \frac{d_i + d_{i+1}}{2}$$

At last, the number of frames is estimated by dividing the *gap* by the d_{avg} and rounding to the nearest integer:

$$number\ of\ missing\ frames = round\left(\frac{gap}{d_{avg}} - 1\right)$$

However, not every transition between cameras always requires interpolation. In some cases, the distance between the last observation of camera i and the first observation of camera $i + 1$ is comparable to the normal frame-to-frame distance within a camera segment. In such situations, the trajectory is considered continuous and interpolation is not necessary.

To avoid unnecessary interpolation in these cases, a distance threshold is applied. The threshold is derived from the natural variability of the observed motion within the camera segments. First, the frame-to-frame distances from each camera trajectory are computed. From these distances, the mean step size and the standard deviation are obtained. To focus on the local behavior near the camera transition, the last 20 observations of camera i and the first 20 observations of camera $i + 1$ are used. The threshold is defined as:

$$threshold = d_{avg} + 1.5 \times d_{std}$$

Here, d_{avg} denote the average frame-to-frame distance computed from the segments of camera i and camera $i + 1$, d_{std} is the corresponding standard deviation.

If the measured gap between two consecutive camera segments is smaller than this threshold, the transition is treated as continuous and no interpolation is applied. Only when the gap is larger than the threshold, interpolation is performed.

3.2.2 Linear Interpolation

Linear interpolation provides a simple and direct way to reconstruct missing data points between two points. It assumes constant motion between these positions, such as when the skater moves in a straight line between the last frame of one camera and the first frame of the next. The interpolated coordinates $(x_{i,k}, y_{i,k})$ are calculated for $k = 1, \dots, n$ as:

$$x_{i,k} = x_{i,last} + (x_{i+1,first} - x_{i,last}) \frac{k}{n+1}, \quad k = 1, \dots, n$$

$$y_{i,k} = y_{i,last} + (y_{i+1,first} - y_{i,last}) \frac{k}{n+1}, \quad k = 1, \dots, n$$

Where $(x_{i,last}, y_{i,last})$ and $(x_{i+1,first}, y_{i+1,first})$ are the known positions before and after the gap, n is the number of missing data points between two consecutive camera segments and k indexes the interpolated points in the gap. This formula divides the straight line between the two known positions into equal segments, filling in the missing data points. As a result, the missing frames are filled by evenly spaced points along the linear path between the two camera segments. It works well for short gaps or for nearly straight motion, as in our case.

3.2.3 Cubic Spline Interpolation

In addition to linear interpolation, cubic spline interpolation is applied to reconstruct gaps between consecutive camera transitions. The mathematical definition of the cubic spline interpolation is described in detail in Section 2.2.3. This section is about how the cubic spline is applied within the reconstruction pipeline of the Apex camera system.

In this thesis, cubic spline interpolation is only used for reconstructing gaps, which means, intervals where no camera observations are available between the last frame of one camera and the first frame of the following camera. The observed camera segments themselves are not modified at all. For each detected gap a cubic spline is fitted between the boundary observations, using the estimated number of missing frames as described in Section 3.2.1. The cubic spline is constructed in a two-dimensional setting by applying the interpolation method independently to the x and y coordinates for each frame. Based on the frame indices of the observed data points and the predicted data points inside the gap, the cubic spline generates intermediate positions that ensure smooth transitions in velocity, acceleration and position across the camera boundary.

Compared to linear interpolation, cubic spline interpolation better suits the nature of speed skating trajectories, particularly in regions where the skaters direction changes gradually. By enforcing continuity in the first and second derivatives, the cubic spline reduces abrupt changes in direction, resulting a more physically plausible reconstructed motion (see Section 5.2). The use of cubic spline interpolation in this context follows a generic trajectory reconstruction approach, adapted to the limitations of the Apex camera system, namely short gaps and the two dimensional world coordinates context. This method therefore is a good alternative for reconstructing missing data in the outer lane data set, where no overlapping observations are detected.

3.3 Handling overlaps

In the inner lane data set captured by the Apex camera system, consecutive cameras occasionally record the same skater simultaneously, leading to overlapping trajectories, where the same skater is duplicated. These overlaps occur, because the field of view of these cameras intersect, as we have seen in Figure 4. The goal of handling the overlap, is to identify these overlaps and merge them into a single, continuous trajectory, to prevent the duplicated noise being visible in the full trajectory.

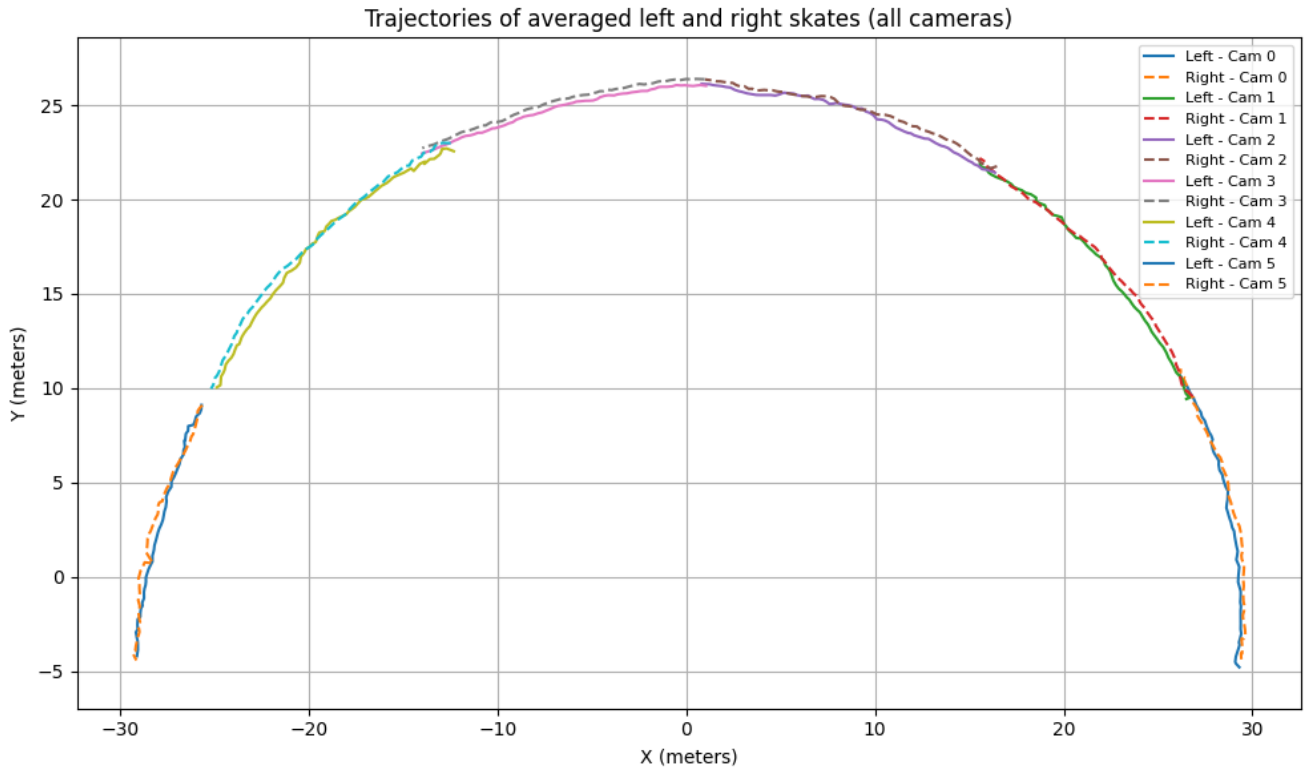


Figure 6: Inner lane of the speed skating arena in Thialf captured by the Apex system.

3.3.1 Overlap Detection

Overlap detection is used to determine whether two consecutive camera segments a skater is observed at the same time or a gap exists between the two segments. This distinction is important, because overlapping segments need a merging strategy, while gaps must be handled using interpolation, see Figure 6. Before any reconstruction is applied, it is therefore first determined if a valid overlap is present between camera i and camera $i + 1$. Overlapping observations occur at the inner lane, where one camera followed by the next camera cover the same region of the track, resulting in duplication. However, this is not always the case. As shown in Figure 6, within a full inner lane trajectory, a transition between cameras can have a gap instead of an overlap. That is why an overlap detection is performed first. If no overlap is detected, the gap between the two segments are interpolated.

When an overlap is detected, it is not immediately clear which observation from camera i should be matched with which observation from camera $i + 1$. This is important, because two camera segments might not start or end at the same frame and the overlapping region can begin earlier in one camera than in the other. To determine the best alignment between overlapping observation, two matching methods are evaluated. These two strategies are defined as follows:

- Head to tail approach: the alignment starts by matching the last observed coordinate of camera i with the closest observed coordinate in the overlapping part of camera $i + 1$. From this match, corresponding observations are paired sequentially by moving backward through the trajectory of camera i and iterating through the overlapping observations of camera $i + 1$ until the first frame of camera $i + 1$.

- Tail to head approach: the alignment starts by matching the first coordinate of camera $i + 1$ with the closest observed coordinate in the overlapping part of camera i . From this match, corresponding observations are paired sequentially by moving forward through the trajectory of camera $i + 1$ and iterating through the overlapping observations of camera i until the last frame of camera i , where no further valid pairs can be formed.

For each direction, this approach pairs points between the two cameras based on the shortest Euclidean distance. Starting from the pair with the smallest distance, the procedure moves sequentially backward and forward through both camera segments to form matching pairs. The direction with the lowest average distance between all paired points is selected as a valid overlap. Using this approach, ensures that the merged segment follows the smoothest possible line between two camera trajectories, to represent the true skating path as closely as possible. Figure 7 shows the sequential pairing by using the euclidean distance before merging. If no overlap is detected, the overlap handler falls back to the interpolation procedure described in Section 3.2, to fill the gaps between segments. This approach prevents data loss and ensures consistent transitions between cameras.

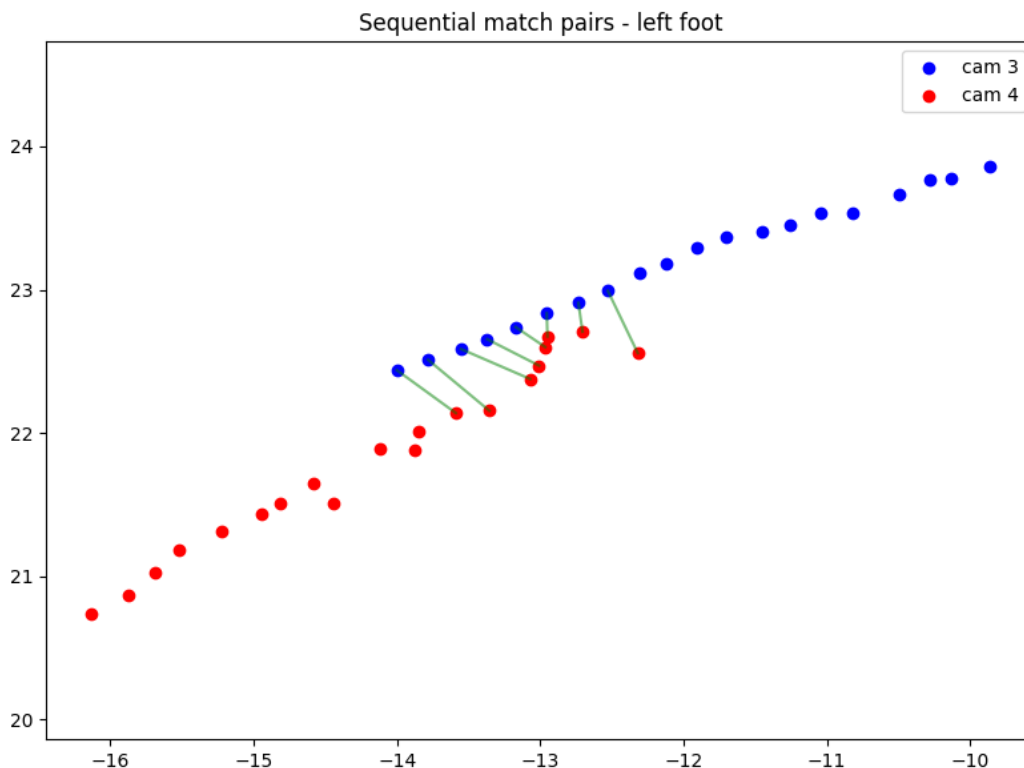


Figure 7: Head to tail/tail to head approach for sequential pairing.

3.3.2 Merging Overlapping Segments

When an overlap is detected between two consecutive camera segments, the same speed skater is observed simultaneously by both cameras. This results in overlapping data points that represent the same observation but might differ slightly due to measurement noise. These overlapping segments must be merged into a single segment. The overlap detection described in Section 3.3.1 yields an

array of matched coordinate pairs. Each pair consists of one observation from camera i and one observation of camera $i + 1$. Let (x_i, y_i) denote the i -th pair of coordinates in the array. The index i iterates over the consecutive matched observations of the array. For each matched pair, the merged coordinates are obtained by averaging the two corresponding observations from the overlapping segments. The merged position is defined as:

$$x_i^{merge} = \frac{x_i^{cam\ i} + x_i^{cam\ i+1}}{2}$$

$$y_i^{merge} = \frac{y_i^{cam\ i} + y_i^{cam\ i+1}}{2}$$

After this merging step, the overlapping camera segments are replaced by the merged coordinates. Additionally, the trajectory obtained can optionally be further processed using smoothing methods, which are described in the next Section.

To assess visual realism of the transition of each segment, as option two additional reconstructions are fitted on top of the merged data points, fitting with linear interpolation and fitting with spline interpolation. These methods are used to visualize and compare how well the transition follows the expected motion through the overlap. See Figure 8 for the result. Their effect on accuracy is later included in Chapter 5.

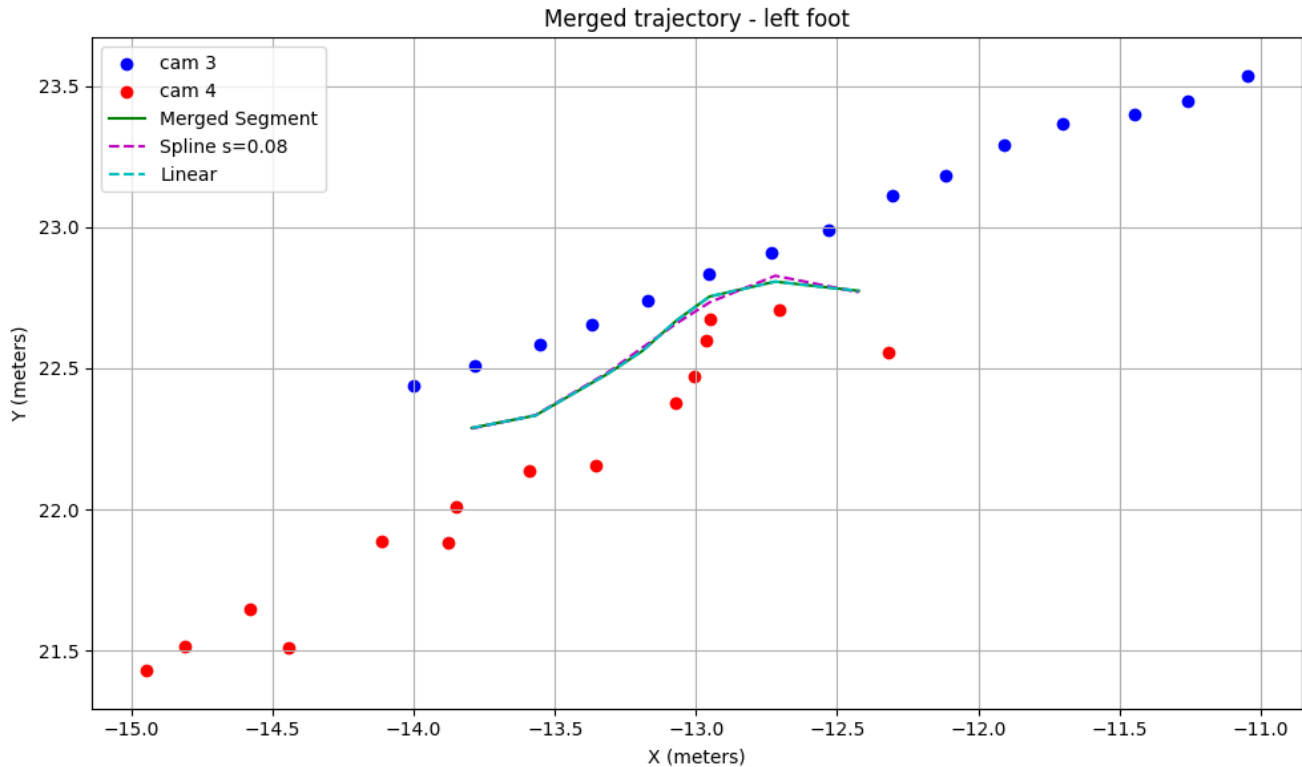


Figure 8: Result after merging and applying linear and spline interpolation.

3.4 Smoothing Methods

3.4.1 Kalman Filter

The Kalman filter used in this research is a smoothing method to improve the stability of the reconstructed inner lane trajectory by reducing noise. After the overlapping segments from two consecutive cameras are merged into a single sequence, the Kalman filter is applied on the trajectory to reduce noise, while keeping the overall motion realistic. The Kalman filter combines the prediction from a constant velocity motion model with the coordinates captured by the Apex multiple camera system. This results in a trajectory that remains close to the original observations but with less noise. The state vector used in this thesis is defined as following:

$$s_t = [x, y, v_x, v_y]^T$$

This contains the position and velocity variables in both horizontal directions.

The Kalman smoother relies on four matrices:

- State transition matrix F : Describes how the state evolves over time. This matrix is fixed.

$$F = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Observation matrix H : This maps the state vector to the observed coordinates by the Apex camera system. This matrix is also fixed.

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

- Process noise Covariance Q : This defines how much variation in motion is allowed between frames.

$$Q = q \begin{bmatrix} \frac{dt^4}{4} & 0 & \frac{dt^3}{2} & 0 \\ 0 & \frac{dt^4}{4} & 0 & \frac{dt^3}{2} \\ \frac{dt^3}{2} & 0 & dt^2 & 0 \\ 0 & \frac{dt^3}{2} & 0 & dt^2 \end{bmatrix}$$

This formulation of the process noise covariance matrix is commonly used in Kalman filtering with a constant velocity model. Here, dt denotes the time interval between two consecutive frames, which is assumed to be constant and is obtained as follows:

$$dt = 1/fps$$

- Measurement noise Covariance R : representing the uncertainty of the observed camera data.

$$R = \begin{bmatrix} R_x & 0 \\ 0 & R_y \end{bmatrix}$$

Here, R_x and R_y denote the variances of the measurement noise in the x -direction and y -direction. They describe the uncertainty of the position measurements captured from the Apex camera system and determine how much the Kalman filter trusts on the observed coordinates in each direction.

Because the state transition matrix (F) and the observation matrix (H) encode the structure of the motion model and the Apex camera measurement process, they all remain constant for all experiments. Only the process noise covariance (Q) and Measurement noise covariance (R) are tuned, as they determine how the filter balances the model prediction with noisy observations.

The Kalman filter is applied over the entire reconstructed inner lane data set, not just the reconstructed overlap region. It produces a continuous path where noise is smoothed, but overall motion is close to the measured data. The final parameter settings used in the experiments for q, R_x, R_y can be seen in Section 5.3.

3.4.2 Gaussian Filter

Gaussian smoothing treats the trajectory as a time series and smooths each coordinate independently by averaging neighboring data points using Gaussian weights. The weighted averaging removes random noise without relying on a physical motion model. The standard deviation σ , which in our case is expressed in number of frames, controls the amount of smoothing:

- Small σ , for example one or two frames, removes high frequency noise but maintains detailed motion.
- A larger σ (larger than four frames) produces a smoother trajectory, but can slightly reduce the sharpness of quick directional adjustments.

σ is applied to both x_t and y_t to keep the shape of the path. The optimal σ is determined by visual inspection and during evaluation using RMSE and MAE (see Section 3.5).

The Gaussian smoothing filter is implemented using the `scipy.ndimage.gaussian_filter1d` function from the `scipy` library and is, like the Kalman filter, applied to the entire observed data set, including the reconstructed data set. At this stage, the trajectory is already continuous, as merging and using Linear and Spline fitting ensure that no overlapping or gaps occurs. Like the Kalman filter, the Gaussian filter smooths the full path to remove noise while maintaining the overall trajectory. This makes it therefore especially suitable for the motion of the speed skater in our case.

3.4.3 implementation and evaluation

Smoothing is a post-processing step in the inner lane of the speed skating arena. After merging the overlapping segments, three variants are produced:

- The raw merged trajectory (averaged matched pairs).
- A linear fit over the merged segment.

- Spline fit over the merged segment

For each of the variants, two fully smoothed versions are generated, one using the Kalman filter and one using the Gaussian filter.

Thus, each trajectory can exist in up to three reconstruction types of trajectories, each with and without smoothing. In Section 3.5, these variants are compared against the manually annotated ground truth data using RMSE and MAE to determine which smoothing improves accuracy after merging and using the interpolation fitting methods.

3.5 Evaluation

The performance of each reconstruction method is evaluated by comparing the reconstructed trajectories with the manually annotated ground truth data set. The goal of this evaluation is to quantify how accurately each method performs to the true skating motion and to assess whether smoothing and fitting lead to measurable improvements compared to the original Apex camera system observations. In this case, two types of data sets are evaluated separately:

- Outer lane, where missing data between cameras are reconstructed using interpolation methods.
- Inner lane, where overlapping/duplicated data between cameras are merged and smoothed.

Each reconstruction variant produces one trajectory per skate, defined by (x_t, y_t) coordinates in world coordinates where t is the frame number thus observation. All trajectories are aligned in time using the global frame numbering of the Apex data set. The evaluation consists of two parts:

- Baseline evaluation: The observed trajectories observed by the Apex camera system are compared with the ground truth data to determine the baseline accuracy of the camera system.
- Reconstructed evaluation: The reconstructed trajectories are compared with the same ground truth.

The difference in errors between these two evaluations indicate whether the applied reconstruction methods actually provide improvement.

Unfortunately, no manually annotated ground truth data are available for the outer lane data set. This set is therefore evaluated by human visual inspection for trajectory continuity, smoothness and realism. RMSE and MAE are only used for evaluation in the inner lane data set, where a manually annotated ground truth data set is available.

3.5.1 Metrics and Variants

As already mentioned, two well known metrics are used to measure the accuracy: RMSE and MAE. These metrics are defined as follows:

$$RMSE = \sqrt{\frac{\sum_{t=1}^N (x_t - x_t^{ground})^2 + (y_t - y_t^{ground})^2}{N}}$$

$$MAE = \frac{\sum_{t=1}^N (|x_t - x_t^{ground}| + |y_t - y_t^{ground}|)}{N}$$

Where x_t and y_t are the reconstructed coordinates, x_t^{ground} and y_t^{ground} are the ground truth coordinates and N is the number of frames, thus observations. The characteristics of these evaluation functions can be explained as follows:

- The RMSE penalizes large deviations more strongly, because the squared differences dominate the sum.
- MAE gives the absolute error per frame and is less affected by outliers.

Both metrics are expressed in meters. For each set, a reconstruction method is evaluated, the following variants are evaluated:

- Outer Lane (interpolation):
 - Linear Interpolation
 - Spline Interpolation
- Inner lane (overlap):
 - Raw merged trajectory
 - Linear fit on merged segment
 - Spline fit on merged segment

Each of the above is evaluated both without smoothing and with either the Kalman filter or the Gaussian filter.

3.5.2 Interpretation

Lower values of the RMSE and MAE indicate that the reconstruction reproduces the ground truth motion accurately. If smoothing significantly reduces these errors compared to the unsmoothed versions, it suggests that these smoothers, in post-processing, improve overall data quality and can reproduce a realistic trajectory with less noise. However, if the differences are small, the Apex system may already provide accurate positional data, making additional reconstruction steps less effective. A detailed analysis of these findings is presented in Chapter 4, where results and visual comparisons are discussed.

4 Experiment Setup

In this chapter the data sets and experimental setup used to evaluate the trajectory reconstruction pipeline presented in Chapter 3 are described. All experiments are based on data collected in the speed skating arena Thialf, captured by the Apex multiple camera system. Two data sets are used:

- Outer lane data set containing gaps between camera transitions.
- Inner lane data set containing overlapping/duplicated observations at each camera transition.

Each data set has a reconstruction challenge, where the inner lane needs a overlap correction and the outer lane interpolation, where these are processed separately but within the same evaluation framework.

4.1 Dataset

The data set was collected in the speed skating arena Thialf, using the Apex multiple camera system, which consists of six synchronized cameras. Each camera covers a different section of the track. The cameras record the motion of an individual skater and export two-dimensional coordinates (x, y) in world coordinates. A single frame corresponds to one time step for a specific camera. For each frame, the data set provides the following:

- Coordinates for the left foot skate and right foot skate.
- Front and back coordinates of each skate.
- Annotated ground truth data, manually labeled and used for evaluation, representing true skater positions independent of the Apex multi-camera system observations.

All coordinates are expressed in the world coordinate system after calibrations and image-to-world transformation. The calibration and transformation procedure is described in Chapter 2 and summarized again in Section 3.1.

To make the trajectories more stable and easier to visualize, the average coordinates of the front and back skates are calculated as `left_skate_avg` and `right_skate_avg`. The midpoint gives a single representative position per skate. This results in a smoother and more consistent visual reference for analysis. These transformations extend the data set with additional new columns.

In addition to the two data sets, manually annotated ground truth trajectories are available for the inner lane data set and are used for evaluation. For the outer lane data set, such annotations are not provided, and therefore evaluation through visualizations comparisons (e.g. speed plots) is performed. Each subset is organized into separate folder containing CSV files per camera for each skate.

4.2 Preprocessing and Aligning

This section describes the data preparation and alignment steps that take place before the reconstruction process. The preprocessing phase (Sections 4.2.1 - 4.2.2) prepares the raw Apex

data for reconstruction by loading, normalizing and stitching camera segments. The post reconstruction phase (Section 4.2.3) aligns trajectories, corrects small inconsistencies and prepares the reconstructed data for evaluation.

4.2.1 Segment Stitching

Consecutive camera segments are joined using the function `stitch_segments()`. This function concatenates the trajectories using their frame indices and keeps track of the boundaries between camera recordings. For the outer lane data set, stitching identifies gaps where frames are missing and records their size for later interpolation, see Section 3.2. For the inner lane data set, stitching identifies overlap regions where two camera segments record the same motion simultaneously, see Section 3.3. The output is one trajectory per skate and per data set, containing either gaps (see Figure 5) or overlapping regions (see Figure 6).

4.2.2 Overlap Handling and Average Merge

In the inner lane, overlapping segments between camera i and camera $i + 1$ are detected by using nearest neighbor matching between the tail of camera i and the head of camera in $i + 1$ (see Section 3.3.1 for the detection logic). The same averaging logic is applied to the ground truth data set so that both the observed data and the manually annotated ground truth share the same temporal and spatial structure.

4.2.3 Length Alignment

Small mismatches in the number of frames occasionally occur after stitching and merging due to rounding or synchronization differences. Before evaluation, a short alignment procedure on the reconstructed data set is applied, ensuring that both the produced and ground truth trajectories have identical lengths. This is determined as follows:

- If the predicted trajectory is one or more frames shorter than the ground truth, one or more additional frames are linearly interpolated in the predicted trajectory.
- If the predicted trajectory is one or more frames longer than the ground truth, the final frame is removed from the predicted trajectory.

In practice, the length difference between the predicted trajectory and the annotated ground truth data has a maximum of one frame. Therefore, only the last frame is removed.

This procedure guarantees that the RMSE and MAE are calculated on synchronized coordinates, ensuring accurate evaluation. Fortunately, after determining the number of missing frames or additional frames, we observe that the difference is always either zero or one. Although the implementation allows for one or more frames as a fall back, in practice only one frame is adjusted.

4.3 Experimental Conditions

Experiments are performed separately for the outer lane and inner lane data sets.

- Outer lane: linear and spline interpolation are applied to fill the missing data between camera segments. Because no ground truth data annotations exist, these trajectories are evaluated visually by humans. The reconstructed line is plotted together with the raw observed Apex camera observations to assess the continuity and plausibility of the interpolated path.
- The inner lane: the merged trajectories form three base variants:
 - Raw Merged
 - Linear fit
 - Spline fit

Each base variant is also processed with two optional smoothing methods, The Kalman filter and the Gaussian filter, both are applied over the entire reconstructed trajectory.

The combination of these base variants and smoothing methods results in a total of nine evaluated variants.

4.4 Execution and Outputs

All experiments are performed by automated scripts that iterate over both outer and inner lane data sets and both skates. Each run performs loading, preprocessing and reconstruction and, when ground truth is available, the RMSE and MAE evaluation are performed as well. The resulting trajectories are saved as CSV files containing the reconstructed (x, y) coordinates and visual plots are generated for each camera transition, as well as the full trajectory views. Before evaluation, the following checks are ensured:

- Equal lengths of both predictions and ground truth trajectories, if not, frame(s) are added or removed on the prediction data set.
- Missing values: frames with missing ground truth annotations are excluded from the evaluation.
- Continuity at camera boundaries.

Figures illustrating typical interpolation gaps and overlap merges are presented in Chapter 5, together with the RMSE and MAE tables and trajectory comparisons.

5 Results

In this section the results of the trajectory reconstruction experiments described in Chapter 4 are presented. The results are divided into two parts corresponding to the two data sets: the outer lane data set, which focuses on the results of interpolations methods and the inner lane data set, which focuses on the overlap handling and smoothing. For the inner lane reconstruction, both visualizations and accuracy metrics are reported using the available annotated ground truth. For the outer lane data set, no ground truth annotated is available. Therefore, these results are evaluated by visual inspection to assess their continuity. Although this method is subjective, it provides a necessary means to verify if the interpolated paths between camera segments follow a realistic motion without visible noise or discontinuities.

5.1 Overview of Results

The goal of these experiments is to evaluate how well different reconstruction methods restore continuous and realistic trajectories of speed skaters captured by the Apex multiple camera system. For the outer lane data set, where no annotated ground truth is available, the quality of the reconstruction is visually assessed by comparing the raw observed trajectories with the interpolated ones. For the inner lane data set, where annotated ground truth is available, reconstruction is evaluated using RMSE and MAE, as described in Section 3.1.5, together with visual comparisons of the reconstructed and annotated trajectories.

5.2 Interpolation Results

The outer lane data set contains small temporal gaps at each segment, which means gaps in each transition between consecutive cameras. Linear and spline interpolation were applied to fill these missing sections. Figures 9 and 10 show representative examples comparing the observations with the interpolated predictions.

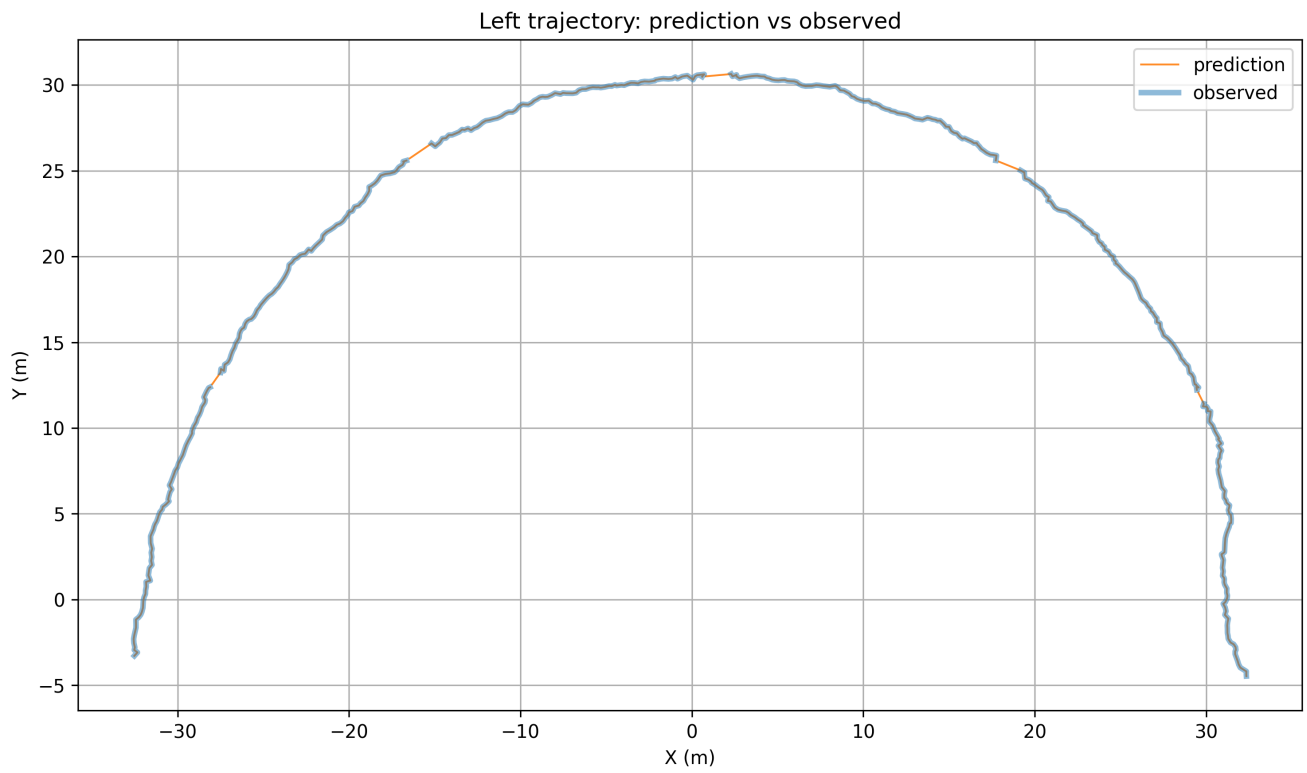


Figure 9: Linear Interpolation result of the outer lane trajectory.

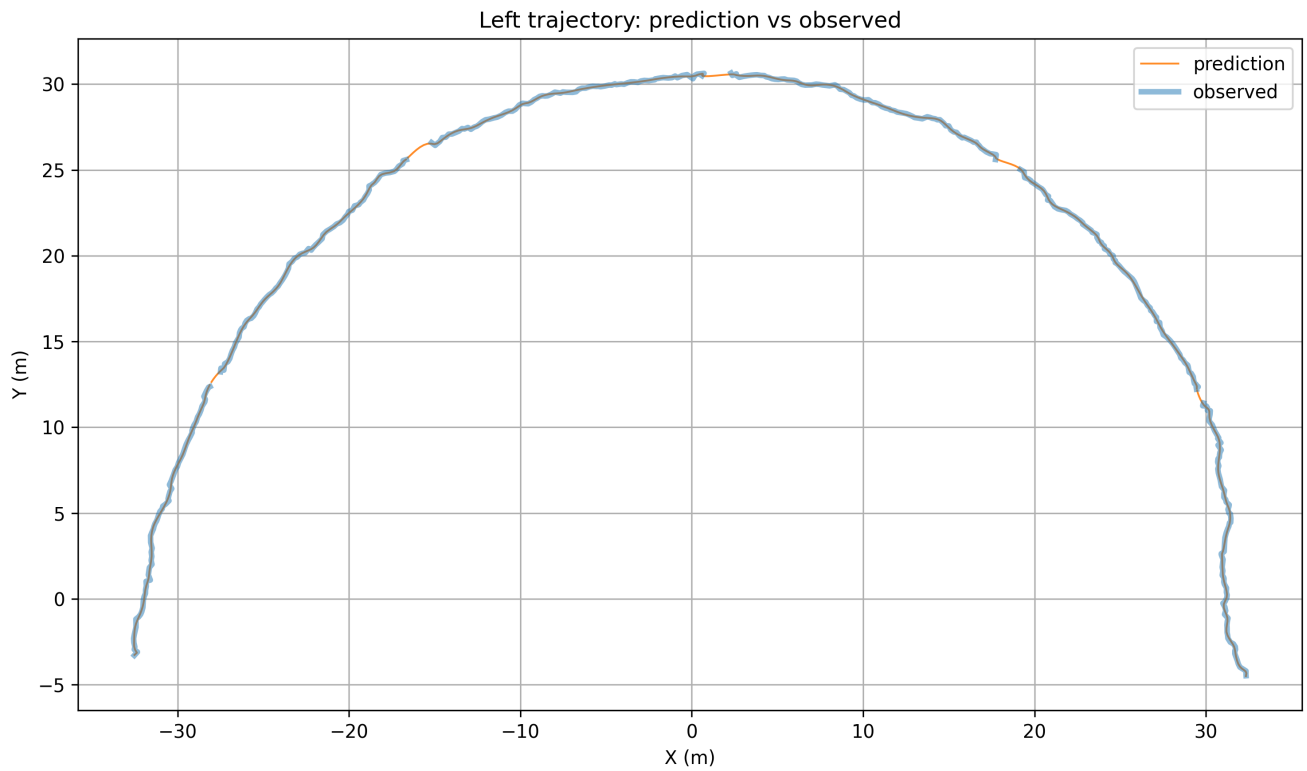


Figure 10: Spline Interpolation result of the outer lane trajectory.

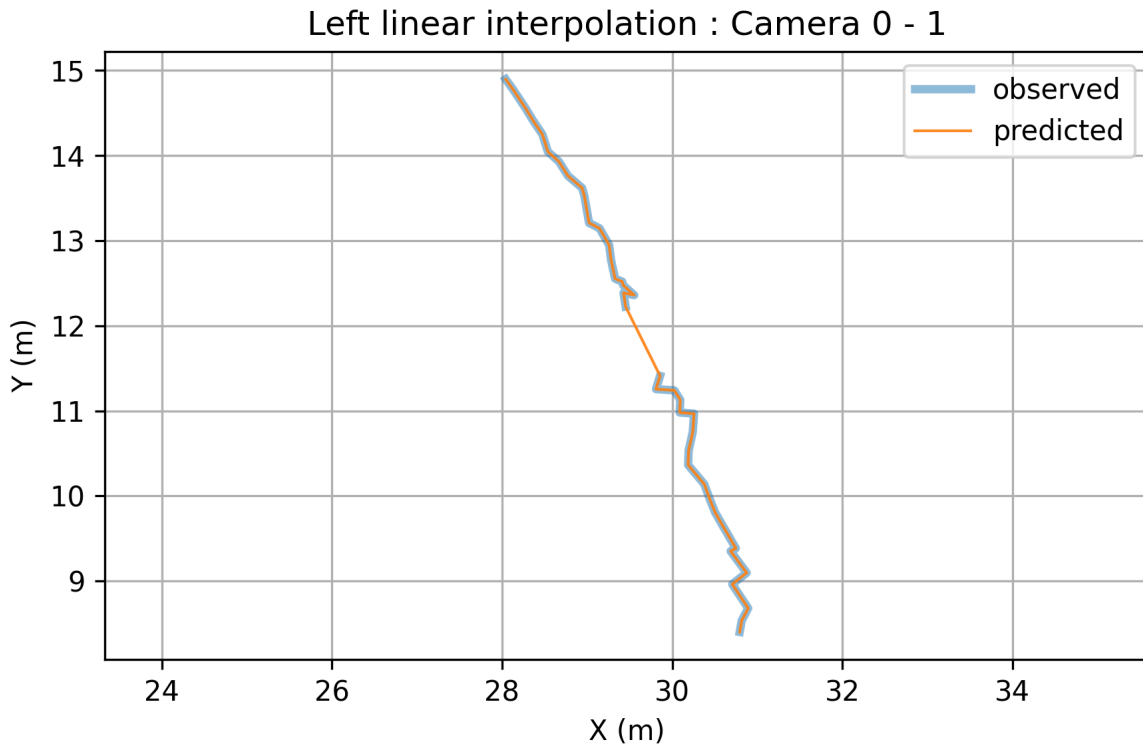


Figure 11: **Linear interpolation** result for the first outer lane trajectory segment (camera 1 to camera 2).

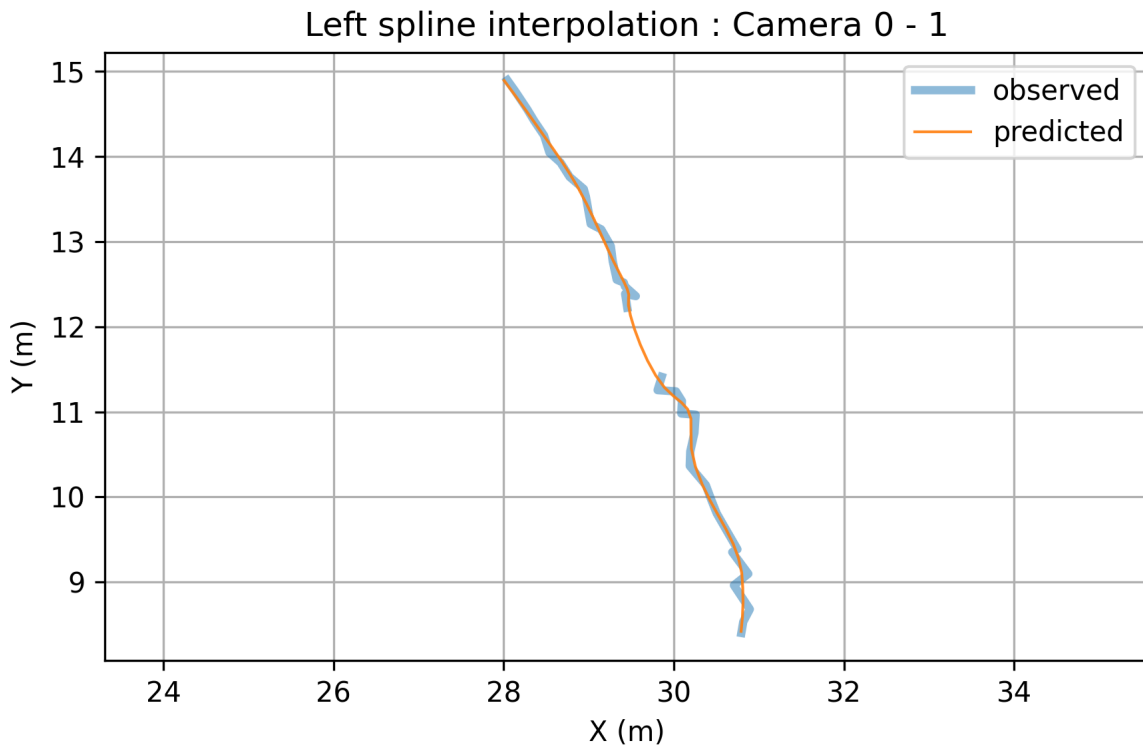


Figure 12: **Spline interpolation** result for the first outer lane trajectory segment (camera 1 to camera 2).

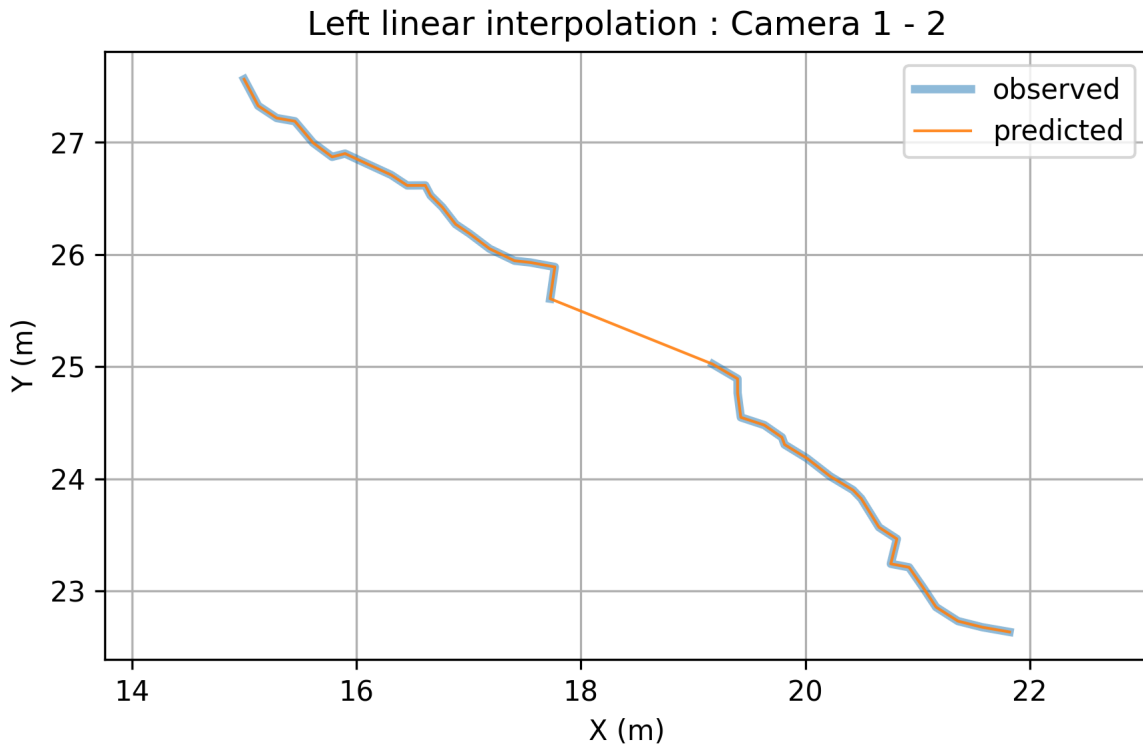


Figure 13: **Linear interpolation** result for the second outer lane trajectory segment (camera 2 to camera 3).

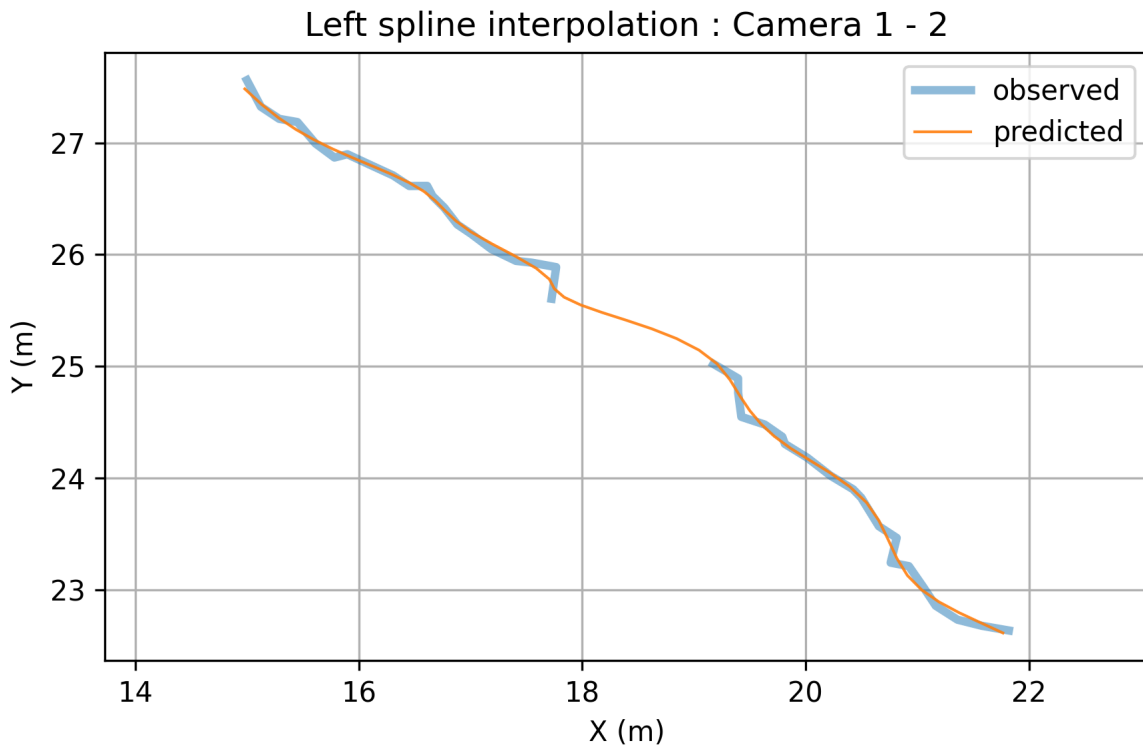


Figure 14: **Spline interpolation** result for the second outer lane trajectory segment (camera 2 to camera 3).

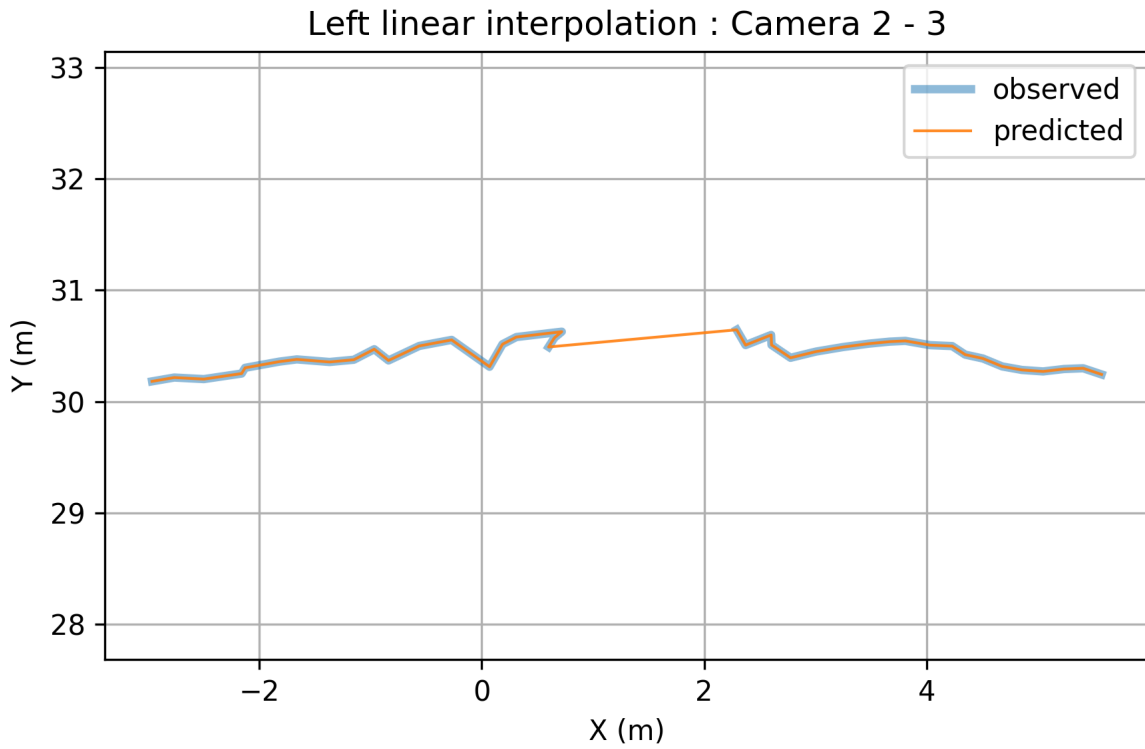


Figure 15: **Linear interpolation** result for the third outer lane trajectory segment (camera 3 to camera 4).

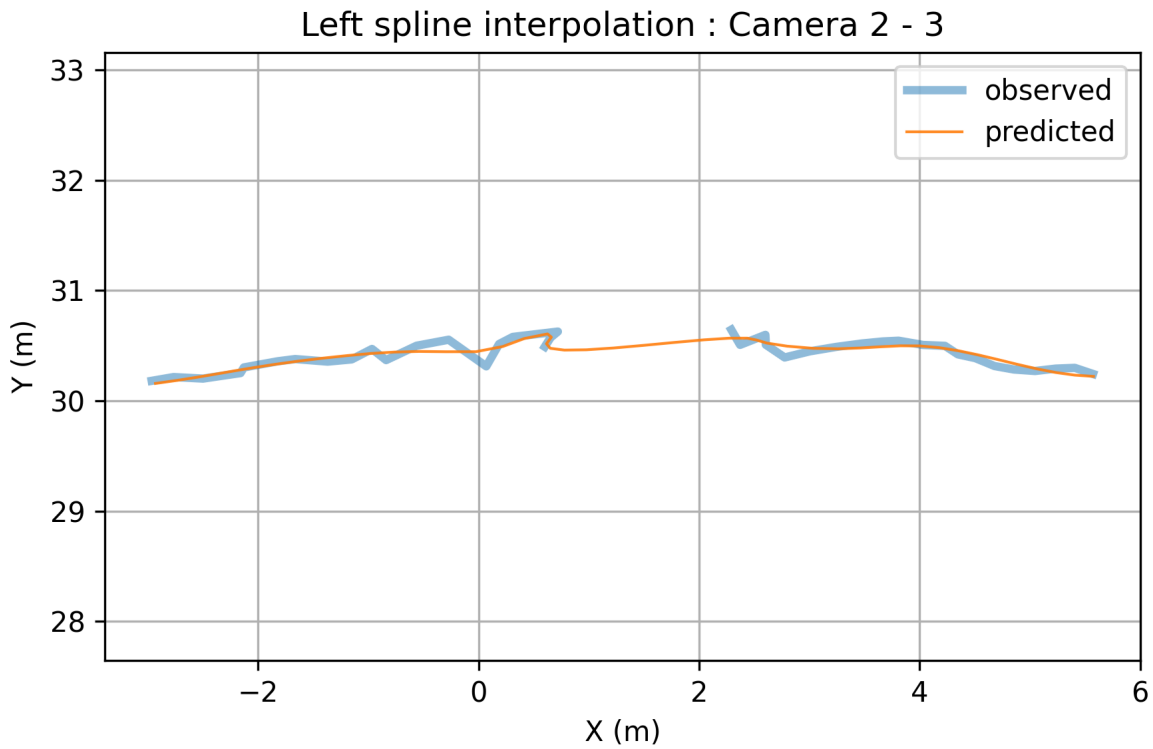


Figure 16: **Spline interpolation** result for the third outer lane trajectory segment (camera 3 to camera 4).

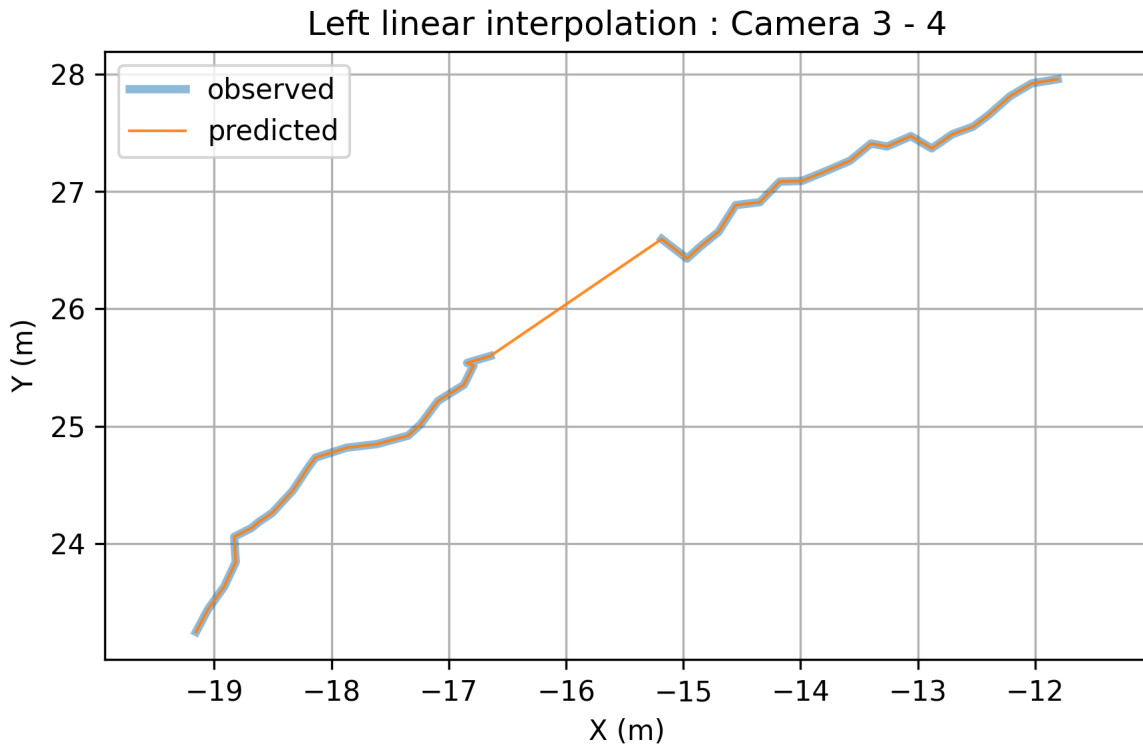


Figure 17: **Linear interpolation** result for the fourth outer lane trajectory segment (camera 4 to camera 5).

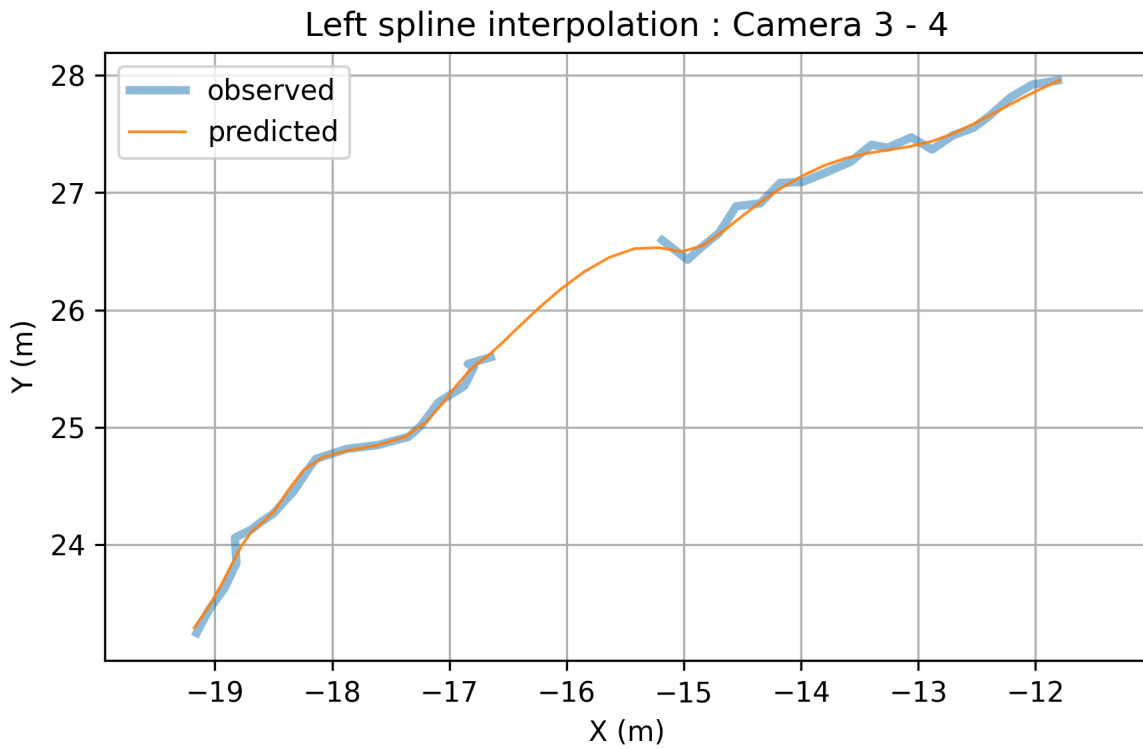


Figure 18: **Spline interpolation** result for the fourth outer lane trajectory segment (camera 4 to camera 5).

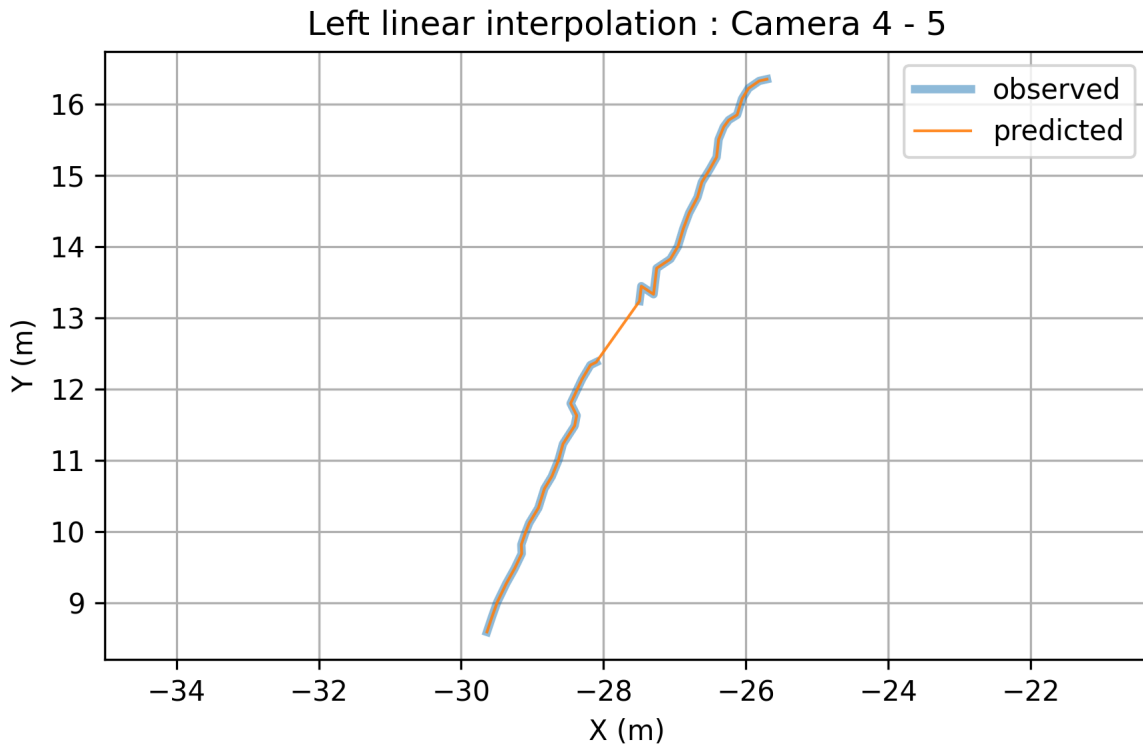


Figure 19: **Linear interpolation** result for the fifth outer lane trajectory segment (camera 5 to camera 6).

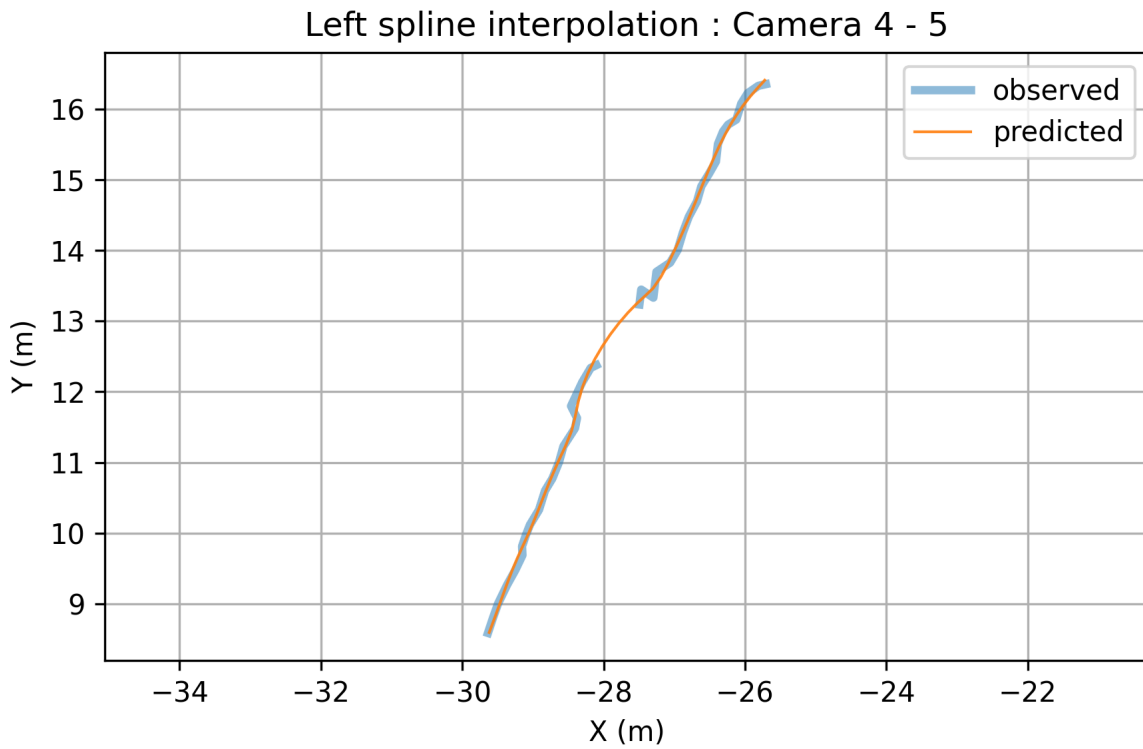


Figure 20: **Spline interpolation** result for the fifth outer lane trajectory segment (camera 5 to camera 6).

Visually, linear interpolation restores continuity between camera zones while maintaining a consistent motion direction. However, linear interpolation does not take noise into account. As shown in Figure 9, the orange line still contains a significant amount of noise. The spline interpolation produces a slightly smoother trajectory near the gap regions and follows the motion trend naturally and is more realistic with less noise. Because both methods rely only on neighboring data points, small deviations may appear in longer gaps. No quantitative evaluation was performed for this outer lane data set. Therefore, visual assessment confirms that both methods successfully connect each transition and maintain physically plausible trajectories.

In addition to trajectory continuity, the effect of interpolation on the estimated skating speed was visually analyzed at the transitions between camera segments. These segments consist of short gaps, where the number of missing frames varies and ranges from only a few frames to approximately ten frames. The exact number of missing frames depends on the distance between the last camera observation i and the first camera observation $i + 1$.

Figures 23 and 22 show the result of the speed over the trajectory after interpolation. Although the speed graph (see Figures 23 and 22) still contains noise, the values remain within a plausible range. No abrupt acceleration or deceleration is introduced at the transition boundaries, indicating that both linear and spline interpolation preserve the overall motion of the skater.

Compared to the raw speed predictions captured by the Apex camera system shown in Figure 21, the interpolated results show a clear improvement in stability. In the raw data set, speed peaks occur at each camera transition due to missing frames, resulting in unrealistic high values. These peaks are a direct consequence of the time gaps between consecutive camera observations. After interpolation, these discontinuities are effectively removed, resulting in a smoother and more consistent speed graph within each segment.

Note that while the transitions at each segment are now smooth, the overall speed graph still has high frequency oscillations. This noise originates from the raw observations of the Apex camera system themselves. However, the focus of this analysis is on the behavior at camera transitions rather than the consistency of the full speed over the trajectory.

Looking at Figures 22 and 23, a difference can be observed between the interpolation methods in terms of speed stability. The speed profile obtained after linear interpolation looks more volatile compared to the results obtained with spline interpolation. This behavior is expected, because linear interpolation connects consecutive points with a straight line segments. As a result, the direction of motion changes abruptly at the boundary of each segment. When speed is estimated from these piecewise linear trajectories, these abrupt directional changes become visible as unstable behavior in the estimates speed. In contrast, spline interpolation enforces smoother transitions by ensuring continuity in both the first (velocity) and second (acceleration) derivatives of the reconstructed trajectory. As a result, the corresponding speed profile is visually smoother and more consistent between camera transitions. Although rapid variations remain due to noise in the observations of the Apex camera system, spline interpolation clearly reduces the instability visible in Figure 22.

Although no quantitative evaluation is available for the outer lane data set, the speed graph (see

Figure 23) confirm that the interpolated trajectories successfully bridge the missing frames without introducing unrealistic accelerations. This supports the visual evaluation that the reconstructed trajectories remain realistic and physically consistent with the expected skating velocities.

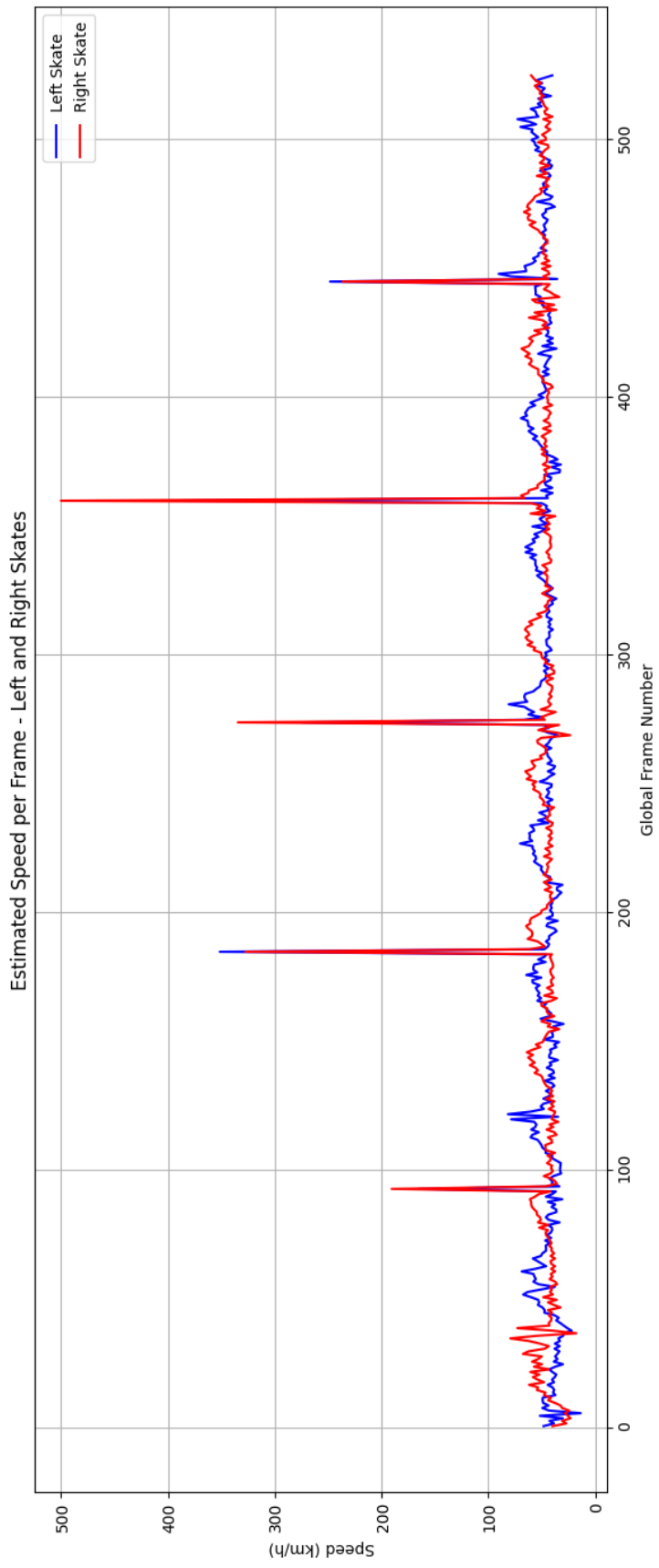


Figure 21: Skating speed per frame captured by the Apex camera system.

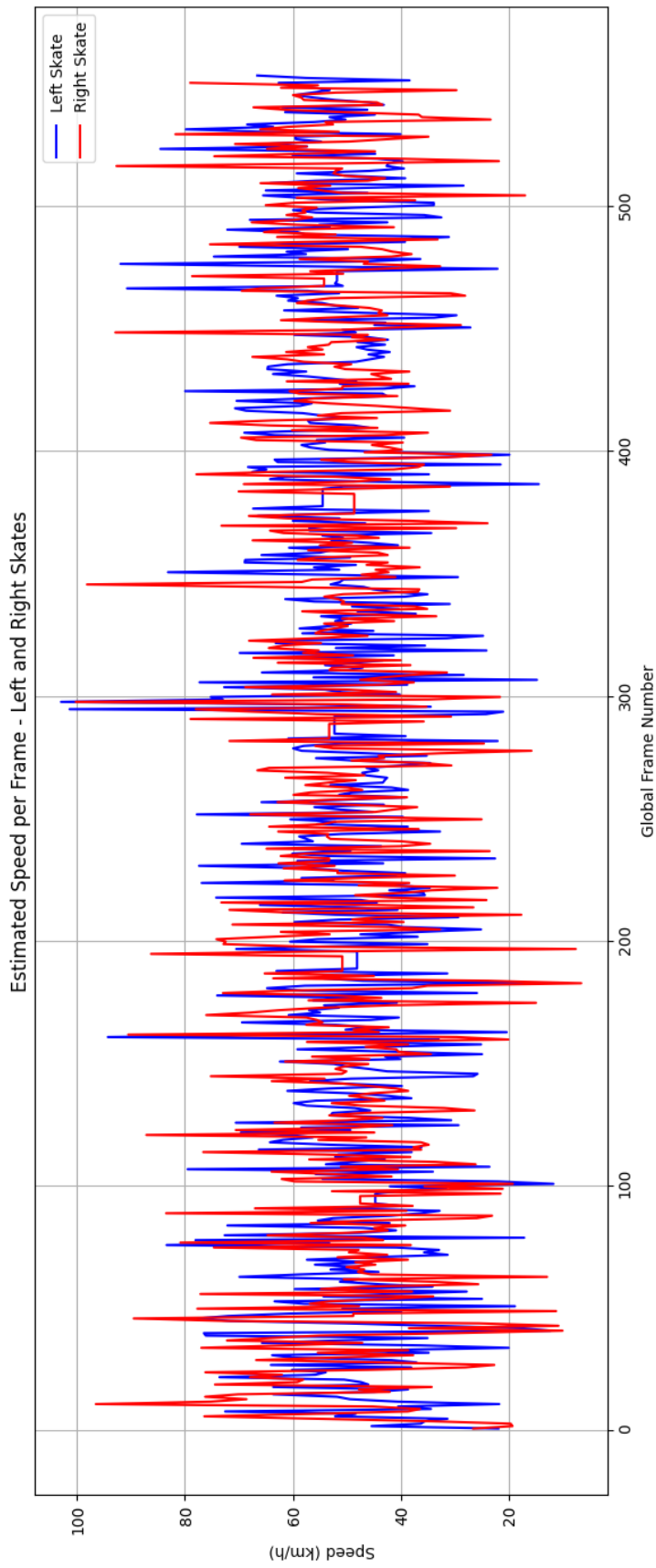


Figure 22: Predicted skating speed per frame after cubic **linear** interpolation.

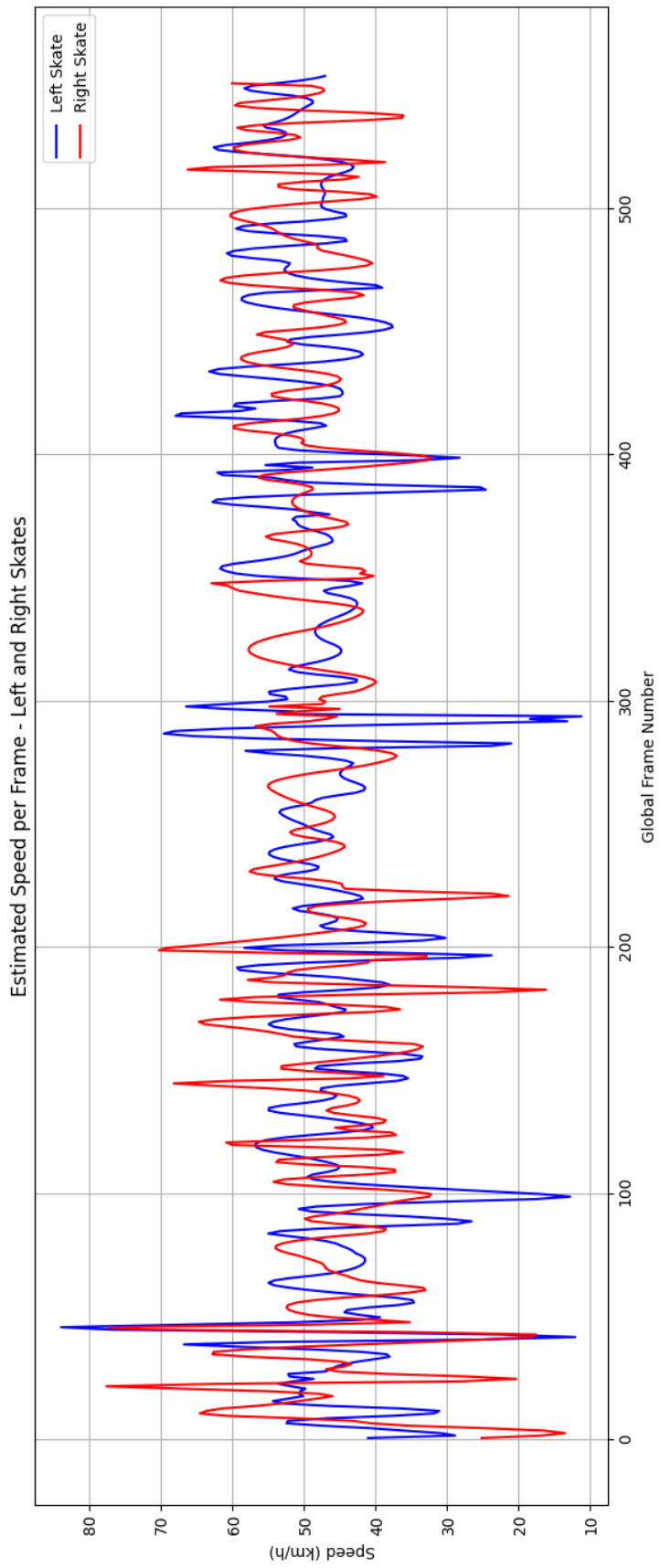


Figure 23: Predicted skating speed per frame after cubic **spline** interpolation.

5.3 Parameter Selection for Overlap

To identify the most suitable parameters for the smoothing methods applied to the overlap reconstruction, a grid search was performed for the spline interpolation smoother, Gaussian filter and Kalman filter methods. Each combination was applied to the reconstructed trajectories and evaluated using the RMSE and MAE against the ground truth.

5.3.1 Spline Parameter Tuning

The smoothing parameter s controls how closely the spline follows the data points. When s is small, the spline follows the data closely, which can cause overfitting to small measurements noise. When s is large, the trajectory becomes smoother, but it may ignore small details of the motion. A grid search on $s \in \{0.5, 1, 2, 5\}$ showed that between 0.5 and 1 the best results were achieved. After finetuning, $s = 0.8$ was selected, resulting in stable and realistic trajectories without overfitting. To obtain this value, an additional sub grid search was performed within the interval $s \in [0.5, 1.0]$, where intermediate values were evaluated using RMSE and MAE against the annotated ground truth in the inner lane.

5.3.2 Gaussian Filter Parameter Tuning

A grid search over the standard deviation (σ) and truncate parameters identified the best performing Gaussian parameters. The window length follows:

$$L = 2 \times \sigma \times \text{truncate} + 1$$

The combination $\sigma = 3$ and $\text{truncate} = 2$ achieved the lowest RMSE (0.129) and provided an optimal trade off between noise reduction and responsiveness. Larger windows resulted in minor gains but increased lag and edge distortion. Therefore, $\sigma = 3$ and $\text{truncate} = 2$ was used.

σ	truncate	RMSE (meters)	MAE (m)
2	3	0.154	0.083
3	2	0.129	0.081
3	3	0.127	0.082

Table 1: Gaussian filter grid search results.

5.3.3 Kalman Filter Parameter Tuning

The Kalman filter used a constant velocity (CV) model with covariance matrices:

$$Q = q \begin{bmatrix} \frac{dt^4}{4} & 0 & \frac{dt^3}{2} & 0 \\ 0 & \frac{dt^4}{4} & 0 & \frac{dt^3}{2} \\ \frac{dt^3}{2} & 0 & dt^2 & 0 \\ 0 & \frac{dt^3}{2} & 0 & dt^2 \end{bmatrix}, \quad R = \begin{bmatrix} R_x & 0 \\ 0 & R_y \end{bmatrix}$$

A grid search over $q = \{1, 2, 4, 6\}$ and $R = \{0.0004, 0.0005, 0.0007, 0.003\}$ showed that $q = 4$ and $R = 0.0004$ provided the best results with a low RMSE and MAE. Smaller R values would produce unrealistic overconfidence, while higher q values would increase noise.

Parameter	Value
Frame rate (fps)	68.48
Time interval (dt)	0.0146 s
Process noise (q)	4
Measurement noise (R_x, R_y)	0.0004
Model	Constant Velocity (CV)
Filter type	RTS smoother

Table 2: Kalman filter final parameters

5.4 Overlap Results

The inner lane data set contains overlapping trajectories between consecutive cameras. After merging, three reconstruction variants were constructed:

- The raw merged trajectories, which are the averaged matched pairs.
- Linear fit on the merged region.
- Spline fit on the merged region.

Each base variant was also smoothed with either the Kalman filter or the Gaussian filter, resulting in six additional versions with smoothing.

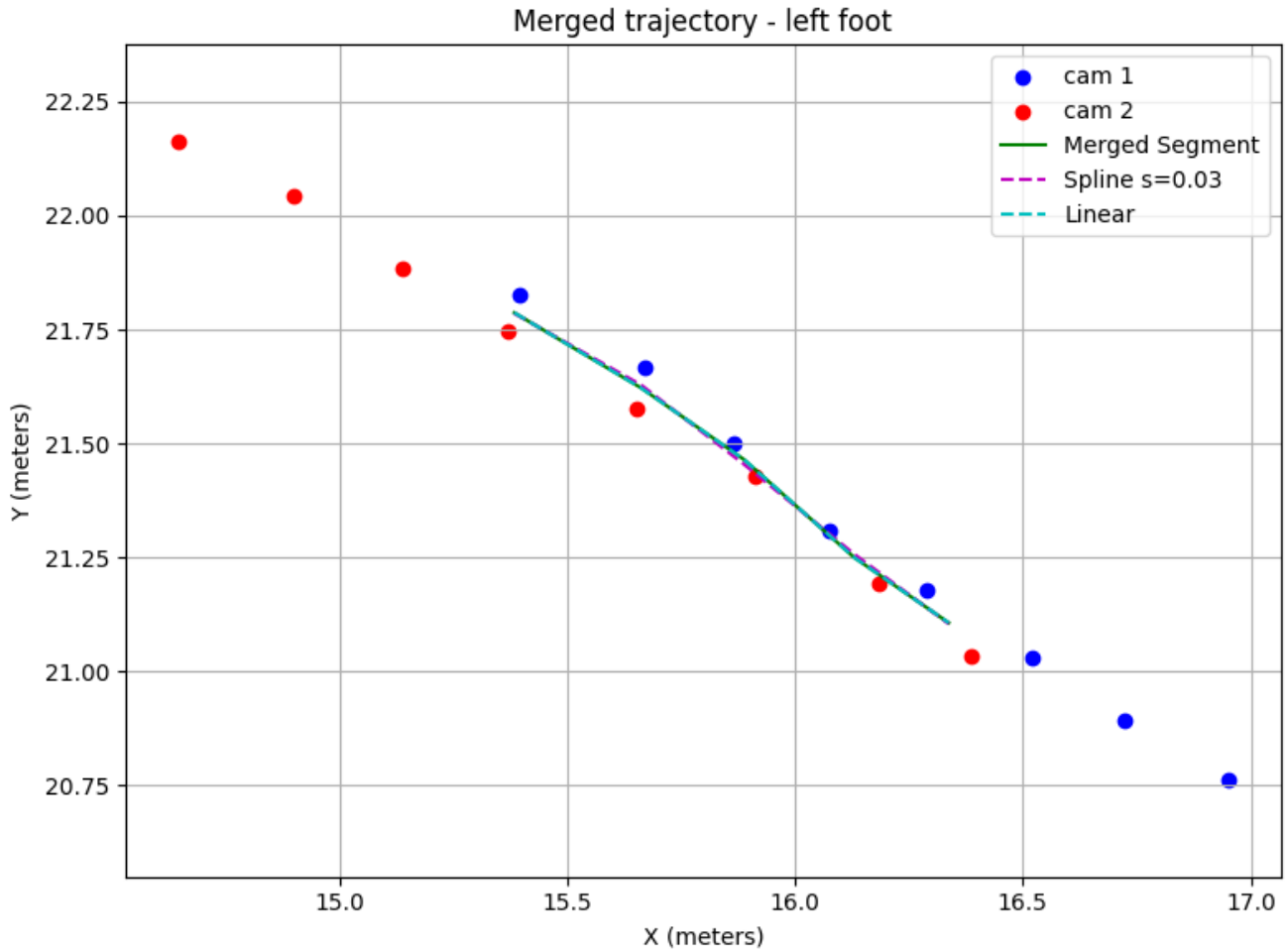


Figure 24: Overlap merging left skate.

Figure 24 shows the three reconstruction variants applied in the merging step. First, the averaged merged segment (green) represents the baseline trajectory. Then the linear fit (cyan) provides a minimal adjustment for continuity and finally, spline fit (magenta) results in a slightly smoother curve while maintaining the overall motion shape.

5.4.1 RMSE and MAE

Table 3 shows the average RMSE and MAE values for each reconstruction variant. Each reconstruction is computed against the manually annotated ground truth trajectories.

Method	Smoothing	RMSE (m)	MAE (m)
Raw merged	None	0.0554	0.0477
Linear fit	None	0.0554	0.0477
Spline fit	None	0.0554	0.0477
Raw merged	Kalman	0.1197	0.0951
Linear fit	Kalman	0.1197	0.0951
Spline fit	Kalman	0.1197	0.0951
Raw merged	Gaussian	0.0738	0.0601
Linear fit	Gaussian	0.0738	0.0601
Spline fit	Gaussian	0.0737	0.06

Table 3: Average RMSE and MAE for the inner lane reconstruction variants for the left skate.

The results of the experiments in Table 3 show that all reconstruction methods achieve very low error values, indicating that the observations of the Apex multiple camera system are already of high positional accuracy. The RMSE and MAE are virtually identical across the raw merged, linear and spline fits, indicating that the overlap regions are too short for the fitting methods to have a measurable impact. The merging step itself already constructs a linear transition between cameras. The Kalman and Gaussian filters increase the RMSE by a factor of two. This is expected because the Apex system observations already have low error, so smoothing modifies accurate data rather than correcting it. The reason is that the baseline RMSE of the Apex multiple camera system observations is already low. This makes it natural that smoothing produces a minimal higher error. The purpose of the smoothing step is, therefore not only to improve the RMSE and MAE but also to remove noise and ensure visually continuous motion. This can also be observed in the raw versus ground truth comparison, see Figure 25, where the raw observed data from the Apex system already align closely with the manually annotated trajectory and show minimal noise. Thus, while linear and spline fitting are negligible for overlap reconstruction, smoothing methods do not improve the quantitative accuracy measured by RMSE and MAE. Using the baseline averaging of matched overlap pairs, obtained from the observed segments, as a reference, smoothing increases the error by a factor of two compared to the raw merged trajectory. Smoothing is therefore not suitable for improving reconstruction accuracy, but can still be useful for visualization purposes by reducing noise.

5.4.2 Visual Results

Figures 25 - 27 show the reconstructed overlap regions together with the corresponding ground truth.

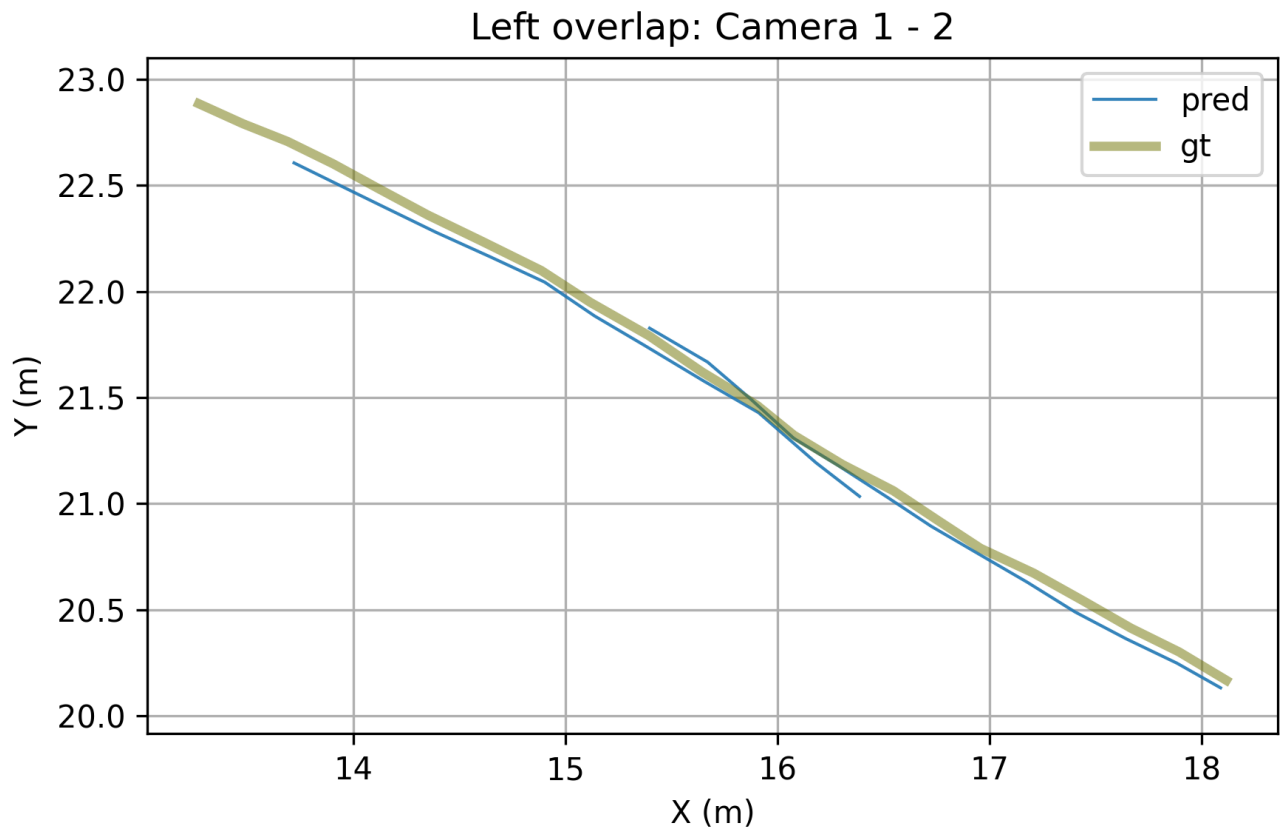


Figure 25: Baseline overlap reconstruction of the left skate, Apex observations(pred) vs. ground truth.

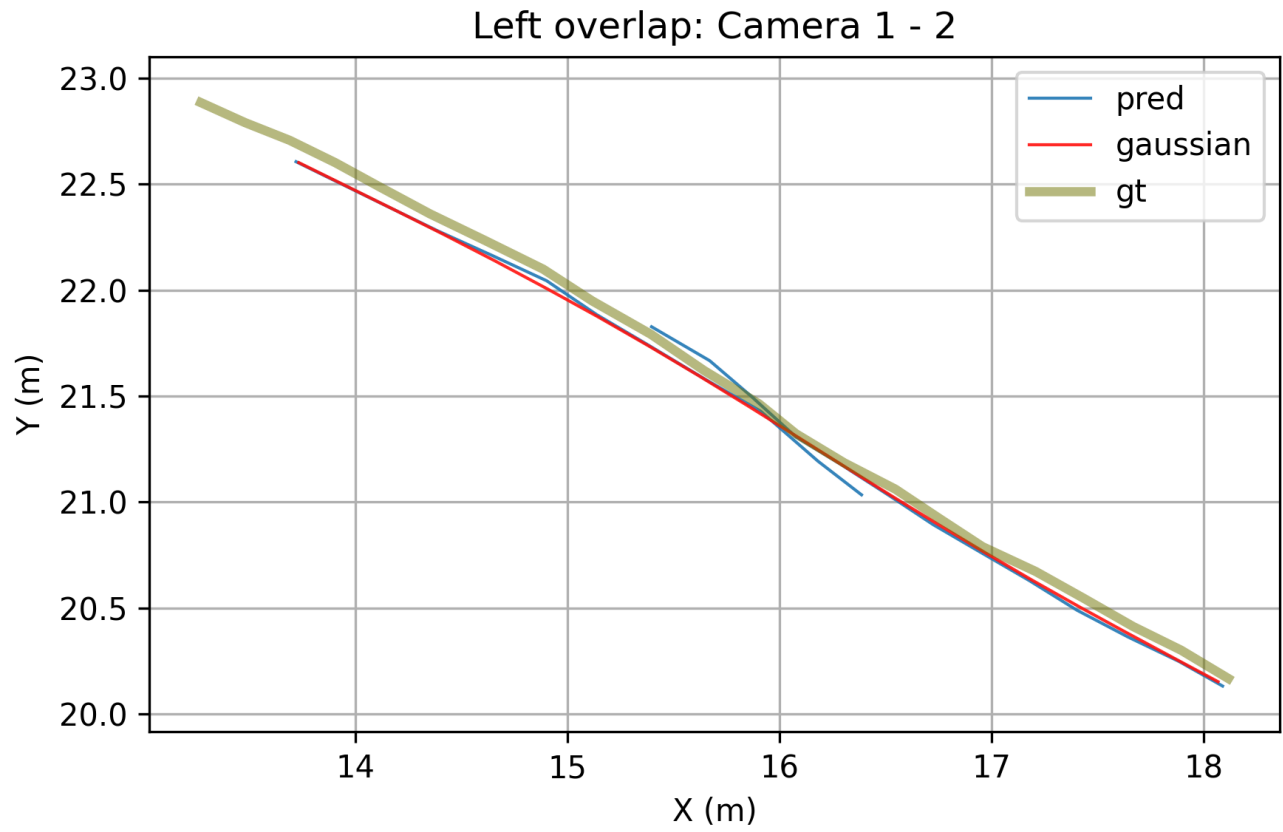


Figure 26: Gaussian smoothed overlap of the left skate, Gaussian filter + Apex observations(pred) vs. ground truth.

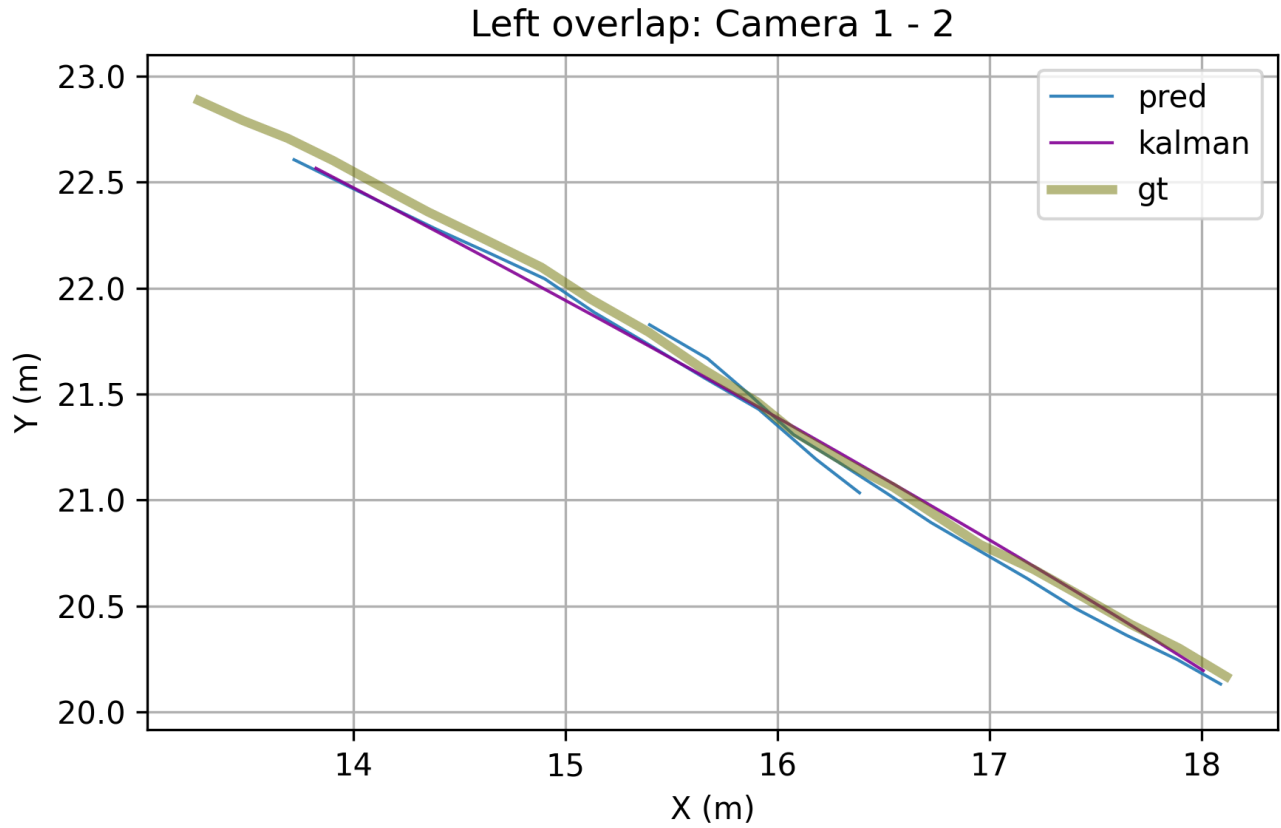


Figure 27: Kalman smoothed overlap of the left skate, Kalman + Apex observations(pred) vs. ground truth.

The smoothed version closely follows the ground truth, resulting in the removal of small noise while preserving the overall trajectory. The Kalman filter provides the smoothest visual trajectory, while the Gaussian filter maintains the sharper local features. To evaluate whether these smoothing effects hold across the inner lane trajectory, Figures 28 to 30 show the smoothed and unsmoothed trajectories over the full inner lane trajectory.

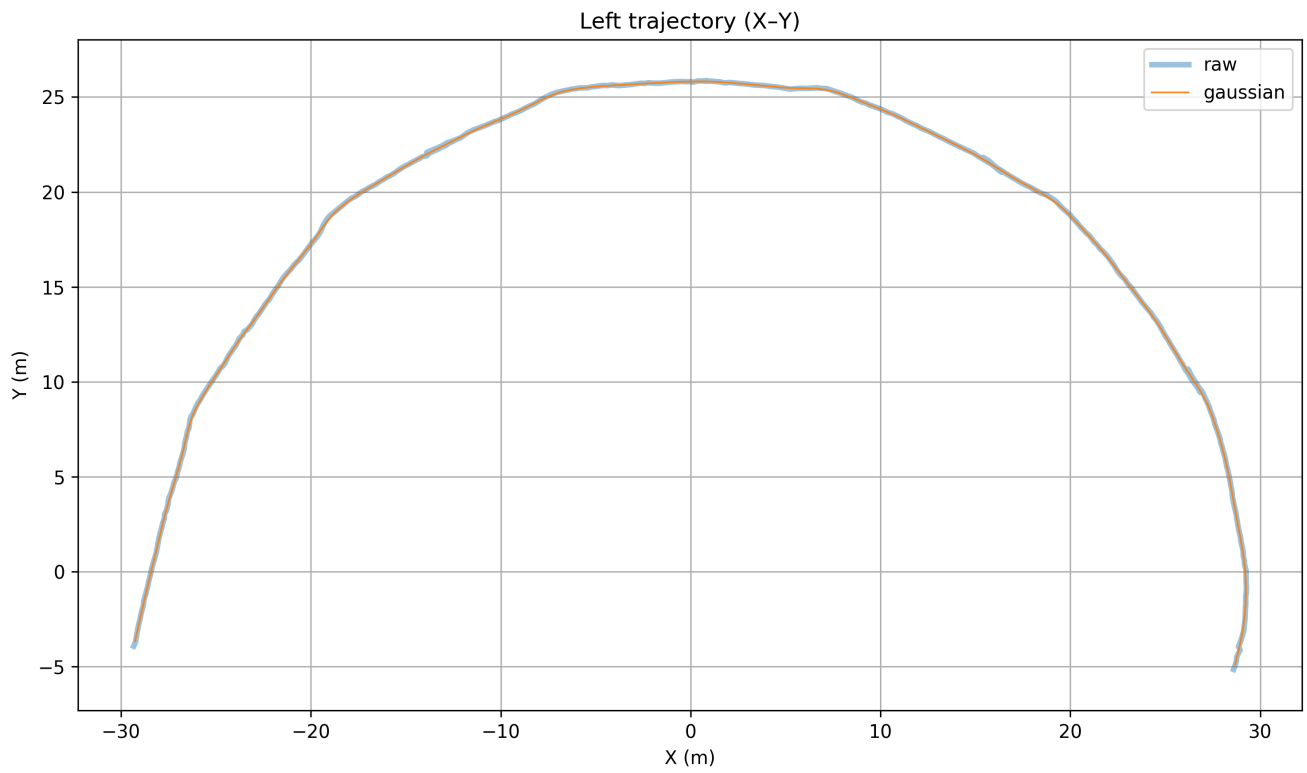


Figure 28: Full trajectory, Gaussian smoothing vs. Apex observed data.

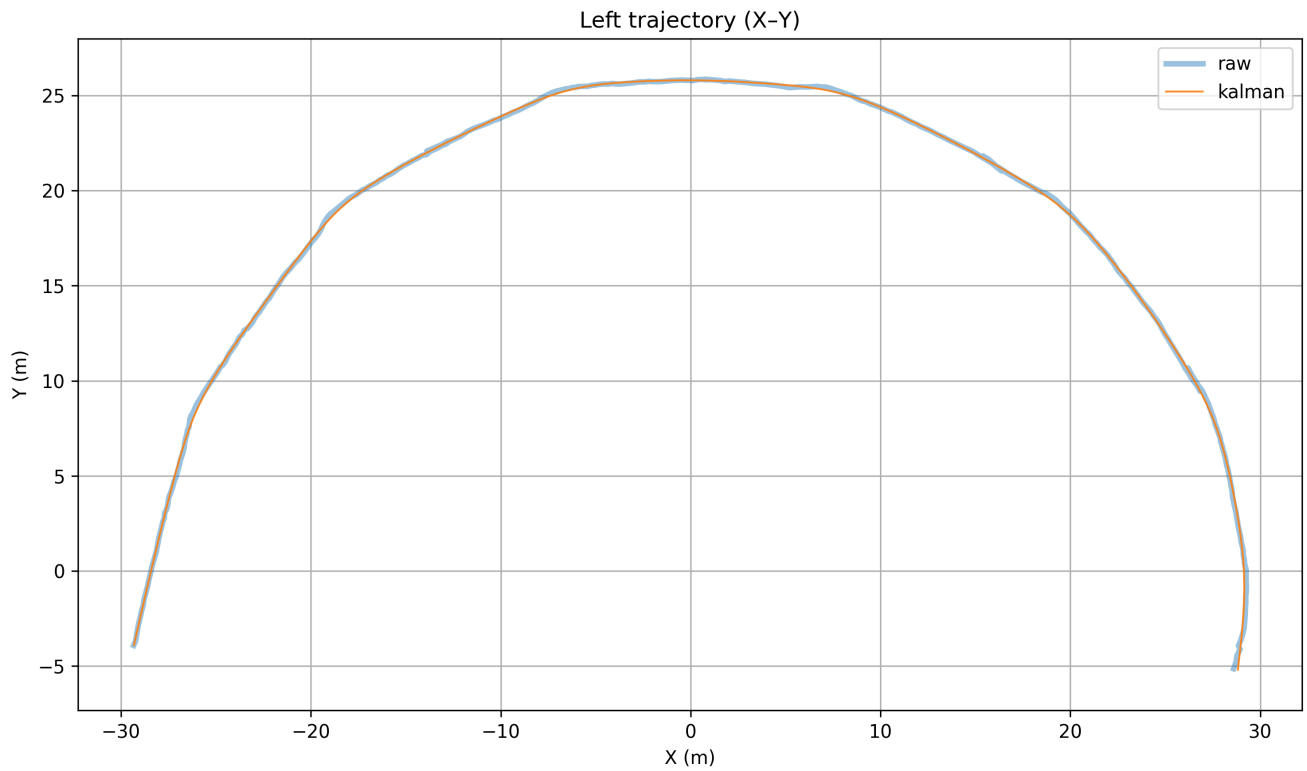


Figure 29: Full trajectory, Kalman smoothing vs. Apex observed data.

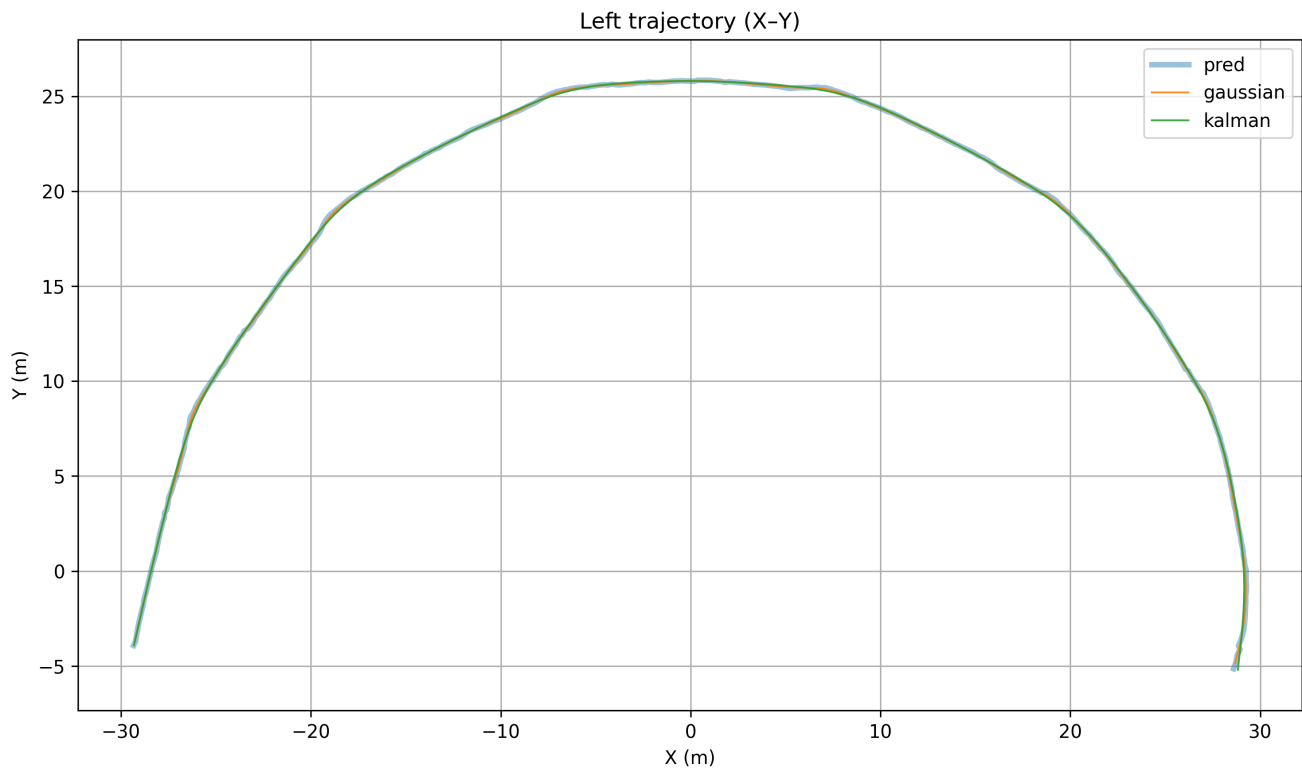


Figure 30: Combined smoothing comparison of the full trajectory, Gaussian smoothing vs. Kalman smoothing vs. Apex observed data.

Across the entire trajectory, both smoothers reduce local noise without distorting the overall trajectory.

6 Discussion and Conclusion

The results of the experiments show that the Apex multiple camera system already provides accurate positional data of the speed skater with both an RMSE and MAE of ≈ 5 cm. Thus, across all reconstructions overall, the RMSE and MAE remained very low, which indicates that the original observations captured by the Apex system are already close to the annotated ground truth trajectory. By computing the averaged tail-to-head or head-to-tail pairs, a continuous and realistic trajectory is obtained in the case of gaps. The additional linear and spline fitting specifically applied to the short reconstructed merged overlap region did not produce any improvements. This confirms that the overlap regions are too short for higher order fitting to have measurable effect on accuracy.

The Gaussian and Kalman filters increase the RMSE, which is expected because these smoothers modify already accurate observations. Therefore, their role is one of the visual refinements instead of minimizing error. The Kalman filter generates smooth trajectories while preserving small variations in speed and direction, where the Gaussian filter produces a motion trajectory that visually aligns even closer to the ground truth and appears very stable, with almost no visible noise. Because the Apex system observations are already very close to the ground truth, it is natural that any additional smoothing introduces a slightly higher RMSE and MAE. This shows that the smoother mainly removes small noise that is difficult to detect numerically in the (x, y) world coordinates, but becomes visible in the visual representation. This can be observed in Figure 25, where the predicted data already align closely with the annotated ground truth data and show minimal noise.

Overall, these findings confirm that the Apex multiple camera system captures motion with minimal noise and that further smoothing mainly improves visual smoothness rather than positional accuracy.

6.1 Limitations

The main limitation of this research is that all smoothing parameters were tuned for a single athlete and a single test session. Therefore, results might vary for other speeds and camera calibration conditions. Another limitation is the length of the overlap regions, which limits the extent which fitting or filtering methods can show an effect. Because each overlap consists of only a few frames, higher-order fitting such as the spline fit adds little benefit beyond simple averaging. Additionally, the outer lane data set lacks ground truth annotations, which prevents evaluation using the RMSE and MAE for the interpolation performance.

An additional observation from the experiments is that the raw Apex system data from the outer lane cameras appears to have more noise than those from the inner lane. The reason for this difference is unclear, but may relate to the angle of the camera or calibration sensitivity near the wider curve of the speed skating arena. If the inner lane data had contained the same level of noise as the outer lane observations, it is reasonable to assume that the Kalman and Gaussian filters would have produced a larger improvement both visually and in accuracy.

6.2 Future Work

For future research, a possible direction is to extend the current two-dimensional world coordinate reconstruction in the Apex multiple camera system to a full synchronized three-dimensional Apex multiple camera system setup. Huang [HSGY25] proposed a bundle adjustment framework that jointly optimizes camera timing, orientation and trajectory parameters to reconstruct accurate three-dimensional motion from asynchronous cameras. The approach they used allows robust trajectory estimation even when the cameras are not perfectly synchronized in time or are misaligned. Adapting this concept to the Thialf speed skating arena could enable a fully synchronized trajectory reconstruction system, where all six Apex cameras are calibrated and optimized at the same time. Additionally, this method could support real time multiple view visualization, giving coaches and analysts synchronized perspectives of a race or training from any virtual angle.

6.3 Conclusion

This research evaluated linear and spline interpolation together with the Kalman filter and Gaussian filter for reconstructing trajectories of speed skaters in the Apex multiple camera system. The results show:

- The averaged merging method already provides accurate and continuous trajectories between cameras in the inner lane.
- Linear and spline fitting offer no advantage for the short overlap regions present in the data set.
- The Kalman and Gaussian filter mainly improve visual quality by reducing noise and ensuring continuity.

The reconstruction of both data sets was achieved successfully. For the inner lane trajectory, the reconstructed overlap regions aligned closely with the manually annotated ground truth, showing minimal errors. For the outer lane, where no ground truth was available, visual evaluation confirmed that the reconstructed trajectories were smooth, realistic and consistent. Hence, the reconstruction pipeline developed in this research effectively restores motion in both camera configurations.

The Apex multiple camera system's object detection already delivers high quality positional data. Additional smoothing is only used for visual interpretation and analytical purposes. For future development, a focus could be on three dimensional reconstruction and synchronization, which could further improve temporal alignment and enable real time visualization of the skating performance.

References

- [DHPJV06] W. Du, J. Hayet, J. Piater-Justus, and J. Verly. Collaborative multi-camera tracking of athletes in team sports. *Workshop on computer vision based analysis in sport environments*, 2, 2006.
- [dR24] M. de Rooij. Videos-based tracking of a speed skater in the corner. Master’s thesis, Leiden University, LIACS, 2024.
- [FB81] M.A. Fischler and R.C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Association for Computing Machinery*, 24, 1981.
- [HSGY25] H. Huang, Y. Shang, B. Guan, and Q. Yu. 3d trajectory reconstruction of moving points based on asynchronous cameras. *Acta Mechanica Sinica*, 41, 2025.
- [Kal60] R.E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82, 1960.
- [KdRM⁺25] A. Knobbe, M. de Rooij, R. Meerhof, G. Toussaint, R. Roos, J.W. Palenstijn, S. Amavarapu, and A. de Leeuw. An ai-based camera system for detailed speed and positional tracking of elite speed skaters in the corners. *(Submitted to) Journal of Sports Science*, 2025.
- [KK23] C. Kontos and D. Karlis. Football analytics based on player tracking data using interpolation techniques for the prediction of missing coordinates. *Statistica Applicata, Italian Journal of Applied Statistics*, 2023.
- [KOH⁺17] A. Knobbe, J. Orié, N. Hofman, B. van der Burgh, and R. Cachucho. Sports analytics for professional speed skating. *Data Mining and Knowledge Discovery*, 31:1–31, 2017.
- [KP25] H. Kordestani and E. Pegah. A study on the direct application of the gaussian kernel smoothing filter for bridge health monitoring. *Infrastructures*, 10, 2025.
- [LLJD15] Q. Li, R. Li, K. Ji, and W. Dai. Kalman filter and its application. *IEEE*, 2015.
- [MSBB⁺15] S. Moritz, A. Sardá, T. Bartz-Beielstein, M. Zaefferer, and J. Stork. Comparison of different methods for univariate time series imputation in r. *ArXiv*, 2015.
- [PHMS21] S. Pathak, M. He, S. Malinchik, and S. Sobolevsky. Pattern ensembling for spatial trajectory reconstruction. *arXiv*, 2021.